9.4

Administración de IBM MQ

Nota

Antes de utilizar esta información y el producto al que da soporte, lea la información en “Avisos” en la página 587.

Esta edición se aplica a la versión 9 release 4 de IBM® MQ y a todos los releases y modificaciones posteriores hasta que se indique lo contrario en nuevas ediciones.

Contenido

Administración de IBM MQ

Para administrar los gestores de colas de IBM MQ y los recursos asociados, elija el método que prefiera de un conjunto de tareas que puede utilizar para activar y gestionar estos recursos.

Acerca de esta tarea

Puede administrar los objetos de IBM MQ de forma local o remota:

Administración local

La administración local es aquella en la que las tareas administrativas se realizan en cualquier gestor de colas definido en el sistema local. Puede acceder a otros sistemas, por ejemplo mediante el programa de emulación de terminal **telnet** de TCP/IP, y llevar a cabo la administración allí. En IBM MQ, esto puede considerarse como administración local porque no hay ningún canal implicado, es decir, la comunicación la gestiona el sistema operativo.

Para obtener más información, consulte “Trabajar con objetos de IBM MQ locales” en la página 131.

Administración remota

IBM MQ soporta la administración desde un único punto de contacto a través de una administración remota. La administración remota permite ejecutar mandatos desde el sistema local que se procesan en otro sistema y también se aplica a IBM MQ Explorer. Por ejemplo, puede emitir un mandato remoto para cambiar una definición de cola en un gestor de colas remoto. No es necesario que se conecte a ese sistema, aunque sí que necesita tener definidos los canales apropiados. El gestor de colas y el servidor de mandatos del sistema de destino deben estar ejecutándose.

Algunos mandatos no se pueden emitir de esta manera, especialmente los que crean o inician gestores de colas y los que inician servidores de mandatos. Para realizar este tipo de tarea, hay que iniciar la sesión en el sistema remoto y ejecutar los mandatos desde allí o crear un proceso que pueda ejecutarlos automáticamente. Esta restricción también se aplica a IBM MQ Explorer.

Para obtener más información, consulte “Trabajar con objetos de IBM MQ remotos” en la página 200.

Existe una serie de métodos diferentes que puede utilizar para crear y administrar los gestores de colas y sus recursos relacionados en IBM MQ. Estos métodos incluyen las interfaces de línea de mandatos, las interfaces gráficas de usuario y una API de administración.

Existen distintos conjuntos de mandatos que puede utilizar para administrar IBM MQ en función de la plataforma:

- “Mandatos de control de IBM MQ” en la página 8
- “mandatos de script de IBM MQ (MQSC)” en la página 8
- “Formatos de mandato programable (PCF)” en la página 8
- La administrative REST API
- IBM i “Lenguaje de control (CL) de IBM i” en la página 10

También hay las otras opciones siguientes para crear y gestionar los objetos de IBM MQ:

- Windows Linux “IBM MQ Explorer” en la página 9
- “IBM MQ Console” en la página 9
- Windows “Microsoft Cluster Service (MSCS)” en la página 10

z/OS Para obtener información sobre las interfaces y opciones de administración en IBM MQ for z/OS, consulte “Administering IBM MQ for z/OS” en la página 461.

Puede automatizar algunas tareas de administración y supervisión para ambos gestores de colas, locales y remotos, mediante mandatos PCF. Estos mandatos también se pueden simplificar utilizando la Interfaz de administración de IBM MQ (MQAI) en algunas plataformas. Si desea más información sobre cómo

automatizar las tareas de administración, consulte “Automatización de la administración de IBM MQ utilizando mandatos PCF” en la página 25.

Conceptos relacionados

Visión general técnica de IBM MQ

Tareas relacionadas

Planificación

Configuración

Referencia relacionada

Comparativa de conjuntos de mandatos

Formas de administrar los gestores de colas de IBM MQ y los recursos asociados

Puede administrar gestores de colas de IBM MQ y recursos asociados utilizando mandatos de control de IBM MQ , mandatos de script de IBM MQ (MQSC), formatos de mandatos programables (PCF), administrative REST API, IBM MQ Consoley IBM MQ Explorer. Para IBM i también puede utilizar IBM i Control Language, y para Windows también puede utilizar Microsoft Cluster Service (MSCS).

Mandatos de control de IBM MQ

Multi

Los mandatos de control ofrecen una manera de realizar una serie de tareas de administración de IBM MQ. Para AIX, Linux®, and Windows, emita estos mandatos en la línea de mandatos del sistema. Para IBM i, emita estos mandatos dentro de un Qshell. Consulte “Administración de IBM MQ for Multiplatforms utilizando mandatos de control” en la página 10.

mandatos de script de IBM MQ (MQSC)

Utilice mandatos MQSC para gestionar objetos de gestor de colas, incluido el propio gestor de colas, colas, definiciones de proceso, canales, canales de conexión de cliente, escuchas, servicios, y objetos de información de autenticación.

ALW En AIX, Linux, and Windows, abra un indicador de mandatos de **runmqsc** y, a continuación, emita mandatos MQSC a un gestor de colas local o remoto desde dicho indicador. Puede hacerlo de forma interactiva, o puede ejecutar una secuencia de mandatos desde un archivo de texto ASCII. Para obtener más información, consulte “Ejecución interactiva de mandatos MQSC en runmqsc” en la página 18 y “Ejecución de mandatos MQSC desde archivos de texto en runmqsc” en la página 23.

IBM i En IBM i, cree una lista de mandatos en un archivo de script y, a continuación, ejecute el archivo utilizando el mandato **STRMQMMQSC** . Para obtener más información, consulte “Administración utilizando mandatos MQSC en IBM i” en la página 400.

z/OS En z/OS, los mandatos MQSC se pueden emitir desde una serie de orígenes, en función del mandato. Para obtener más información, consulte “Sources from which you can issue MQSC and PCF commands on IBM MQ for z/OS” en la página 461.

Formatos de mandato programable (PCF)

Los formatos de mandato programable (PCF) definen mensajes de mandato y de respuesta que se pueden intercambiar entre un programa y cualquier gestor de colas (que admita PCF) en una red. Puede utilizar los mandatos PCF en el programa de aplicación de gestión de sistemas para la administración de objetos de IBM MQ: los objetos de información de autenticación, los canales, los escuchas de canales, las listas de nombres, las definiciones de proceso, los gestores de colas, las colas, los servicios y las clases de almacenamiento. La aplicación puede operar desde un único punto de la red para comunicar

información de mandato y de respuesta a cualquier gestor de colas, local o remoto, utilizando el gestor de colas local.

Si desea más información sobre los PCF, consulte "Introducción a los Formatos de mandato programable de IBM MQ" en la página 26.

Para la definición de los PCF y las estructuras para los mandatos y las respuestas, consulte Referencia de formatos de mandato programable.

administrative REST API

administrative REST API proporciona una interfaz RESTful que puede utilizar para administrar IBM MQ. Cuando utiliza administrative REST API, se invocan métodos HTTP en un URL que representa un objeto de IBM MQ. Por ejemplo, puede solicitar información sobre las instalaciones de IBM MQ utilizando el método HTTP GET en el URL siguiente:

```
https://localhost:9443/ibmmq/rest/v1/admin/installation
```

Puede utilizar administrative REST API con la implementación de HTTP/REST de un lenguaje de programación o utilizando herramientas como cURL o un complemento de navegador de cliente REST.

For more information, see The administrative REST API

IBM MQ Console

Puede utilizar IBM MQ Console para administrar IBM MQ desde un navegador web.

Para obtener más información, consulte "Administración utilizando IBM MQ Console" en la página 93.

IBM MQ Explorer

Utilizando IBM MQ Explorer, puede realizar las acciones siguientes:

- Definir y controlar diversos recursos como, por ejemplo, gestores de colas, colas, definiciones de proceso, listas de nombres, canales, canales de conexión de cliente, escuchas, servicios y clústeres.
- Iniciar o detener un gestor de colas local y los procesos asociados al mismo.
- Ver gestores de colas y sus objetos asociados en su estación de trabajo o desde otras estaciones de trabajo.
- Comprobar el estado de gestores de colas, clústeres y canales.
- Comprobar qué aplicaciones, usuarios o canales tienen una cola determinada abierta, a partir del estado de la cola.

En sistemas Windows y Linux for x86-64 , puede iniciar IBM MQ Explorer utilizando el menú del sistema o el archivo ejecutable `MQExplorer` .

En Linux, para iniciar correctamente IBM MQ Explorer, debe poder grabar un archivo en el directorio de inicio y debe existir un directorio de inicio.

Para obtener más información, consulte "Administración utilizando IBM MQ Explorer" en la página 122.

Puede utilizar IBM MQ Explorer para administrar gestores de colas remotos en otras plataformas incluyendo z/OS.

A partir de IBM MQ 9.3.0, IBM MQ Explorer se ha eliminado del paquete de instalación de IBM MQ . Permanece disponible como descarga independiente y se puede instalar desde la descarga autónoma de IBM MQ Explorer disponible en Fix Central. Para obtener más información, consulte Instalación y desinstalación de IBM MQ Explorer como una aplicación autónoma en Linux y Windows.

Lenguaje de control (CL) de IBM i

IBM i

Esta es la forma preferida de emitir mandatos de administración a IBM MQ for IBM i. Los mandatos se pueden emitir en la línea de mandatos o bien escribiendo un programa de CL. Estos mandatos realizan funciones similares a los mandatos PCF, pero el formato es diferente. Los mandatos CL se han diseñado de forma exclusiva para los servidores y las respuestas CL son legibles para las personas, mientras que los mandatos PCF son independientes de la plataforma y ambos formatos, de mandato y de respuesta, están pensados para el uso del programa.

Para obtener todos los detalles del lenguaje de control (CL) de IBM i , consulte "Gestión de IBM MQ for IBM i utilizando mandatos CL" en la página 385 y IBM MQ for IBM i mandatos CL.

Microsoft Cluster Service (MSCS)

Windows

Microsoft Cluster Service (MSCS) le permite conectar servidores a un *clúster*, lo que proporciona una mayor disponibilidad de datos y aplicaciones y facilita la gestión del sistema. MSCS puede detectar y recuperarse automáticamente de los errores del servidor o de las aplicaciones.

Es importante no confundir los clústeres en el sentido de MSCS con los clústeres de IBM MQ. La diferencia es la siguiente:

Clústeres de IBM MQ
: Son grupos de dos o más gestores de colas en uno o varios sistemas que proporcionan una interconexión automática y permiten que se compartan las colas entre los mismos para fines de equilibrio de carga y redundancia.

Clústeres de MSCS
: Son Grupos de sistemas, conectados entre sí y configurados de tal modo que si se produce un error en uno de ellos, MSCS ejecuta una *sustitución por anomalía*, transfiere los datos de estado de las aplicaciones del sistema anómalo a otro sistema del clúster y reinicia su ejecución en el mismo.

El tema Soporte de Microsoft Cluster Service (MSCS) proporciona información detallada sobre cómo configurar el sistema IBM MQ for Windows para utilizar MSCS.

Tareas relacionadas

"Administración de IBM MQ utilizando mandatos MQSC" en la página 12
Puede utilizar mandatos MQSC para gestionar objetos de gestor de colas, incluido el propio gestor de colas, colas, definiciones de proceso, canales, canales de conexión de cliente, escuchas, servicios, listas de nombres, clústeres y objetos de información de autenticación. Los mandatos MQSC están disponibles en todas las plataformas.

Referencia relacionada

Referencia de administración

Multi Administración de IBM MQ for Multiplatforms utilizando mandatos de control

Los mandatos de control ofrecen una manera de realizar una serie de tareas de administración de IBM MQ. Para AIX, Linuxy Windows, emita estos mandatos en la línea de mandatos del sistema. Para IBM i, emita estos mandatos dentro de un Qshell.

Antes de empezar

Al utilizar los mandatos de control que funcionan en un gestor de colas, debe utilizar el mandato desde la instalación asociada con el gestor de colas con el que está trabajando.

Cuando se utilizan mandatos de control que funcionan en un gestor de colas configurado para utilizar la autenticación de conexiones con CHCKLOCL(REQUIRED) y se observa un error de conexión, puede:

- Proporcionar un ID de usuario y una contraseña si el mandato de control lo permite.
- Utilizar equivalentes MQSC de los mandatos de control allí donde existan.
- Iniciar el gestor de colas utilizando la opción -ns, cuando deben ejecutarse mandatos de control que no pueden conectarse.

Nota: Distintas plataformas pueden aceptar argumentos de mandato especificados en un orden diferente. En concreto, esto significa que es posible que los mandatos que funcionan en Linux no funcionen en otras plataformas. Por este motivo, siempre debe especificar los argumentos tal como se especifican en los diagramas de sintaxis.

Para obtener una lista completa de los mandatos de control, consulte la referencia de mandatos de control deIBM MQ.

Procedimiento

- Linux AIX

  Utilice mandatos de control en sistemas AIX and Linux .

  En sistemas IBM MQ for AIX or Linux, especifique mandatos de control en una ventana de shell.

  Si desea emitir mandatos de control, su ID de usuario debe ser miembro del grupo mqm para la mayoría de los mandatos de control. Para obtener más información, consulte Autorización para administrar IBM MQ en AIX, Linux, and Windows. Además, tenga en cuenta la información específica del entorno. para la plataforma o plataformas que utilice su empresa.

  En los entornos UNIX and Linux, los mandatos de control, incluido el nombre del mandato propiamente dicho, los indicadores y los argumentos que pueda haber distinguen entre mayúsculas y minúsculas. Por ejemplo, en el mandato:

  ```
  crtmqm -u SYSTEM.DEAD.LETTER.QUEUE jupiter.queue.manager
  ```

  - El nombre del mandato debe ser `crtmqm`, no `CRTMQM`.
  - El indicador debe ser `-u`, no `-U`.
  - La cola de mensajes no entregados es `SYSTEM.DEAD.LETTER.QUEUE`.
  - El argumento se ha especificado como `jupiter.queue.manager`, lo que es distinto de `JUPITER.queue.manager`.

  Preste atención y escriba los mandatos exactamente igual que en los ejemplos.

- Windows

  Utilice mandatos de control en sistemas Windows .

  En IBM MQ for Windows, especifique los mandatos de control en un indicador de mandatos.

  Si desea emitir mandatos de control, su ID de usuario debe ser miembro del grupo mqm para la mayoría de los mandatos de control. Para obtener más información, consulte Autorización para administrar IBM MQ en AIX, Linux, and Windows. Además, tenga en cuenta la información específica del entorno. para la plataforma o plataformas que utilice su empresa.

  Los mandatos de control y sus distintivos no distinguen entre mayúsculas y minúsculas, pero los argumentos en dichos mandatos como, por ejemplo, nombres de cola y nombres de gestor de colas, sí distinguen entre mayúsculas y minúsculas.

  Por ejemplo, en el mandato:

  ```
  crtmqm /u SYSTEM.DEAD.LETTER.QUEUE jupiter.queue.manager
  ```

  - El nombre del mandato puede entrarse en mayúsculas o minúsculas o en una combinación de ambas. Todos ellos son válidos: `crtmqm`, `CRTMQM`y `CRTmqm`.
  - El indicador puede entrarse como `-u`, `-U`, `/u` o `/U`.

  - SYSTEM.DEAD.LETTER.QUEUE y `jupiter.queue.manager` deben especificarse exactamente como se muestra.
- IBM i

  Utilice mandatos de control en sistemas IBM i .

  En IBM MQ for IBM i, ejecute los mandatos de control desde un entorno Qshell. Para utilizar el Qshell, escriba STRQSH en la línea de mandatos de IBM i . Puede salir y volver a la línea de mandatos en cualquier momento pulsando F3.

  Un pequeño número de mandatos de control no están soportados en IBM i. Por ejemplo, los mandatos de varias instalaciones no están soportados porque no puede tener más de una copia de IBM MQ en un sistema IBM i . Los mandatos que no están soportados en IBM i se marcan como ALW en la referencia de mandatos de control de IBM MQ.

Referencia relacionada

Referencia de los mandatos de control de IBM MQ

Administración de IBM MQ utilizando mandatos MQSC

Puede utilizar mandatos MQSC para gestionar objetos de gestor de colas, incluido el propio gestor de colas, colas, definiciones de proceso, canales, canales de conexión de cliente, escuchas, servicios, listas de nombres, clústeres y objetos de información de autenticación. Los mandatos MQSC están disponibles en todas las plataformas.

Acerca de esta tarea

Los mandatos MQSC disponibles se detallan en la Referencia de mandatos MQSC.

El modo en que se emiten mandatos MQSC depende de la plataforma.

- ALW En AIX, Linux, and Windows, emita mandatos MQSC para un gestor de colas desde el indicador de mandatos de **runmqsc** . Puede utilizar este indicador de mandatos de varias maneras:
  - De forma interactiva, emitiendo mandatos MQSC desde un teclado. Consulte “Ejecución interactiva de mandatos MQSC en runmqsc” en la página 18.
  - Emisión de mandatos MQSC desde un archivo de texto ASCII. Consulte “Ejecución de mandatos MQSC desde archivos de texto en runmqsc” en la página 23.
  - Emisión de mandatos MQSC en un gestor de colas remoto. Consulte “Emisión de mandatos MQSC en un gestor de colas remoto” en la página 206.
- IBM i En IBM i, cree una lista de mandatos en un archivo de script y, a continuación, ejecute el archivo utilizando el mandato **STRMQMMQSC** . Para obtener más información, consulte “Administración utilizando mandatos MQSC en IBM i” en la página 400.
- z/OS En z/OS, los mandatos MQSC se pueden emitir desde una serie de orígenes, en función del mandato. Para obtener más información, consulte “Sources from which you can issue MQSC and PCF commands on IBM MQ for z/OS” en la página 461.

Procedimiento

- “Sintaxis del mandato MQSC” en la página 13
- “MQSC: caracteres especiales y valores genéricos” en la página 15
- “Ejecución interactiva de mandatos MQSC en runmqsc” en la página 18
- “Ejecución de mandatos MQSC desde archivos de texto en runmqsc” en la página 23
- “Configuración automática de un script de mandato de script de WebSphere MQ en el inicio” en la página 24

Tareas relacionadas

Resolución de problemas con mandatos MQSC

Referencia relacionada

runmqsc (ejecutar mandatos MQSC)

Sintaxis del mandato MQSC

Puede utilizar mandatos MQSC para gestionar objetos del gestor de colas. Los mandatos MQSC están disponibles en todas las plataformas. Algunos elementos de la sintaxis de comandos son específicos de la plataforma.

Secuencia de parámetros

Cada mandato empieza por un parámetro primario (un verbo) y a éste le sigue un parámetro secundario (un sustantivo). Después le sigue el nombre o nombre genérico del objeto (entre paréntesis) si es que lo hay, como suele ocurrir en la mayoría de los mandatos. A continuación, los parámetros pueden aparecer, normalmente, en cualquier orden; si un parámetro tiene un valor correspondiente, éste debe aparecer directamente después del parámetro con el que está relacionado.

Nota: z/OS En z/OS, no es necesario que el parámetro secundario sea el segundo.

Espacios en blanco y comas

Las palabras clave, los paréntesis y los valores pueden estar separados por el número de blancos y de comas que se desee. Una coma mostrada en los diagramas de sintaxis siempre puede ser reemplazada por uno o más espacios en blanco. Debe haber como mínimo un blanco inmediatamente antes de cada parámetro (después del parámetro primario) excepto en z/OS .

Al principio o final de un mandato puede aparecer un número cualquiera de espacios en blanco, así como entre parámetros, signos de puntuación y valores. Por ejemplo, el mandato siguiente es válido:

```
ALTER QLOCAL ('Account' )      TRIGDPTH ( 1)
```

Los blancos dentro de un par de comillas son significativos.

Las comas adicionales pueden aparecer en cualquier lugar donde estén permitidos los espacios en blanco y se tratan como si fueran espacios en blanco (a menos, por supuesto, que estén dentro de series entre comillas).

Parámetros repetidos

Los parámetros repetidos no están permitidos. Repetir un parámetro con su versión "NO", como en REPLACE NOREPLACE, tampoco está permitido.

Series y comillas simples

Las series que contienen espacios en blanco, caracteres en minúsculas o caracteres especiales deben estar entre comillas simples, a menos que se cumpla una de las condiciones siguientes:

- Los caracteres especiales son uno o varios de los caracteres siguientes:
  - Punto (.)
  - Barra inclinada (/)
  - Subrayado (_)
  - Signo de porcentaje (%)
- z/OS El mandato se emite desde los paneles de control y operaciones de IBM MQ for z/OS.

- La serie es un valor genérico que acaba con un asterisco. (en IBM i estos se deben especificar entre comillas simples)
- La serie es un solo asterisco, por ejemplo, TRACE(*) (en IBM i estas se deben especificar entre comillas simples)
- La serie es una especificación de rango que contiene dos puntos, por ejemplo, CLASS(01:03)

Si la cadena propiamente dicha contiene comillas simples, las comillas simples se representan con dos comillas simples.

Multi En Multiplatforms, una serie que no contiene caracteres (es decir, dos comillas simples sin espacio entre medias) se interpreta como un espacio en blanco entre comillas simples, es decir, se interpreta del mismo modo que (''). La excepción a esto es si el atributo que se utiliza es uno de los atributos siguientes, cuando dos comillas simples sin espacio se interpretan como una serie de longitud cero:

- `TOPICSTR`
- `SUB`
- `USERDATA`
- `SELECTOR`

z/OS En z/OS, si desea un espacio en blanco entre comillas simples, debe especificarlo como tal (''). Una serie que no contiene caracteres ('') es lo mismo que especificar ().

Los espacios en blanco al final en atributos de serie que se basan en tipos MQCHARV como, por ejemplo, SELECTOR, los datos de subusuario, son significativos, es decir, `'abc  '` no es igual que `'abc'`.

Paréntesis vacíos

Un signo de abrir paréntesis seguido de un signo de cerrar paréntesis, sin ninguna información significativa entre ellos, no es válido, excepto cuando se indique de forma específica. Por ejemplo, la serie siguiente no es válida:

```
NAME ( )
```

Minúsculas y mayúsculas

Las palabras clave no son sensibles a las mayúsculas y minúsculas: AltER, alter y ALTER son todas correctas.

Lo que no esté entre comillas se convertirá en mayúsculas.

Sinónimos

Se han definido sinónimos para algunos parámetros. Por ejemplo, DEF siempre es un sinónimo de DEFINE; por lo tanto, DEF QLOCAL es válido. Sin embargo, los sinónimos no son simplemente palabras abreviadas; DEFI no es un sinónimo de DEFINE.

Nota: No hay ningún sinónimo para el parámetro DELETE. Esto es así para evitar borrar accidentalmente objetos al utilizar DEF, el sinónimo de DEFINE.

Caracteres especiales

Los mandatos MQSC utilizan determinados caracteres especiales que tienen determinados significados. Para obtener más información sobre estos caracteres especiales y cómo utilizarlos, consulte “MQSC: caracteres especiales y valores genéricos” en la página 15.

Tareas relacionadas

Resolución de problemas con mandatos MQSC

Referencia relacionada

runmqsc (ejecutar mandatos MQSC)

MQSC: caracteres especiales y valores genéricos

Algunos caracteres, por ejemplo, barra inclinada invertida (\) y comillas dobles (") tienen significados especiales cuando se utilizan con mandatos MQSC. Algunos caracteres especiales que se pueden utilizar con parámetros pueden tener valores genéricos, pero se deben especificar correctamente.

Preceder barra inclinada invertida (\) y comillas dobles (") con un \, es decir, especifique \\ o \" si desea \ o " en el texto.

Siempre que un parámetro pueda tener un valor genérico, se especifica con un asterisco (*) final, por ejemplo, ABC*. Un valor genérico significa todos los valores que empiezan por; por lo tanto, ABC* significa todos los valores que empiezan por ABC. Si en el valor se utilizan caracteres que requieren comillas, se debe colocar el asterisco dentro de las comillas, por ejemplo, 'abc*'. El asterisco debe ser el último, o el único, carácter del valor.

No se permite usar signos de cerrar interrogación (?) y de dos puntos (:) en los valores genéricos.

Cuando necesite utilizar cualquiera de estos caracteres especiales en un campo (por ejemplo, como parte de una descripción), deberá delimitar la serie completa entre comillas simples.

Tabla 1. Descripciones de caracteres que tienen significados especiales

Carácter	Descripción
	Los espacios en blanco se usan como separadores. Varios espacios en blanco son equivalentes a un solo espacio en blanco, excepto en series que van entre póstrofos ('). Los blancos finales de estos atributos de serie basados en tipos MQCHARV se tratan como significativos.
,	Se utilizan comas como separadores. Varias comas equivalen a una sola coma, excepto en las series encerradas entre apóstrofos (').
'	Un apóstrofo indica el principio o el final de una serie. IBM MQ deja todos los caracteres encerrados entre comillas exactamente como se han introducido. Los apóstrofos incluidos no se incluyen al calcular la longitud de la serie.
''	IBM MQ trata las comillas simples que aparecen dentro de una serie como un carácter al calcular la longitud de la serie y la serie no ha terminado.
=	z/OS En z/OS, un signo de igual indica el inicio del valor de un parámetro que termina con una coma o un espacio en blanco.
(	Un signo de abrir paréntesis indica el principio del valor o la lista de valores de un parámetro.
)	Un signo de cerrar paréntesis indica el final del valor o la lista de valores de un parámetro.
:	Los dos puntos indican un rango inclusivo. Por ejemplo (1:5) significa (1,2,3,4,5). Esta notación solo se puede utilizar en los mandatos **TRACE**.
*	Un asterisco significa todo. Por ejemplo, **DISPLAY TRACE (*)** significa mostrar todos los rastreos y **DISPLAY QUEUE (PAY*)** significa mostrar todas las colas con nombres que empiezan por PAY.

Sintaxis del archivo de entrada MQSC

Si tiene mandatos largos o está utilizando una secuencia determinada de mandatos repetidamente, puede utilizar un archivo de entrada para emitir mandatos MQSC. El contenido del archivo de entrada debe seguir la sintaxis descrita en este tema.

Visión general

Los mandatos MQSC se introducen a través del *dispositivo de entrada estándar*, al que también se hace referencia como `stdin`. Normalmente es el teclado, pero puede especificar que la entrada procede de un archivo de entrada.

Puede utilizar este archivo de entrada con cualquiera de las siguientes herramientas específicas de la plataforma:

- ALW El mandato **runmqsc** en AIX, Linux, and Windows. Consulte "Ejecución de mandatos MQSC desde archivos de texto en runmqsc" en la página 23
- IBM i El mandato **STRMQM** en IBM i. Consulte "Administración utilizando mandatos MQSC en IBM i" en la página 400
- z/OS Los archivos de inicialización CSQINP1, CSQINP2 y CSQINPX o el programa de utilidad de proceso por lotes CSQUTIL en z/OS. Consulte "Sources from which you can issue MQSC and PCF commands on IBM MQ for z/OS" en la página 461

Sintaxis

Sintaxis del archivo de entrada MQSC:

- Por motivos de portabilidad entre los entornos IBM MQ, limite la longitud de línea en los archivos de mandatos MQSC a 72 caracteres.
- Cada mandato debe empezar en una línea nueva.
- Una línea que tenga un asterisco (*) en la primera posición se ignora. Esto se puede usar para insertar comentarios en el archivo.
- Las líneas en blanco se ignoran.
- Un signo más (+) indica que el mandato continúa desde el primer carácter no en blanco de la línea siguiente. Si utiliza + para continuar un mandato, recuerde que debe dejar al menos un espacio en blanco antes del siguiente parámetro (excepto en z/OS donde no es necesario). Los comentarios o las líneas en blanco se descartan cuando el mandato se vuelve a ensamblar en una sola serie.
- Un signo menos (-) indica que el mandato va a continuar desde el principio de la línea siguiente. Los comentarios o las líneas en blanco se descartan cuando el mandato se vuelve a ensamblar en una sola serie.
- Los mandatos MQSC que están contenidos en un mandato Escape PCF (formato de mandato programable) no pueden continuar con el signo más o el signo menos. Todo el mandato debe estar contenido en un solo mandato Escape. Para obtener información sobre los mandatos PCF, consulte "Introducción a los Formatos de mandato programable de IBM MQ" en la página 26.
- En Multiplatforms, y en z/OS para los mandatos emitidos desde el programa de utilidad por lotes CSQUTIL, puede utilizar un carácter de punto y coma (;) para terminar un mandato, incluso si ha especificado un signo más (+) al final de la línea anterior.
- Una línea no debe terminar en un carácter de control del teclado (por ejemplo, un tabulador).
- Si ejecuta el mandato **runmqsc** en modalidad de cliente redirigiendo `stdin` desde un archivo de texto y proporciona el distintivo **-u** para proporcionar credenciales, el mandato **runmqsc** no solicita una contraseña y, en su lugar, la contraseña se lee de `stdin`. Debe asegurarse de que la primera línea de datos proporcionada a través de `stdin` es la contraseña. Esto se puede realizar utilizando herramientas de línea de mandatos como "echo" o "cat" y pasando la contraseña seguida por el script MQSC en el mandato **runmqsc** `stdin`.
- Windows En Windows, si determinados caracteres especiales como el signo de la libra (£) y el NOT lógico (¬) se utilizan en un script de mandato (por ejemplo, como parte de una descripción de objeto), se visualizan de forma distinta en la salida de un mandato como, por ejemplo, **DISPLAY QLOCAL**.

Consulte también "Sintaxis del mandato MQSC" en la página 13.

Ejemplos

El ejemplo siguiente es un extracto de un archivo de mandatos MQSC que muestra el mandato **DEFINE QLOCAL**.

```
DEFINE QLOCAL(ORANGE.LOCAL.QUEUE) REPLACE +
DESCR(' ') +
PUT(ENABLED) +
DEFPRTY(0) +
DEFPSIST(NO) +
GET(ENABLED) +
MAXDEPTH(5000) +
MAXMSGL(1024) +
DEFSOPT(SHARED) +
NOHARDENBO +
USAGE(NORMAL) +
NOTRIGGER;
```

Figura 1. Extracto de un archivo de mandatos MQSC

Cuando el mandato **runmqsc** se completa, se devuelve un informe. El ejemplo siguiente es un extracto de un informe:

```
Starting MQSC for queue manager jupiter.queue.manager.
.
.
12:   DEFINE QLOCAL('ORANGE.LOCAL.QUEUE') REPLACE +
:      DESCR(' ') +
:      PUT(ENABLED) +
:      DEFPRTY(0) +
:      DEFPSIST(NO) +
:      GET(ENABLED) +
:      MAXDEPTH(5000) +
:      MAXMSGL(1024) +
:      DEFSOPT(SHARED) +
:      NOHARDENBO +
:      USAGE(NORMAL) +
:      NOTRIGGER;
AMQ8006: IBM MQ queue created.
:
.
.
```

Figura 2. Extracto de un archivo de informe de mandatos MQSC

También puede utilizar los archivos de mandatos MQSC de ejemplo para ayudarle a crear el archivo de texto:

amqscos0.tst
Definiciones de objetos utilizadas por programas de ejemplo.

amqscic0.tst
Definiciones de colas para transacciones CICS.

Linux AIX En AIX and Linux, estos archivos se encuentran en el directorio *MQ_INSTALLATION_PATH*/samp. *MQ_INSTALLATION_PATH* representa el directorio de alto nivel en el que está instalado IBM MQ.

Windows En Windows, estos archivos se encuentran en el directorio *MQ_INSTALLATION_PATH*\tools\mqsc\samples. *MQ_INSTALLATION_PATH* representa el directorio de alto nivel en el que está instalado IBM MQ.

ALW Ejecución interactiva de mandatos MQSC en runmqsc

En AIX, Linux, and Windows, puede utilizar el indicador de mandatos de **runmqsc** para emitir mandatos MQSC a un gestor de colas de forma interactiva. La ejecución interactiva es especialmente adecuada para pruebas rápidas.

Antes de empezar

Debe utilizar el mandato **runmqsc** desde la instalación asociada al gestor de colas con el que está trabajando. Puede averiguar con qué instalación está asociado un gestor de colas utilizando el mandato `dspmq -o installation`.

Puede hacer que sea más fácil ver que está en un entorno MQSC y ver algunos detalles del entorno actual estableciendo una solicitud de su elección utilizando la variable de entorno **MQPROMPT** . Para obtener más información, consulte "Establecimiento del indicador de mandatos MQSC" en la página 20.

Linux AIX Cuando ejecuta mandatos MQSC de forma interactiva en plataformas AIX and Linux , el indicador de mandatos de **runmqsc** también da soporte a funciones adicionales del editor de línea de mandatos. Consulte "Habilitación de la recuperación y finalización de mandatos, y claves de mandatos Emacs, para runmqsc" en la página 22

Acerca de esta tarea

El mandato **runmqsc** se utiliza para abrir un indicador de mandatos desde el que puede emitir mandatos MQSC. Estos mandatos y su sintaxis se describen en la publicación MQSC commands reference.

Cuando inicie el indicador de mandatos de **runmqsc** tal como se describe en esta tarea, establezca el indicador para que se ejecute en una de las tres modalidades, en función de los distintivos establecidos en el mandato:

- *Modalidad de verificación*, donde los mandatos MQSC se verifican en un gestor de colas loca, pero no se ejecutan.
- *Modalidad directa*, en la que los mandatos MQSC se ejecutan en un gestor de colas local.
- *Modalidad indirecta*, en la que los mandatos MQSC se ejecutan en un gestor de colas remoto.

El procedimiento siguiente establece la solicitud para que se ejecute en modalidad directa. Otras opciones se ilustran en los ejemplos que siguen los pasos principales.

Procedimiento

1. Abra una ventana de mandatos o shell y especifique el mandato siguiente:

   ```
   runmqsc QMgrName
   ```

   Donde *NombreGestorColas* especifica el nombre del gestor de colas que desea que procese los mandatos MQSC. Puede dejar *NombreGestorColas* en blanco para procesar mandatos MQSC en el gestor de colas predeterminado.
2. Escriba en los mandatos MQSC, según sea necesario. Por ejemplo, para crear una cola local llamada ORANGE.LOCAL.QUEUE, especifique el mandato siguiente:

   ```
   DEFINE QLOCAL (ORANGE.LOCAL.QUEUE)
   ```

   En los mandatos que tienen demasiados parámetros y no caben en una sola línea, utilice los caracteres de continuación para indicar que un mandato continúa en la línea siguiente:

   - Un signo menos (-) indica que el mandato debe continuar desde el principio de la siguiente línea.
   - Un signo más (+) indica que el mandato debe continuar desde el primer carácter no en blanco en la línea siguiente.

La entrada de mandato termina con el carácter final de una línea no en blanco que no es un carácter de continuación. También puede terminar explícitamente la entrada de un mandato con un punto y coma (;).

3. Deje de trabajar con mandatos MQSC especificando el mandato siguiente:

```
end
```

De forma alternativa, puede utilizar el mandato **exit** , el mandato **quit** o el carácter EOF para el sistema operativo.

Resultados

Cuando emite mandatos MQSC, el gestor de colas devuelve mensajes de operador que confirman sus acciones o que le indican los errores que ha cometido. Por ejemplo, el mensaje siguiente confirma que se ha creado una cola:

```
AMQ8006: IBM MQ queue created.
```

El mensaje siguiente indica que ha realizado un error de sintaxis:

```
AMQ8405: Syntax error detected at or near end of command segment below:-
AMQ8426: Valid MQSC commands are:

ALTER
CLEAR
DEFINE
DELETE
DISPLAY
END
PING
REFRESH
RESET
RESOLVE
RESUME
START
STOP
SUSPEND
4 : end
```

Estos mensajes se envían al dispositivo de salida estándar, que de forma predeterminada es la pantalla. Si no ha especificado el mandato correctamente, consulte la información de referencia para que el mandato encuentre la sintaxis correcta. Consulte Referencia de mandatos MQSC.

Ejemplo

A continuación se muestran las variantes del mandato `runmqsc` *QMgrName* que se utiliza en los pasos anteriores. Estas variantes crean configuraciones diferentes del indicador de mandatos de **runmqsc** .

- El mandato siguiente utiliza el filtrado de mandatos para pasar un único mandato MQSC al intérprete MQSC.

  En Windows:

  ```
  echo display chstatus(*) | runmqsc QMname
  ```

  En Linux:

  ```
  echo "display chstatus(*)" | runmqsc QMname
  ```

- El mandato siguiente no especifica un nombre de gestor de colas, por lo que los mandatos MQSC se procesan en el gestor de colas predeterminado.

  ```
  runmqsc
  ```

- Este mandato envía mandados al gestor de colas QMREMOTE utilizando el gestor de colas QMLOCAL para enviarlos.

```
runmqsc -w 30 -m QMLOCAL QMREMOTE
```

- Este mandato verifica que la sintaxis del mandato es correcta en un gestor de colas local sin ejecutar los mandatos. Tenga en cuenta que los mandatos que se van a verificar se leen desde un archivo de entrada `myprog.in`.

```
runmqsc -f myprog.in -v QmgrName
```

Para obtener más información sobre cómo trabajar con archivos de entrada y salida, consulte "Ejecución de mandatos MQSC desde archivos de texto en runmqsc" en la página 23.

Qué hacer a continuación

Para obtener todos los detalles de la sintaxis del mandato **runmqsc** , los parámetros opcionales y los códigos de retorno, consulte runmqsc (ejecutar mandatos MQSC).

Tareas relacionadas

"Ejecución de mandatos MQSC desde archivos de texto en runmqsc" en la página 23
Si tiene mandatos largos o está utilizando una secuencia determinada de mandatos repetidamente, puede utilizar un archivo de texto para emitir mandatos MQSC. Puede redirigir `stdin` desde un archivo de texto. También puede redirigir la salida a un archivo.

Referencia relacionada

Referencia de mandatos MQSC

ALW Establecimiento del indicador de mandatos MQSC

En AIX, Linux, and Windows, utilice la variable de entorno **MQPROMPT** para establecer la solicitud que se visualiza al ejecutar el mandato **runmqsc** . Esto hace que sea más fácil ver que está en un entorno MQSC y ver algunos detalles del entorno actual.

Acerca de esta tarea

Puede establecer la solicitud que se muestra cuando se ejecuta el mandato **runmqsc**. La solicitud se inserta cuando el mandato **runmqsc** se ejecuta de forma interactiva y cuando la entrada se redirige a **runmqsc** desde un archivo o desde el dispositivo de entrada estándar (stdin).

Puede incluir texto sin formato en el indicador de mandatos y también puede insertar variables de entorno utilizando la notación +VARNAME+ del mismo modo que las definiciones de objeto de servicio de IBM MQ. Para obtener más información, consulte "Utilización de inserciones sustituibles en definiciones de servicio" en la página 190.

IBM MQ proporciona algunas otras inserciones sustituibles adicionales, que se describen en la tabla siguiente.

Inserción sustituible	Descripción
MQ_HOST_NAME	Nombre de host del sistema
MQ_FILE_SEP	Separador de archivos específico de la plataforma: • Linux AIX En sistemas AIX and Linux , el **MQ_FILE_SEP** es /. • Windows En sistemas Windows , la ubicación de **MQ_FILE_SEP** es \
MQ_PATH_SEP	Separador de vías de acceso específico de la plataforma:

Inserción sustituible	Descripción
	• Linux AIX En sistemas AIX and Linux , el **MQ_PATH_SEP** es :. • Windows En sistemas Windows , la ubicación de **MQ_PATH_SEP** es ;
MQ_DATE_TIME	Fecha y hora del sistema local en un formato YYYY-MM-DD hh:mm:ss.SSS fijo, por ejemplo: `2020-12-25 17:41:37.408`

Notas:

- Los valores de inserciones sustituibles de MQ están relacionados con el sistema host y la instalación de IBM MQ con los que está asociado el mandato **runmqsc**.
- **MQPROMPT** está limitado a un máximo de 256 caracteres cuando se expanden las inserciones. Las expansiones de **MQPROMPT** sobre este valor dan como resultado que toda la serie **MQPROMPT** se trunque sin las expansiones.

Ejemplo

El ejemplo siguiente establece la solicitud en MQSC:

- Linux AIX

```
export MQPROMPT="MQSC"
```

- Windows

```
set "MQPROMPT=MQSC"
```

AIX El ejemplo siguiente establece la variable **MQPROMPT** en un sistema AIX . La solicitud se establece para visualizar un nombre de usuario (tomado de la variable de entorno del sistema asociada), el nombre del gestor de colas y el nombre de host IBM MQ (tomado de las inserciones sustituibles de IBM MQ):

```
sh> export MQPROMPT="+USER+ @ +QMNAME+ @ +MQ_HOST_NAME+> "
sh> runmqsc MY.QMGR
5724-H72 (C) Copyright IBM Corp. 1994, 2024.
Starting MQSC for queue manager MY.QMGR.

myuser @ MY.QMGR @ aix1> DISPLAY QMSTATUS
```

```
C:\ > set "MQPROMPT=+USERNAME+ @ +QMNAME+ @ +MQ_HOST_NAME+> "
C:\ > runmqsc MY.QMGR
5724-H72 (C) Copyright IBM Corp. 1994, 2024.
Starting MQSC for queue manager MY.QMGR.

myuser @ MY.QMGR @ WIN1> DISPLAY QMSTATUS
```

El ejemplo siguiente añade una indicación de fecha y hora a los ejemplos de **MQPROMPT** anteriores, tomados de las inserciones sustituibles de MQ :

```
sh> export MQPROMPT="+MQ_DATE_TIME+ +USER+ @ +QMNAME+ @ +MQ_HOST_NAME+> "
sh> runmqsc MY.QMGR
5724-H72 (C) Copyright IBM Corp. 1994, 2024.
Starting MQSC for queue manager MY.QMGR.

2020-11-24 18:10:00.404 myuser @ MY.QMGR @ aix1> DISPLAY QMSTATUS
```

```
C:\ > set "MQPROMPT=+MQ_DATE_TIME+ +USERNAME+ @ +QMNAME+ @ +MQ_HOST_NAME+> "
C:\ > runmqsc MY.QMGR
5724-H72 (C) Copyright IBM Corp. 1994, 2024.
Starting MQSC for queue manager MY.QMGR.

2020-11-24 18:10:01.007 myuser @ MY.QMGR @ WIN1> DISPLAY QMSTATUS
```

Linux AIX Habilitación de la recuperación y finalización de mandatos, y claves de mandatos Emacs, para `runmqsc`

Utilice el indicador de mandatos de **runmqsc** en AIX and Linux para habilitar la recuperación de mandatos, la finalización de mandatos y las teclas de mandatos de Emacs.

Acerca de esta tarea

En sistemas AIX and Linux , puede hacer que las siguientes funciones adicionales del editor de línea de mandatos estén disponibles en el indicador de mandatos de **runmqsc** :

- Recuperación de mandatos especificados anteriormente utilizando la tecla de flecha arriba y la tecla de flecha abajo
- Capacidad de completar automáticamente la siguiente palabra clave de un mandato utilizando la tecla de tabulación y la barra espaciadora
- Teclas de mandato Emacs o funciones de teclas de mandato similares

Para utilizar estas funciones, la biblioteca de curses debe estar instalada. Si la biblioteca de curses no está instalada en el sistema, el indicador de **runmqsc** no tiene las funciones del editor de línea de mandatos y se muestra un mensaje cuando se inicia el indicador de mandatos de **runmqsc** . El nombre de la biblioteca de curses a instalar dependerá de la plataforma UNIX:

- AIX En AIX, instale `curses`.
- Linux En Linux, instale `ncurses`.

Procedimiento

- Instale `ncurses` o `curses`.

  Nota: El ejemplo siguiente utiliza instrucciones para Linux

  Ejecute el mandato siguiente para buscar los paquetes de `ncurses` existentes:

```
rpm -qa | grep -i ncurses
```

  Los paquetes de `ncurses` necesarios son los siguientes:

```
ncurses-term-6.1-7.20180224.el8.noarch
ncurses-6.1-7.20180224.el8.x86_64
ncurses-base-6.1-7.20180224.el8.noarch
ncurses-c++-libs-6.1-7.20180224.el8.x86_64
ncurses-libs-6.1-7.20180224.el8.x86_64
ncurses-compat-libs-6.1-7.20180224.el8.x86_64
ncurses-devel-6.1-7.20180224.el8.x86_64
```

  Puede instalar todos los paquetes de `ncurses` necesarios listados en el texto anterior ejecutando el mandato siguiente:

```
yum install ncurses*
```

- Personalice los enlaces de teclas de Emacs.

  Puede personalizar las teclas que están enlazadas a los mandatos. Por ejemplo, puede enlazar las teclas a enlaces vi en lugar de los enlaces de teclas de Emacs personalizadas.

Las claves se personalizan editando el archivo `.editrc` que se almacena en el directorio inicial. Para obtener más información, consulte editoric en las páginas FreeBSD man.

- Inhabilite las teclas de mandato de recuperación de mandatos, finalización de mandatos y Emacs.

  Para ello, establezca la variable de entorno **MQ_OVERRIDE_LIBEDIT_LOAD** en TRUE.

  Esta variable de entorno se puede utilizar como método alternativo cuando el indicador de mandatos de **runmqsc** muestra el siguiente mensaje informativo:

```
AMQ8521I: Command completion and history unavailable
```

ALW Ejecución de mandatos MQSC desde archivos de texto en runmqsc

Si tiene mandatos largos o está utilizando una secuencia determinada de mandatos repetidamente, puede utilizar un archivo de texto para emitir mandatos MQSC. Puede redirigir `stdin` desde un archivo de texto. También puede redirigir la salida a un archivo.

Antes de empezar

Esta tarea presupone que ha creado un archivo de texto que contiene los mandatos MQSC que desea ejecutar. Para ver sintaxis detallada y ejemplos de estos archivos, consulte "Sintaxis del archivo de entrada MQSC" en la página 15.

Puede establecer el indicador de mandatos MQSC en una solicitud de su elección utilizando la variable de entorno **MQPROMPT**. Para obtener más información, consulte "Establecimiento del indicador de mandatos MQSC" en la página 20.

Acerca de esta tarea

La entrada para el mandato **runmqsc** se toma del *dispositivo de entrada estándar*, que también se conoce como `stdin`. Normalmente es el teclado, pero puede especificar que la entrada procede de un puerto serie o de un archivo.

La salida para el mandato **runmqsc** es la salida del *dispositivo de salida estándar*, que se también se conoce como `stdout`. Normalmente es una pantalla, pero puede redirigir la salida a un puerto serie o a un archivo.

Procedimiento

1. En un gestor de colas local, verifique que la sintaxis del mandato en el archivo es correcta sin ejecutar los mandatos.

   Utilice el distintivo **-v** en el mandato **runmqsc** , junto con una de las opciones siguientes:

   - Utilice la opción **-f** para identificar el nombre de archivo de texto de entrada. Por ejemplo:

```
runmqsc -f myprog.in -v localQmgrName
```

   No puede especificar un gestor de colas remoto al verificar mandatos. Es decir, no puede especificar el distintivo **-w**.

   El informe que se devuelve es similar al que se muestra en la Figura 2 en la página 17.
2. Cuando la sintaxis del mandato sea correcta, elimine el distintivo **-v** y vuelva a ejecutar el mandato **runmqsc**.

   Tenga en cuenta que ahora puede especificar un gestor de colas remoto.

   - Ejecute (por ejemplo) el mandato siguiente:

```
runmqsc -f myprog.in QmgrName
```

Figura 1 en la página 17 muestra un extracto de un archivo de mandatos como, por ejemplo, `myprog.in` y Figura 2 en la página 17 muestra el extracto correspondiente de la salida de un archivo de informe como, por ejemplo, `results.out`.

Qué hacer a continuación

Para obtener todos los detalles de la sintaxis del mandato **runmqsc** , los parámetros opcionales y los códigos de retorno, consulte runmqsc (ejecutar mandatos MQSC).

Tareas relacionadas

"Establecimiento del indicador de mandatos MQSC" en la página 20
En AIX, Linux, and Windows, utilice la variable de entorno **MQPROMPT** para establecer la solicitud que se visualiza al ejecutar el mandato **runmqsc** . Esto hace que sea más fácil ver que está en un entorno MQSC y ver algunos detalles del entorno actual.

"Ejecución interactiva de mandatos MQSC en runmqsc" en la página 18
En AIX, Linux, and Windows, puede utilizar el indicador de mandatos de **runmqsc** para emitir mandatos MQSC a un gestor de colas de forma interactiva. La ejecución interactiva es especialmente adecuada para pruebas rápidas.

Referencia relacionada

Referencia de mandatos MQSC

Multi Configuración automática de un script de mandato de script de WebSphere MQ en el inicio

Puede configurar el gestor de colas para aplicar automáticamente el contenido de un script MQSC, o conjunto de scripts MQSC, en cada inicio de gestor de colas.

Puede utilizar esta funcionalidad para tener una configuración que se pueda modificar y se reproduzca automáticamente en el siguiente reinicio del gestor de colas. Como ejemplo, si el script o los scripts se encuentran en un disco montado, es posible tener una configuración centralizada en la que se aplica la última versión a cada gestor de colas a medida que se inician.

Un escenario en particular en el que puede ser de utilidad, es para asegurarse de que un clúster uniforme contiene las mismas definiciones en todos los gestores de colas del clúster, teniendo un conjunto único de configuración que todos aplican. Para obtener un ejemplo de esto, consulte Creación de un clúster uniforme.

Antes de empezar

Puede utilizar:

1. Un único script y crear un archivo de texto utilizando los mandatos MQSC.
2. Un conjunto de scripts MQSC:
   - Para identificar un directorio en el que existirán las configuraciones
   - En dicho directorio, cree archivos, cada uno con la extensión `.mqsc`, por ejemplo `queues.mqsc`.

Dado que este script se vuelve a aplicar en cada inicio del gestor de colas, es importante que se puedan reproducir los mandatos. Por ejemplo, un mandato **DEFINE** debe incluir la serie **REPLACE**, de lo contrario, el mandato aparecerá como una anomalía en el segundo inicio del gestor de colas, porque el objeto ya existe.

Tenga en cuenta que en un script de mandato de script de WebSphere MQ, cualquier línea con un `*` antepuesto se trata como un comentario.

Habilitación de la configuración automática de scripts de mandato de script de WebSphere MQ

Importante: No debe emitir mandatos para canales de tipo MQTT, ya que no están soportados para la configuración automática durante el inicio.

Puede configurar un nuevo gestor de colas utilizando el distintivo **-ic** en el mandato **crtmqm** y apuntar a un archivo o directorio específico. El valor suministrado se almacena en el archivo `qm.ini` bajo la stanza AutoConfig, como el atributo **MQSCConfig**.

Para configurar un gestor de colas existente para habilitar la configuración de mandato de script de WebSphere MQ automática, añada el atributo de stanza AutoConfig **MQSCConfig**, que apunte a un archivo o directorio válido. Por ejemplo:

```
AutoConfig:
   MQSCConfig=C:\mq_configuration\uniclus.mqsc
```

¿Cómo funciona el trabajo de configuración automática?

Durante el inicio del gestor de colas, la configuración identificada por el atributo de stanza **AutoConfig MQSCConfig** se pasa a través de la validación **runmqsc** , para garantizar una sintaxis válida y, a continuación, se almacena en el árbol de datos del gestor de colas en el directorio `autocfg` como un único archivo `cached.mqsc`.

Cuando se procesan varios archivos de un directorio, se procesan en orden alfabético y, si contiene un mandato end o quit de MQSC, el resto del contenido de dicho archivo se omite.

Durante el primer inicio del gestor de colas, la incapacidad para leer el archivo o el directorio, o un archivo con sintaxis de mandato de script de WebSphere MQ que no es válida, impide que se inicie el gestor de colas, con un mensaje de error adecuado en el registro cronológico de errores de la consola y del gestor de colas.

En los reinicios posteriores, si el archivo o el directorio al que se apunta es ilegible o contiene una sintaxis de mandato de script de WebSphere MQ no válida, se utiliza el archivo almacenado en memoria caché anteriormente y se destaca esto en un mensaje grabado en el registro cronológico de errores del gestor de colas.

En el punto en que el contenido de `cached.mqsc` se aplica al gestor de colas, cuando se han aplicado todos los mandatos MQSC, el gestor de colas está habilitado para que las aplicaciones se conecten. El registro **runmqsc** de la configuración que se está aplicando se almacena en el directorio de errores del gestor de colas, como un archivo llamado `autocfgmqsc.LOG`.

Además, cualquier mandato de script de WebSphere MQ que no se complete correctamente, se registra en el registro cronológico de errores del gestor de colas, e identifica porqué falla el mandato.

Automatización de la administración de IBM MQ utilizando mandatos PCF

Puede decidir que podría ser beneficioso para su instalación automatizar algunas tareas de administración y supervisión. Puede automatizar tareas de administración para los gestores de colas locales y remotos utilizando mandatos de formato de mandato programable (PCF). En esta sección se presupone que tiene experiencia en la administración de objetos de IBM MQ.

mandatos PCF

Los mandatos PCF (formato de mandato programable) de IBM MQ se pueden utilizar para programar tareas de administración en un programa de administración. De este modo, desde un programa puede manipular objetos de gestores de colas (colas, definiciones de procesos, listas de nombres, canales, canales de conexión de clientes, escuchas, servicios y objetos de información de autenticación), e incluso manipular los mismos gestores de colas.

Los mandatos PCF abarcan el mismo tipo de funciones que las proporcionadas por los mandatos MQSC. Puede escribir un programa que emita mandatos PCF a cualquier gestor de colas de la red desde un solo nodo. De este modo, puede centralizar y automatizar las tareas de administración.

Cada mandato PCF es una estructura de datos que se incluye en la parte de datos de aplicación de un mensaje de IBM MQ. Cada mandato se envía al gestor de colas de destino utilizando la función MQPUT de MQI del mismo modo que cualquier otro mensaje. Si el servidor de mandatos se está ejecutando en el gestor de colas que recibe el mensaje, el servidor de mandatos lo interpreta como un mensaje de mandato y ejecuta el mandato. Para obtener las respuestas, la aplicación emite una llamada MQGET y los datos de respuesta se devuelven en otra estructura de datos. La aplicación puede entonces procesar la respuesta y actuar en conformidad.

Nota: A diferencia de los mandatos MQSC, los mandatos PCF y sus respuestas no están en un formato de texto legible por el usuario.

En resumen, estas son algunas de las cosas necesarias para crear un mensaje de mandato PCF:

Descriptor de mensaje
Es un descriptor de mensaje estándar de IBM MQ, en el que:

- El tipo de mensaje (*MsqType*) es MQMT_REQUEST.
- El formato del mensaje (*Format*) es MQFMT_ADMIN.

Datos de la aplicación
Contienen el mensaje PCF, incluida la cabecera PCF, en el que:

- El tipo de mensaje PCF (*Type*) especifica MQCFT_COMMAND.
- El identificador de mandato especifica el mandato, por ejemplo, *Change Queue* (MQCMD_CHANGE_Q).

Para obtener una descripción completa de las estructuras de datos PCF y cómo implementarlas, consulte “Introducción a los Formatos de mandato programable de IBM MQ” en la página 26.

Atributos de objetos PCF

Los atributos de objetos en PCF no están limitados a ocho caracteres, como sí lo están para los mandatos MQSC. Se muestran en esta guía en cursiva. Por ejemplo, el equivalente PCF de RQMNAME es *RemoteQMgrName*.

PCF de escape

Los PCF de escape son mandatos PCF que contienen mandatos MQSC dentro del texto del mensaje. Los PCF se pueden utilizar para enviar mandatos a un gestor de colas remoto. Para obtener más información sobre los PCF de escape, consulte Escape.

Introducción a los Formatos de mandato programable de IBM MQ

Los formatos de mandato programable (PCF) definen mensajes de mandato y de respuesta que se pueden intercambiar entre un programa y cualquier gestor de colas (que admita PCF) en una red. Los PCF simplifican la administración del gestor de colas y otras tareas de administración de red. Pueden utilizarse para resolver el problema de la compleja administración de redes distribuidas, especialmente cuando las redes crecen en tamaño y complejidad.

Los formatos de mandatos programables están soportados en todas las plataformas IBM MQ .

Problema que resuelven los mandatos PCF

La administración de redes distribuidas puede ser compleja. Los problemas de administración continúan creciendo a medida que las redes presentan mayor tamaño y complejidad.

Como ejemplos de administración específicos de mensajes y colas, cabe citar:

- Gestión de recursos.

Por ejemplo, la creación y supresión de colas.

- Supervisión del rendimiento.

  Por ejemplo, una mayor profundidad de cola o un mayor índice de mensajes.

- Control.

  Por ejemplo, ajustar los parámetros de colas tales como la profundidad máxima de cola, la longitud máxima de los mensajes, y la habilitación e inhabilitación de colas.

- Direccionamiento de mensajes.

  Definición de rutas alternativas a través de una red.

Los mandatos PCF de IBM MQ pueden utilizarse para simplificar la administración del gestor de colas y otras tareas de administración de la red. Los mandatos PCF permiten utilizar una única aplicación para realizar la administración de red desde un único gestor de colas en la red.

¿Qué son los PCF?

Los PCF definen mensajes de mandato y de respuesta que se pueden intercambiar entre un programa y cualquier gestor de colas (que admita PCF) en una red. Puede utilizar los mandatos PCF en el programa de aplicación de gestión de sistemas para la administración de objetos de IBM MQ: los objetos de información de autenticación, los canales, los escuchas de canales, las listas de nombres, las definiciones de proceso, los gestores de colas, las colas, los servicios y las clases de almacenamiento. La aplicación puede operar desde un único punto de la red para comunicar información de mandato y de respuesta a cualquier gestor de colas, local o remoto, utilizando el gestor de colas local.

Cada gestor de colas tiene una cola de administración con un nombre de cola estándar y la aplicación puede enviar mensajes de mandato PCF a dicha cola. Cada gestor de colas tiene también un servidor de mandatos para prestar servicio a los mensajes de mandato desde la cola de administración. Los mensajes de mandatos PCF pueden, por consiguiente, ser procesados por cualquier gestor de colas de la red y los datos de respuesta pueden devolverse a la aplicación mediante la cola de respuesta especificada. Los mensajes de mandatos PCF y de respuesta se envían y reciben utilizando la Interfaz de colas de mensajes (MQI) habitual.

Para obtener una lista de los mandatos PCF disponibles, incluidos sus parámetros, consulte Definiciones de los formatos de mandato programable.

Utilización de los formatos de mandato programable de IBM MQ

Puede utilizar los PCF en un programa de gestión de sistemas para la administración remota de IBM MQ.

Esta sección incluye:

- “Mensajes de mandato PCF” en la página 27
- “Respuestas PCF en IBM MQ” en la página 30
- z/OS “Extended responses” en la página 32
- Reglas de denominación de objetos de IBM MQ
- “Comprobación de autorización para mandatos PCF en IBM MQ” en la página 34

Mensajes de mandato PCF

Los mensajes de mandato PCF constan de una cabecera PCF, parámetros identificados en dicha cabecera y los datos de mensaje definidos por el usuario. Los mensajes se emiten mediante llamadas de la interfaz de cola de mensajes.

Cada mandato y sus parámetros se envían como un mensaje de mandato independiente que contiene una cabecera PCF seguida de varias estructuras de parámetros; para obtener más información sobre la cabecera PCF, consulte MQCFH - Cabecera PCF y para ver un ejemplo de una estructura de parámetros, consulte MQCFST - Parámetro de serie PCF. La cabecera PCF identifica el mandato y las estructuras de parámetro que siguen en el mismo mensaje. Cada estructura de mandato proporciona un parámetro al mandato.

La respuestas a los mandatos, generadas por el servidor de mandatos, tienen una estructura similar. Hay una cabecera PCF, seguida de varias estructuras de parámetros. Las respuestas pueden constar de más de un mensaje, pero los mandatos siempre constan de un único mensaje.

Multi En Multiplatforms, la cola a la que se envían los mandatos PCF siempre se denomina SYSTEM.ADMIN.COMMAND.QUEUE.

z/OS En z/OS, los mandatos se envían a SYSTEM.COMMAND.INPUT, aunque SYSTEM.ADMIN.COMMAND.QUEUE puede ser un alias para él. El servidor de mandatos que presta servicio a esta cola envía las respuestas a la cola definida por los campos *ReplyToQ* y *ReplyToQMgr* en el descriptor de mensaje del mensaje de mandato.

Cómo emitir mensajes de mandatos PCF

Utilice las llamadas normales de la interfaz de cola de mensajes (MQI), MQPUT, MQGET, etc., para colocar y recuperar mensajes de respuestas y de mandatos PCF en y desde sus colas.

Nota:

Asegúrese de que el servidor de mandatos se ejecuta en el gestor de colas de destino para que el mandato PCF se procese en dicho gestor de colas.

Para obtener una lista de los archivos de cabecera suministrados, consulte Archivos COPY, de cabecera, de inclusión y de módulo de IBM MQ.

Descriptor de mensaje para un mandato PCF

El descriptor de mensaje de IBM MQ está completamente documentado en MQMD - Descriptor de mensaje.

Un mensaje de mandato PCF contiene los siguientes campos en el descriptor de mensaje:

Informe
Cualquier valor válido, según sea necesario.

MsgType
Este campo debe ser MQMT_REQUEST para indicar un mensaje que requiere una respuesta.

Caducidad
Cualquier valor válido, según sea necesario.

Comentarios
Se establece en MQFB_NONE.

Multi **Encoding**
Si está enviando a un sistema IBM MQ for Multiplatforms, establezca este campo en la codificación utilizada para los datos del mensaje. Si es necesario se realiza la conversión.

Multi **CodedCharSetId**
Si está enviando a un sistema IBM MQ for Multiplatforms, establezca este campo en el identificador de juego de caracteres codificado utilizado para los datos del mensaje. Si es necesario se realiza la conversión.

Formato
Se establece en MQFMT_ADMIN.

Prioridad
Cualquier valor válido, según sea necesario.

Persistence
Cualquier valor válido, según sea necesario.

MsgId
La aplicación de envío puede especificar cualquier valor, o puede especificarse MQMI_NONE para solicitar al gestor de colas que genere un identificador de mensaje exclusivo.

CorrelId
La aplicación emisora puede especificar cualquier valor, o puede especificarse MQCI_NONE para indicar que no hay identificador de correlación.

ReplyToQ
Nombre de la cola que debe recibir la respuesta.

ReplyToQMgr
Nombre del gestor de colas para la respuesta (o en blanco).

Campos de contexto de mensaje
Estos campos pueden establecerse en cualquier valor válido, según sea necesario. Normalmente, la opción MQPMO_DEFAULT_CONTEXT se utiliza para establecer los campos de contexto de mensaje en los valores predeterminados.

Si utiliza una estructura MQMD versión 2, debe establecer los siguientes campos adicionales:

GroupId
Se establece en MQGI_NONE.

MsgSeqNumber
Se establece en 1.

Desplazamiento
Se establece en 0.

MsgFlags
Se establece en MQMF_NONE.

OriginalLength
Se establece en MQOL_UNDEFINED.

Envío de datos de usuario

Las estructuras PCF también pueden utilizarse para enviar datos de mensaje definidos por el usuario. En este caso, el campo *Format* del descriptor de mensaje debe establecerse en MQFMT_PCF.

Envío y recepción de mensajes PCF en una cola especificada

Envío de mensajes PCF a una cola especificada

Para enviar un mensaje a una cola especificada, la llamada mqPutBag convierte el contenido del paquete especificado en un mensaje PCF y envía el mensaje a la cola especificada. El contenido del paquete permanece intacto después de la llamada.

Como entrada para esta llamada, debe proporcionar:

- Un manejador de conexión MQI.
- Un manejador de objeto para la cola en la que va a colocarse el mensaje.
- Un descriptor de mensaje. Para obtener más información sobre el descriptor de mensaje, consulte MQMD - Descriptor de mensaje.
- Las opciones de transferencia de mensajes utilizando la estructura MQPMO. Para obtener más información acerca de la estructura MQPMO, consulte MQPMO – Opciones de transferir mensaje.
- El manejador del paquete que se convertirá en un mensaje.

  Nota: Si el paquete contiene un mensaje de administración y se utilizó la llamada mqAddInquiry para insertar valores en el paquete, el valor del elemento de datos MQIASY_COMMAND debe ser un mandato INQUIRE reconocido por la MQAI.

Para obtener una descripción completa de la llamada mqPutBag, consulte mqPutBag.

Recepción de los mensajes PCF de una cola especificada

Para recibir un mensaje de una cola especificada, la llamada mqGetBag obtiene un mensaje PCF de una cola especificada y convierte los datos del mensaje en un paquete de datos.

Como entrada para esta llamada, debe proporcionar:

- Un manejador de conexión MQI.
- Un manejador de objeto de la cola de la que se debe leer el mensaje.
- Un descriptor de mensaje. Dentro de la estructura MQMD, el parámetro **Format** debe ser MQFMT_ADMIN, MQFMT_EVENT o MQFMT_PCF.

  Nota: Si el mensaje se recibe dentro de una unidad de trabajo (es decir, con la opción MQGMO_SYNCPOINT) y el mensaje tiene un formato no soportado, la unidad de trabajo puede restituirse. El mensaje se reincorporará en la cola y podrá recuperarse utilizando la llamada MQGET en lugar de la llamada mqGetBag. Para obtener más información sobre el descriptor de mensaje, consulte MQGMO - Opciones de obtención de mensajes.
- Las opciones de obtención de mensajes utilizando la estructura MQPMO. Para obtener más información acerca de la estructura MQGMO, consulte MQMD - Descriptor de mensaje.
- El manejador del paquete para contener el mensaje convertido.

Para obtener una descripción completa de la llamada mqGetBag, consulte mqGetBag.

Respuestas PCF en IBM MQ

En respuesta a cada mandato, el servidor de mandatos genera uno o varios mensajes de respuesta. Un mensaje de respuesta tiene un formato parecido a un mensaje de mandato.

La cabecera PCF tiene el mismo valor de identificador de mandato que el mandato para el que es una respuesta (consulte MQCFH - Cabecera PCF para obtener más detalles). El identificador del mensaje y el identificador de correlación se establecen de acuerdo con las opciones de informe de la solicitud.

Si el tipo de cabecera PCF del mensaje de mandato es MQCFT_COMMAND, sólo se generan respuestas estándar. Estos mandatos sólo están soportados en Multiplatforms. Las aplicaciones antiguas no dan soporte a PCF en z/OS ; El Explorador de IBM MQ Windows es una de estas aplicaciones (sin embargo, el Explorador de IBM WebSphere MQ 6.0 o posterior de IBM MQ no da soporte a PCF en z/OS).

Si el tipo de cabecera PCF del mensaje de mandato es MQCFT_COMMAND_XR, se generan respuestas ampliadas o estándar. Estos mandatos están soportados en z/OS y algunos Multiplatforms. Los mandatos emitidos en z/OS sólo generan respuestas ampliadas.

Si un mandato especifica un nombre de objeto genérico, se devuelve una respuesta independiente en su propio mensaje para cada objeto coincidente. Para la generación de respuestas, un mandato con un nombre genérico se trata como varios mandatos individuales (excepto para el campo de control MQCFC_LAST o MQCFC_NOT_LAST). En caso contrario, un mensaje de mandato genera un mensaje de respuesta.

Determinadas respuestas PCF pueden devolver una estructura aun cuando no se solicite. Esta estructura se muestra en la definición de la respuesta (Definiciones de los formatos de mandato programable) como *siempre devuelta*. Esto se explica porque para estas respuestas es necesario indicar el nombre de los objetos en la respuesta para identificar el objeto al que se aplican los datos.

Descriptor de mensaje para una respuesta

Un mensaje de respuesta tiene los siguientes campos en el descriptor de mensaje:

MsgType
Este campo es MQMT_REPLY.

MsgId
Este campo se genera en el gestor de colas.

CorrelId
Este campo se genera de acuerdo con las opciones de informe del mensaje de mandato.

Formato
Este campo es MQFMT_ADMIN.

Encoding
Se establece en MQENC_NATIVE.

CodedCharSetId
Se establece en MQCCSI_Q_MGR.

Persistence
Igual que en el mensaje de mandato.

Prioridad
Igual que en el mensaje de mandato.

La respuesta se genera con MQPMO_PASS_IDENTITY_CONTEXT.

Multi *Respuestas estándar*

Los mensajes de mandato con el tipo de cabecera MQCFT_COMMAND generan respuestas estándar. Estos mandatos sólo están soportados en Multiplatforms.

Hay tres tipos de respuesta estándar:

- respuesta OK
- Respuesta de error
- Respuesta de datos

respuesta OK

Esta respuesta consiste en un mensaje que empieza con una cabecera de formato de mandatos, con un campo *CompCode* con el valor MQCC_OK o MQCC_WARNING.

Para MQCC_OK, el valor de *Reason* es MQRC_NONE.

Para MQCC_WARNING, *Reason* identifica la naturaleza de la advertencia. En este caso, la cabecera de formato de mandato puede ir seguida de una o más estructuras de parámetro de aviso apropiadas para este código de razón.

En cualquier caso, para un mandato de consulta es posible que sigan varias estructuras de parámetro, como se describe en las siguientes secciones.

Respuesta de error

Si el mandato tiene un error, se envía uno o varios mensajes de respuesta de error (es posible que se envíe más de uno incluso para un mandato que normalmente sólo tendría un mensaje de respuesta). Estos mensajes de error tienen el valor MQCFC_LAST o MQCFC_NOT_LAST establecido, según corresponda.

Cada mensaje de este tipo empieza por una cabecera de formato de respuesta, con MQCC_FAILED como valor de *CompCode* y un campo *Reason* que identifica el error en cuestión. Por lo general, cada mensaje describe un error distinto. Además, cada mensaje tiene ninguna o una estructura de parámetro de error (nunca más de una) después de la cabecera. Esta estructura de parámetros, si existe, es una estructura MQCFIN, con un campo *Parameter* que contiene uno de estos valores:

- MQIACF_PARAMETER_ID

  El campo *Value* de la estructura es el identificador del parámetro que tenía el error (por ejemplo, MQCA_Q_NAME).
- MQIACF_ERROR_ID

Este valor se utiliza con MQRC_UNEXPECTED_ERROR como valor *Reason* (en la cabecera de formato de mandato). El campo *Value* de la estructura MQCFIN es el código de razón inesperado recibido por el servidor de mandatos.

- MQIACF_SELECTOR

  Este valor se produce si una estructura de lista (MQCFIL) enviada con el mandato contiene un selector duplicado o uno que no es válido. El campo *Reason* de la cabecera de formato de mandato identifica el error, y el campo *Value* de la estructura MQCFIN es el valor de parámetro de la estructura MQCFIL del mandato que presentaba el error.

- MQIACF_ERROR_OFFSET

  Este valor se produce cuando se produce un error de comparación de datos en el mandato Sondear canal. El campo *Value* de la estructura es el desplazamiento del error de comparación de Ping Channel.

- MQIA_CODED_CHAR_SET_ID

  Este valor se produce cuando el identificador del conjunto de caracteres codificado del descriptor de mensaje del mensaje de mandato PCF entrante no coincide con el del gestor de colas de destino. El campo *Value* de la estructura es el identificador del conjunto de caracteres codificado del gestor de colas.

El último (o único) mensaje de respuesta de error es una respuesta de resumen, con MQCC_FAILED como valor de *CompCode* y un campo *Reason* con el valor MQRCCF_COMMAND_FAILED. Este mensaje no tiene ninguna estructura de parámetro después de la cabecera.

Respuesta de datos

Esta respuesta consta de una respuesta OK (como se ha descrito anteriormente) para un mandato de consulta. La respuesta OK va seguida por estructuras adicionales que contienen los datos solicitados como se describe en Definiciones de los formatos de mandato programable.

Las aplicaciones no deben depender de que estas estructuras de parámetro adicionales se devuelvan en un orden determinado.

z/OS *Extended responses*

Commands issued on z/OS generate extended responses.

There are three types of extended response:

- Message response, with type MQCFT_XR_MSG
- Item response, with type MQCFT_XR_ITEM
- Summary response, with type MQCFT_XR_SUMMARY

Each command can generate one, or more, sets of responses. Each set of responses comprises one or more messages, numbered sequentially from 1 in the *MsgSeqNumber* field of the PCF header. The *Control* field of the last (or only) response in each set has the value MQCFC_LAST. For all other responses in the set, this value is MQCFC_NOT_LAST.

Any response can include one, or more, optional MQCFBS structures in which the *Parameter* field is set to MQBACF_RESPONSE_SET, the value being a response set identifier. Identifiers are unique and identify the set of responses which contain the response. For every set of responses, there is an MQCFBS structure that identifies it.

Extended responses have at least two parameter structures:

- An MQCFBS structure with the *Parameter* field set to MQBACF_RESPONSE_ID. The value in this field is the identifier of the set of responses to which the response belongs. The identifier in the first set is arbitrary. In subsequent sets, the identifier is one previously notified in an MQBACF_RESPONSE_SET structure.
- An MQCFST structure with the *Parameter* field set to MQCACF_RESPONSE_Q_MGR_NAME, the value being the name of the queue manager from which the set of responses come.

Many responses have additional parameter structures, and these structures are described in the following sections.

You cannot determine in advance how many responses there are in a set other than by getting responses until one with MQCFC_LAST is found. Neither can you determine in advance how many sets of responses there are as any set might include MQBACF_RESPONSE_SET structures to indicate that additional sets are generated.

Extended responses to Inquire commands

Inquire commands normally generate an item response (type MQCFT_XR_ITEM) for each item found that matches the specified search criteria. The item response has a *CompCode* field in the header with a value of MQCC_OK, and a *Reason* field with a value of MQRC_NONE. It also includes other parameter structures describing the item and its requested attributes, as described in Definitions of the Programmable Command Formats.

If an item is in error, the *CompCode* field in the header has a value of MQCC_FAILED and the *Reason* field identifies the particular error. Additional parameter structures are included to identify the item.

Certain Inquire commands might return general (not name-specific) message responses in addition to the item responses. These responses are informational, or error, responses of the type MQCFT_XR_MSG.

If the Inquire command succeeds, there might, optionally, be a summary response (type MQCFT_XR_SUMMARY), with a *CompCode* value of MQCC_OK, and a *Reason* field value of MQRC_NONE.

If the Inquire command fails, item responses might be returned, and there might optionally be a summary response (type MQCFT_XR_SUMMARY), with a *CompCode* value of MQCC_FAILED, and a *Reason* field value of MQRCCF_COMMAND_FAILED.

Extended responses to commands other than Inquire

Successful commands generate message responses in which the *CompCode* field in the header has a value of MQCC_OK, and the *Reason* field has a value of MQRC_NONE. There is always at least one message; it might be informational (MQCFT_XR_MSG) or a summary (MQCFT_XR_SUMMARY). There might optionally be additional informational (type MQCFT_XR_MSG) messages. Each informational message might include a number of additional parameter structures with information about the command; see the individual command descriptions for the structures that can occur.

Commands that fail generate error message responses (type MQCFT_XR_MSG), in which the *CompCode* field in the header has a value of MQCC_FAILED and the *Reason* field identifies the particular error. Each message might include a number of additional parameter structures with information about the error: see the individual error descriptions for the structures that can occur. Informational message responses might be generated. There might, optionally, be a summary response (MQCFT_XR_SUMMARY), with a *CompCode* value of MQCC_FAILED, and a *Reason* field value of MQRCCF_COMMAND_FAILED.

Extended responses to commands using CommandScope

If a command uses the **CommandScope** parameter, or causes a command using the **CommandScope** parameter to be generated, there is an initial response set from the queue manager where the command was received. Then a separate set, or sets, of responses is generated for each queue manager to which the command is directed (as if multiple individual commands were issued). Finally, there is a response set from the receiving queue manager which includes an overall summary response (type MQCFT_XR_SUMMARY). The MQCACF_RESPONSE_Q_MGR_NAME parameter structure identifies the queue manager that generates each set.

The initial response set has the following additional parameter structures:

- MQIACF_COMMAND_INFO (MQCFIN). Possible values in this structure are MQCMDI_CMDSCOPE_ACCEPTED or MQCMDI_CMDSCOPE_GENERATED.
- MQIACF_CMDSCOPE_Q_MGR_COUNT (MQCFIN). This structure indicates the number of queue managers to which the command is sent.

Comprobación de autorización para mandatos PCF en IBM MQ

Cuando se procesa un mandato PCF, se utiliza el *UserIdentifier* del descriptor de mensaje en el mensaje de mandato para las comprobaciones de autorización sobre objetos necesarias de IBM MQ. La comprobación de autorización se implementa de forma diferente en cada plataforma como se describe en este tema.

Las comprobaciones se realizan en el sistema en el que se está procesando el mandato; por lo tanto, este ID de usuario debe existir en el sistema de destino y tener las autorizaciones necesarias para procesar el mandato. Si el mensaje proviene de un sistema remoto, una forma de alcanzar el ID existente en el sistema de destino es tener un ID de usuario coincidente en los sistemas local y remoto.

Nota: z/OS Para obtener más información sobre la comprobación de autorización en z/OS, consulte Tarea 1: Identificar los parámetros del sistema z/OS.

IBM MQ for IBM i

IBM i

Para procesar cualquier mandato PCF, el ID de usuario debe tener autorización *dsp* para el objeto de IBM MQ en el sistema de destino.

Además, se realizan comprobaciones de autorización de objetos de IBM MQ para determinados mandatos PCF, como se muestra en la Tabla 2 en la página 35.

En la mayoría de los casos, estas comprobaciones son las mismas comprobaciones que las comprobaciones realizadas por los mandatos CL de IBM MQ equivalentes emitidos en un sistema local. Consulte Configuración de la seguridad en IBM i para obtener más información sobre la correlación de autorizaciones de IBM MQ con autorizaciones del sistema IBM i, y los requisitos de autorización para los mandatos CL de IBM MQ. En Seguridad a nivel de enlace mediante una salida de seguridad se ofrece información detallada sobre la seguridad.

Para procesar alguno de los mandatos siguientes, el ID de usuario debe ser miembro del perfil de grupos QMQMADM:

- Sondear canal
- Cambiar canal
- Copiar canal
- Crear canal
- Suprimir canal
- Restablecer canal
- Resolver canal
- Iniciar canal
- Detener canal
- Iniciar iniciador de canal
- Iniciar escucha de canal

IBM MQ for UNIX, Linux, and Windows

ALW

Para procesar cualquier mandato PCF, el ID de usuario debe tener autorización *dsp* para el objeto de gestor de colas en el sistema de destino. Además, se realizan comprobaciones de autorización de objetos de IBM MQ para determinados mandatos PCF, como se muestra en la Tabla 2 en la página 35.

Para procesar cualquiera de los mandatos siguientes, el ID de usuario debe pertenecer al grupo *mqm*.

Nota: Para Windows **únicamente**, el ID de usuario puede pertenecer al grupo *Administradores* o al grupo *mqm*.

- Cambiar canal
- Copiar canal
- Crear canal
- Suprimir canal
- Sondear canal
- Restablecer canal
- Iniciar canal
- Detener canal
- Iniciar iniciador de canal
- Iniciar escucha de canal
- Resolver canal
- Restablecer clúster
- Renovar clúster
- Suspender gestor de colas
- Reanudar gestor de colas

Autorizaciones de objetos de IBM MQ para Multiplatforms

Multi

Tabla 2. Autorizaciones de objetos

Mandato	**Autorización de objetos de IBM MQ**	**Autorización de clases (para tipo de objeto)**
Modificar información de autorización	dsp y chg	n/d
Cambiar canal	dsp y chg	n/d
Modificar escucha de canal	dsp y chg	n/d
Modificar canal de conexión de cliente	dsp y chg	n/d
Modificar lista de nombres	dsp y chg	n/d
Modificar proceso	dsp y chg	n/d
Modificar cola	dsp y chg	n/d
Cambiar gestor de colas	chg *véase la nota 3 y la nota 5*	n/d
Cambiar servicio	dsp y chg	n/d
Borrar cola	clr	n/d
Copiar información de autenticación	dsp	crt
Copiar información de autenticación (reemplazar) *ver nota 1*	*de:* dsp *a:* chg	crt
Copiar canal	dsp	crt
Copiar canal (reemplazar) *ver nota 1*	*de:* dsp *a:* chg	crt
Copiar escucha de canal	dsp	crt

Tabla 2. Autorizaciones de objetos (continuación)		
Mandato	**Autorización de objetos de IBM MQ**	**Autorización de clases (para tipo de objeto)**
Copiar escucha de canal (reemplazar) *ver nota 1*	*de:* dsp *a:* chg	crt
Copiar canal de conexión de cliente	dsp	crt
Copiar canal de conexión de cliente (reemplazar) *ver nota 1*	*de:* dsp *a:* chg	crt
Copiar lista de nombres	dsp	crt
Copiar lista de nombres (reemplazar) *ver nota 1*	*de:* dsp *a:* dsp y chg	crt
Copiar proceso	dsp	crt
Copiar proceso (reemplazar) *ver nota 1*	*de:* dsp *a:* chg	crt
Copiar cola	dsp	crt
Copiar cola (reemplazar) *ver nota 1*	*de:* dsp *a:* dsp y chg	crt
Crear información de autenticación	*(información de autenticación predeterminada del sistema)* dsp	crt
Crear información de autenticación (reemplazar) *ver nota 1*	*(información de autenticación predeterminada del sistema)* dsp *a:* chg	crt
Crear canal	*(canal predeterminado del sistema)* dsp	crt
Crear canal (reemplazar) *ver nota 1*	*(canal predeterminado del sistema)* dsp *a:* chg	crt
Crear escucha de canal	*(escucha predeterminado del sistema)* dsp	crt
Crear escucha de canal (reemplazar) *ver nota 1*	*(escucha predeterminado del sistema)* dsp *a:* chg	crt
Crear canal de conexión de cliente	*(canal predeterminado del sistema)* dsp	crt
Crear canal de conexión de cliente (reemplazar) *ver nota 1*	*(canal predeterminado del sistema)* dsp *a:* chg	crt
Crear lista de nombres	*(lista de nombres predeterminada del sistema)* dsp	crt
Crear lista de nombres (reemplazar) *ver nota 1*	*(lista de nombres predeterminada del sistema)* dsp *a:* dsp y chg	crt
Crear proceso	*(proceso predeterminado del sistema)* dsp	crt
Crear proceso (reemplazar) *ver nota 1*	*(proceso predeterminado del sistema)* dsp *a:* chg	crt
Crear cola	*(cola predeterminada del sistema)* dsp	crt
Crear cola (reemplazar) *ver nota 1*	*(cola predeterminada del sistema)* dsp *a:* dsp y chg	crt

Tabla 2. Autorizaciones de objetos (continuación)		
Mandato	**Autorización de objetos de IBM MQ**	**Autorización de clases (para tipo de objeto)**
Crear servicio	*(cola predeterminada del sistema)* dsp	crt
Crear servicio (reemplazar) *ver nota 1*	*(cola predeterminada del sistema)* dsp *a:* chg	crt
Suprimir información de autenticación	dsp y dlt	n/d
Suprimir registro de autorización	*(objeto de gestor de colas)* chg *ver nota 4*	*ver nota 4*
Suprimir canal	dsp y dlt	n/d
Suprimir escucha de canal	dsp y dlt	n/d
Suprimir canal de conexión de cliente	dsp y dlt	n/d
Suprimir lista de nombres	dsp y dlt	n/d
Suprimir proceso	dsp y dlt	n/d
Suprimir cola	dsp y dlt	n/d
Suprimir servicio	dsp y dlt	n/d
Consultar información de autenticación	dsp	n/d
Consultar registros de autorización	*ver nota 4*	*ver nota 4*
Consultar canal	dsp	n/d
Consultar escucha de canal	dsp	n/d
Consultar estado de canal (para **ChannelType** MQCHT_CLSSDR)	inq	n/d
Consultar canal de conexión de cliente	dsp	n/d
Consultar lista de nombres	dsp	n/d
Consultar proceso	dsp	n/d
Consultar cola	dsp	n/d
Consultar gestor de colas	*ver nota 3*	n/d
Consultar estado de la cola	dsp	n/d
Consultar servicio	dsp	n/d
Sondear canal	ctrl	n/d
Sondear gestor de colas	*ver nota 3*	n/d
Renovar gestor de colas	(objeto de gestor de colas) chg	n/d
Renovar seguridad (para **SecurityType** MQSECTYPE_SSL)	(objeto de gestor de colas) chg	n/d
Restablecer canal	ctrlx	n/d

Tabla 2. Autorizaciones de objetos (continuación)		
Mandato	**Autorización de objetos de IBM MQ**	**Autorización de clases (para tipo de objeto)**
Restablecer gestor de colas	(objeto de gestor de colas) chg	n/d
Restablecer estadísticas de la cola	dsp y chg	n/d
Resolver canal	ctrlx	n/d
Establecer registro de autorización	*(objeto de gestor de colas)* chg *ver nota 4*	*ver nota 4*
Iniciar canal	ctrl	n/d
Detener canal	ctrl	n/d
Detener conexión	(objeto de gestor de colas) chg	n/d
Iniciar escucha	ctrl	n/d
Detener escucha	ctrl	n/d
Iniciar servicio	ctrl	n/d
Detener servicio	ctrl	n/d
Escape	*ver nota 2*	*ver nota 2*

Notas:

1. Este mandato se aplica si el objeto que debe reemplazarse ya existe, de lo contrario la comprobación de autorización es como para Crear o Copiar sin reemplazar.
2. La autorización necesaria se determina mediante el mandato MQSC definido por el texto de escape y es equivalente a uno de los mandatos anteriores.
3. Para procesar cualquier mandato PCF, el ID de usuario debe tener autorización dsp para el objeto de gestor de colas en el sistema de destino.
4. Este mandato PCF está autorizado a menos que el servidor de mandatos se haya iniciado con el parámetro -a. De forma predeterminada, el servidor de mandatos se inicia al iniciarse el gestor de colas, sin el parámetro -a. Para obtener más información, consulte Referencia de formatos de mandatos programables.
5. Otorgar a un ID de usuario autorización chg para un gestor de colas concede la capacidad de establecer registros de autorización para todos los grupos y usuarios. Esta autorización no debe otorgarse a usuarios ni aplicaciones ordinarios.

IBM MQ también proporciona algunos puntos de salida de seguridad de canal para que pueda suministrar sus propios programas de salida de usuario para la comprobación de seguridad. Para obtener más información, consulte Visualización de un canal.

Multi Utilizar la MQAI para simplificar el uso de los PCF

La interfaz de administración de IBM MQ (MQAI) es una interfaz de programación para IBM MQ que está disponible en AIX, IBM i, Linux, y Windows. Realiza tareas de administración en un gestor de colas de IBM MQ mediante paquetes de datos para manejar las propiedades (o parámetros) de objetos de forma que es más fácil que utilizando los formatos de mandato programable (PCF).

La MQAI realiza tareas de administración en un gestor de colas mediante el uso de *paquetes de datos*. Los paquetes de datos permiten manejar las propiedades (o parámetros) de los objetos de forma más sencilla que utilizando mandatos PCF.

Las ventajas de utilizar la MQAI son las siguientes:

Simplificar el uso de los mensajes PCF

La MQAI es una forma más fácil de administrar IBM MQ. Si utiliza la MQAI, no tiene que escribir sus propios mensajes PCF. Esto evita los problemas asociados a las estructuras de datos complejas.

Para pasar parámetros en programas escritos utilizando llamadas MQI, el mensaje PCF debe contener el mandato y detalles de los datos de tipo entero o de serie de caracteres. Para crear esta configuración manualmente, tiene que añadir varias sentencias al programa para cada estructura y asignar espacio de memoria. Esta tarea puede resultar larga y laboriosa.

Los programas escritos utilizando la MQAI pasan parámetros en el paquete de datos adecuado y tan solo se requiere una sentencia para cada estructura. La utilización de paquetes de datos de la MQAI elimina la necesidad de manejar matrices y asignar almacenamiento y proporciona cierto grado de aislamiento de los detalles del PCF.

Manejar las condiciones de error con más facilidad

Es difícil obtener códigos de retorno de los mandatos PCF. La MQAI permite al programa manejar más fácilmente las condiciones de error.

Intercambiar datos entre aplicaciones

Los datos de aplicación se envían en formato PCF y los empaqueta y desempaqueta la MQAI. Si los datos del mensaje constan de enteros y series de caracteres, puede utilizar la MQAI para sacar partido de la conversión de datos incorporada de IBM MQ para datos PCF. Esto evita la necesidad de grabar salidas de conversión de datos.

Después de haber creado y llenado los paquetes de datos, puede enviar un mensaje de mandato de administración al servidor de mandatos de un gestor de colas, utilizando la llamada `mqExecute`. Esta llamada espera los mensajes de respuesta. La llamada `mqExecute` maneja el intercambio con el servidor de mandatos y devuelve respuestas en un *paquete de respuestas*.

Ejemplos de cómo utilizar la MQAI

Los programas de ejemplo siguientes muestran el uso de la interfaz de administración de IBM MQ para realizar las diversas tareas:

- `amqsaicq.c`: crear una cola local.
- `amqsaiem.c`: mostrar sucesos en la pantalla utilizando un supervisor de sucesos simple.
- `amqsailq.c`: imprimir una lista de todas las colas locales y sus profundidades actuales.
- `amqsaicl.c`: imprimir una lista de todos los canales y sus tipos.

Creación de una aplicación MQAI

Para crear la aplicación utilizando la MQAI, debe enlazar con las mismas bibliotecas, tal como lo haría para IBM MQ. Para obtener información sobre cómo crear las aplicaciones de IBM MQ, consulte Creación de una aplicación de procedimientos.

Consejos y sugerencias para configurar IBM MQ utilizando MQAI

La MQAI utiliza los mensajes PCF para enviar mandatos de administración al servidor de mandatos, en lugar de tratar directamente con el propio servidor de mandatos. Encontrará consejos para configurar IBM MQ utilizando la MQAI en “Consejos y sugerencias para utilizar MQAI para configurar IBM MQ” en la página 39.

Referencia relacionada

Referencia de la interfaz de administración de IBM MQ

Multi Consejos y sugerencias para utilizar MQAI para configurar IBM MQ

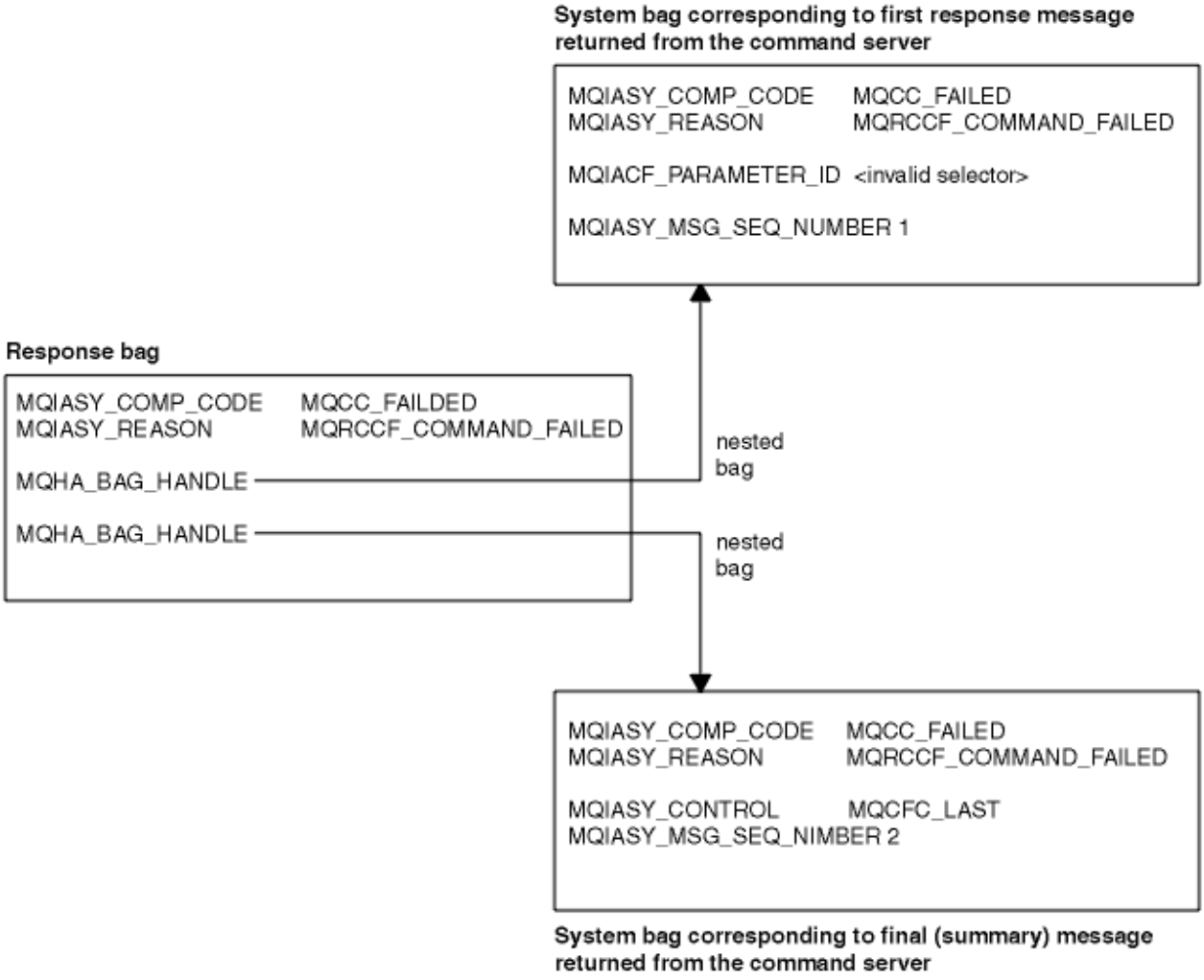
La Interfaz de administración de IBM MQ (MQAI) utiliza los mensajes PCF para enviar mandatos de administración al servidor de mandatos, en lugar de tratar directamente con el propio servidor de

mandatos. A continuación se proporcionan algunas sugerencias para configurar IBM MQ utilizando la MQAI.

- Las series de caracteres en IBM MQ se rellenan con blancos hasta una longitud fija. Utilizando C, las series terminadas en nulo normalmente pueden proporcionarse como parámetros de entrada a las interfaces de programación de IBM MQ.
- Para borrar el valor de un atributo de serie, establézcalo en un único carácter en blanco en lugar de en una serie vacía.
- Piense con antelación los atributos que desea cambiar y realice consultas únicamente sobre los mismos.
- Determinados atributos no pueden cambiarse, por ejemplo un nombre de cola o un tipo de canal. Asegúrese de que intenta cambiar únicamente los atributos que se pueden modificar. Consulte la lista de parámetros necesarios y opcionales para el objeto de cambio PCF específico. Consulte Definiciones de los formatos de mandatos programables.
- Si una llamada MQAI falla, algunos detalles del error se devuelven al paquete de respuesta. Se pueden encontrar más detalles en un paquete anidado al que puede acceder el selector MQHA_BAG_HANDLE. Por ejemplo, si una llamada mqExecute falla con un código de razón MQRCCF_COMMAND_FAILED, esta información se devuelve en el paquete de respuesta. Una causa posible de este código de razón es que un selector especificado no es válido para el tipo de mensaje de mandato y este detalle de la información se encuentra en un paquete anidado al que puede acceder un manejador de paquete.

  Para obtener más información sobre MQExecute, consulte “Envío de mandatos de administración al servidor de mandatos qm utilizando la llamada mqExecute” en la página 74

  El siguiente diagrama muestra este escenario:



Multi Temas de la interfaz de administración de IBM MQ avanzados

Información sobre la indexación, conversión de datos y uso del descriptor de mensaje

Indexación
Los índices se utilizan cuando se sustituyen o eliminan los elementos de datos existentes de un paquete para conservar el orden de inserción.

Conversión de datos
Las series contenidas en un paquete de datos MQAI (interfaz de administración de IBM MQ) pueden estar en distintos juegos de caracteres codificados y pueden convertirse utilizando la llamada mqSetInteger.

Uso del descriptor de mensaje
La interfaz de administración de IBM MQ genera un descriptor de mensaje que se establece en un valor inicial cuando se crea el paquete de datos.

Multi *Indexación en la interfaz de administración de IBM MQ*

Los índices se utilizan cuando se sustituyen o eliminan los elementos de datos existentes de un paquete. Existen tres tipos de indexación, que permite recuperar fácilmente los elementos de datos.

Cada selector y valor dentro de un elemento de datos de un paquete tiene tres números de asociados:

- El índice relativo a otros elementos que tienen el mismo selector.
- El índice relativo a la categoría de selector (usuario o sistema) al que pertenece el elemento.
- El índice relativo a todos los elementos de datos del paquete (usuario y sistema).

Esto permite la indexación según los selectores de usuario, los selectores del sistema, o ambos como se muestra en la Figura 3 en la página 42.

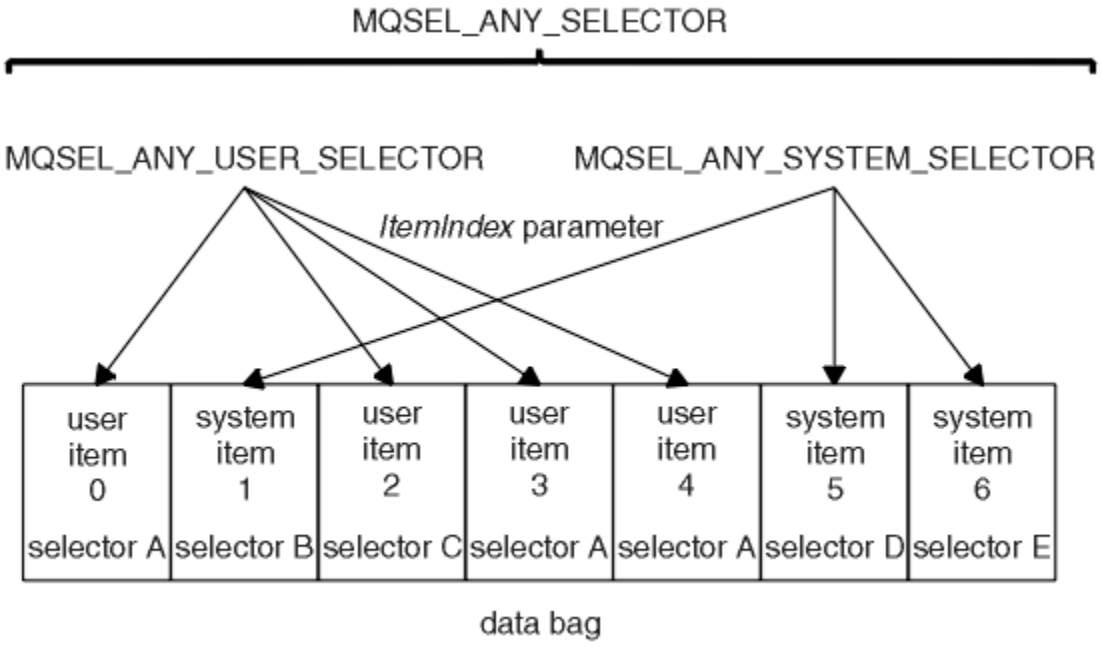


Figura 3. Indexación

En la Figura 3 en la página 42, se puede hacer referencia al elemento de usuario 3 (selector A) mediante los pares de índice siguientes:

- selector A (ItemIndex 1)
- MQSEL_ANY_USER_SELECTOR (ItemIndex 2)
- MQSEL_ANY_SELECTOR (itemIndex 3)

El índice tiene base cero como una matriz en c; si hay ‘n' apariciones, el índice oscila entre cero y ‘n-1', sin espacios.

Los índices se utilizan cuando se sustituyen o eliminan los elementos de datos existentes de un paquete. Cuando se utilizan de esta manera, el orden de inserción se conserva pero los índices de otros elementos de datos pueden verse afectados. Para obtener ejemplos de esto, consulte “Cambiar información dentro de un paquete” en la página 71 y “Supresión de elementos de datos” en la página 73.

Los tres tipos de indexación permiten una recuperación fácil de elementos de datos. Por ejemplo, si hay tres instancias de un selector concreto en un paquete, la llamada mqCountItems puede contar el número de instancias de ese selector y las llamadas mqInquire* pueden especificar tanto el selector como el índice para consultar solamente los valores. Esto es útil para los atributos que pueden tener una lista de valores como algunas de las salidas en los canales.

Multi *Proceso de conversión de datos en la interfaz de administración de IBM MQ*

Las series contenidas en un paquete de datos MQAI pueden estar en diversos juegos de caracteres codificados. Estas series se pueden convertir utilizando la llamada mqSetInteger.

Al igual que los mensajes PCF, las series contenidas en un paquete de datos MQAI pueden estar en diversos juegos de caracteres codificados. Normalmente, todas las series en un mensaje PCF están en el mismo juego de caracteres codificado; es decir, el mismo conjunto que el gestor de colas.

Cada elemento de serie del paquete de datos contiene dos valores; la propia serie y el CCSID.
La serie que se añade al paquete se obtiene a partir del parámetro **Buffer** de la llamada mqAddString o mqSetString. El CCSID se obtiene del elemento del sistema que contiene un selector de MQIASY_CODED_CHAR_SET_ID. Esto se conoce como el *CCSID de paquete* y se puede cambiar utilizando la llamada mqSetInteger.

Al consultar el valor de una serie incluida en un paquete de datos, el CCSID es un parámetro de salida de la llamada.

Tabla 3 en la página 43 muestra las normas aplicadas al convertir los paquetes de datos en mensajes y viceversa:

Tabla 3. Proceso de CCSID

Llamada de MQAI	**CCSID**	**Entrada para llamada**	**Salida para llamada**
mqBagToBuffer	CCSID incorrecto (1)	Se ignora	Sin modificar
mqBagToBuffer	CCSID de series del paquete	Se utiliza	Sin modificar
mqBagToBuffer	CCSID de series del almacenamiento intermedio	No aplicable	Se copia de los CCSID de series del paquete
mqBufferToBag	CCSID incorrecto (1)	Se ignora	Sin modificar
mqBufferToBag	CCSID de series del almacenamiento intermedio	Se utiliza	Sin modificar
mqBufferToBagmqBufferToBag	CCSID de series del paquete	No aplicable	Se copia de los CCSID de serie del almacenamiento intermedio
mqPutBag	CCSID de MQMD	Se utiliza	Sin modificar (2)
mqPutBag	CCSID incorrecto (1)	Se ignora	Sin modificar
mqPutBag	CCSID de series del paquete	Se utiliza	Sin modificar
mqPutBag	CCSID de series del mensaje enviado	No aplicable	Se copia de los CCSID de series del paquete
mqGetBolsa	CCSID de MQMD	Se utiliza para la conversión de datos de mensaje	Establecido en CCSID de datos devueltos (3)
mqGetBolsa	CCSID incorrecto (1)	Se ignora	Sin modificar
mqGetBolsa	CCSID de series de mensaje	Se utiliza	Sin modificar
mqGetBolsa	CCSID de series del paquete	No aplicable	Se copia de los CCSID de series del mensaje
mqExecute	CCSID de paquete de solicitud	Utilizado para MQMD del mensaje de solicitud (4)	Sin modificar
mqExecute	CCSID de paquete de respuesta	Se utiliza para la conversión de datos del mensaje de respuesta (4)	Establecido en CCSID de datos devueltos (3)
mqExecute	CCSID de series de paquete de solicitud	Se utiliza para el mensaje de solicitud	Sin modificar

Tabla 3. Proceso de CCSID (continuación)			
Llamada de MQAI	**CCSID**	**Entrada para llamada**	**Salida para llamada**
mqExecute	CCSID de series en paquete de respuesta	No aplicable	Se copia de los CCSID de serie del mensaje de respuesta

Notas:

1. El CCSID de paquete es el elemento del sistema con el selector MQIASY_CODED_CHAR_SET_ID.
2. MQCCSI_Q_MGR se cambia por el CCSID del gestor de colas real.
3. Si se solicita la conversión de datos, el CCSID de datos devuelto es el mismo que el valor de salida. Si no se solicita la conversión de datos, el CCSID de datos devueltos es el mismo que el valor de mensaje. Tenga en cuenta que se devuelve ningún mensaje si se solicita la conversión de datos pero falla.
4. Si el CCSID es MQCCSI_DEFAULT, se utilizará el CCSID del gestor de colas.

Conceptos relacionados

"Conversión de datos entre juegos de caracteres codificados" en la página 208
El gestor de colas puede convertir datos de mensaje en formatos definidos de IBM MQ (también conocidos como formatos incorporados) de un juego de caracteres codificado a otro, siempre que ambos juegos de caracteres estén relacionados con un único idioma o un grupo de idiomas similares.

"El archivo ccsid_part2.tbl" en la página 210
El archivo `ccsid_part2.tbl` se utiliza para proporcionar información CCSID adicional. El archivo `ccsid_part2.tbl` sustituye el archivo `ccsid.tbl` que se ha utilizado antes de IBM MQ 9.0.

Multi

Uso del descriptor de mensaje en la interfaz de administración de IBM MQ

El descriptor de mensaje que se la interfaz de administración de IBM MQ genera se establece en un valor inicial cuando se crea el paquete de datos.

El tipo de mandato PCF se obtiene del elemento del sistema con el selector MQIASY_TYPE. Al crear el paquete de datos, el valor inicial de este elemento se establece en función del tipo de paquete que se crea:

Tabla 4. Tipo de mandato PCF	
Tipo de paquete	**Valor inicial del elemento MQIASY_TYPE**
MQCBO_ADMIN_BAG	MQCFT_COMMAND
MQCBO_COMMAND_BAG	MQCFT_COMMAND
MQCBO_*	MQCFT_USER

Cuando la MQAI genera un descriptor de mensaje, los valores utilizados en los parámetros **Format** y **MsgType** dependen del valor del elemento del sistema con el selector MQIASY_TYPE tal como se muestra en la Tabla 4 en la página 44.

Tabla 5. Formato y parámetros MsgType del MQMD		
Tipo de mandato PCF	**Formato**	**MsgType**
MQCFT_COMMAND	MQFMT_ADMIN	MQMT_REQUEST
MQCFT_REPORT	MQFMT_ADMIN	MQMT_REPORT
MQCFT_RESPONSE	MQFMT_ADMIN	MQMT_REPLY
MQCFT_TRACE_ROUTE	MQFMT_ADMIN	MQMT_DATAGRAM
MQCFT_EVENT	MQFMT_EVENT	MQMT_DATAGRAM

Tabla 5. Formato y parámetros MsgType del MQMD (continuación)		
Tipo de mandato PCF	**Formato**	**MsgType**
MQCFT_*	MQFMT_PCF	MQMT_DATAGRAM

Tabla 5 en la página 44 muestra que si crea un paquete de administración o un paquete de mandatos, el *Format* del descriptor de mensaje es MQFMT_ADMIN y el *MsgType* es MQMT_REQUEST. Esto resulta adecuado para un mensaje de petición PCF que se envía al servidor de mandatos cuando se espera una respuesta.

Otros parámetros del descriptor de mensaje adoptan los valores que se muestran en la Tabla 6 en la página 45.

Tabla 6. Valores de descriptor de mensaje	
Parámetro	**Valor**
StrucId	MQMD_STRUC_ID
Version	MQMD_VERSION_1
Report	MQRO_NONE
MsgType	Consulte Tabla 5 en la página 44
Expiry	30 segundos (vea "1" en la página 45)
Feedback	MQFB_NONE
Encoding	MQENC_NATIVE
CodedCharSetId	depende del CCSID del paquete (nota "2" en la página 45)
Format	Consulte Tabla 5 en la página 44
Priority	MQPRI_PRIORITY_AS_Q_DEF
Persistence	MQPER_NOT_PERSISTENT
MsgId	MQMI_NONE
CorrelId	MQCI_NONE
BackoutCount	0
ReplyToQ	vea la nota "3" en la página 45
ReplyToQMgr	en blanco

Notas:

1. Este valor puede alterarse en la llamada mqExecute utilizando el parámetro **OptionsBag**. Para obtener información sobre esto, consulte mqExecute.
2. Consulte "Proceso de conversión de datos en la interfaz de administración de IBM MQ" en la página 42.
3. El nombre de la cola de respuesta especificada por el usuario o la cola dinámica temporal generada por la MQAI para los mensajes del tipo MQMT_REQUEST. De lo contrario, espacio en blanco.

Multi Programa C de ejemplo para crear una cola local (amqsaicq.c)

El programa C de ejemplo `amqsaicq.c` crea una cola local utilizando MQAI.

```
/******************************************************************************/
```

```
/*                                                                  */
/* Program name: AMQSAICQ.C                                         */
/*                                                                  */
/* Description:  Sample C program to create a local queue using the */
/*               IBM MQ Administration Interface (MQAI).            */
/*                                                                  */
/* Statement:    Licensed Materials - Property of IBM               */
/*                                                                  */
/*               84H2000, 5765-B73                                  */
/*               84H2001, 5639-B42                                  */
/*               84H2002, 5765-B74                                  */
/*               84H2003, 5765-B75                                  */
/*               84H2004, 5639-B43                                  */
/*                                                                  */
/*               (C) Copyright IBM Corp. 1999, 2024                 */
/*                                                                  */
/********************************************************************/
/*                                                                  */
/* Function:                                                        */
/*    AMQSAICQ is a sample C program that creates a local queue and is an */
/*    example of the use of the mqExecute call.                     */
/*                                                                  */
/*     - The name of the queue to be created is a parameter to the program. */
/*                                                                  */
/*     - A PCF command is built by placing items into an MQAI bag.  */
/*       These are:-                                                */
/*            - The name of the queue                               */
/*            - The type of queue required, which, in this case, is local. */
/*                                                                  */
/*     - The mqExecute call is executed with the command MQCMD_CREATE_Q. */
/*       The call generates the correct PCF structure.              */
/*       The call receives the reply from the command server and formats into */
/*       the response bag.                                          */
/*                                                                  */
/*     - The completion code from the mqExecute call is checked and if there */
/*       is a failure from the command server then the code returned by the */
/*       command server is retrieved from the system bag that is    */
/*       embedded in the response bag to the mqExecute call.        */
/*                                                                  */
/* Note: The command server must be running.                        */
/*                                                                  */
/*                                                                  */

/********************************************************************/
/*                                                                  */
/* AMQSAICQ has 2 parameters - the name of the local queue to be created */
/*                          - the queue manager name (optional)     */
/*                                                                  */
/********************************************************************/
/********************************************************************/
/* Includes                                                         */
/********************************************************************/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>

#include <cmqc.h>                       /* MQI                      */
#include <cmqcfc.h>                     /* PCF                      */
#include <cmqbc.h>                      /* MQAI                     */

void CheckCallResult(MQCHAR *, MQLONG , MQLONG );
void CreateLocalQueue(MQHCONN, MQCHAR *);

int main(int argc, char *argv[])
{
   MQHCONN hConn;                         /* handle to IBM MQ connection   */
   MQCHAR QMName[MQ_Q_MGR_NAME_LENGTH+1]=""; /* default QMgr name          */
   MQLONG connReason;                     /* MQCONN reason code            */
   MQLONG compCode;                       /* completion code               */
   MQLONG reason;                         /* reason code                   */

   /*****************************************************************/
   /* First check the required parameters                           */
   /*****************************************************************/
   printf("Sample Program to Create a Local Queue\n");
   if (argc < 2)
   {
     printf("Required parameter missing - local queue name\n");
     exit(99);
   }
```

```
   /*****************************************************************************/
   /* Connect to the queue manager                                              */
   /*****************************************************************************/
   if (argc > 2)
      strncpy(QMName, argv[2], (size_t)MQ_Q_MGR_NAME_LENGTH);
      MQCONN(QMName, &hConn, &compCode, &connReason);

/*****************************************************************************/
/* Report reason and stop if connection failed                               */
/*****************************************************************************/
   if (compCode == MQCC_FAILED)
   {
      CheckCallResult("MQCONN", compCode, connReason);
      exit( (int)connReason);
   }


/*****************************************************************************/
/* Call the routine to create a local queue, passing the handle to the       */
/* queue manager and also passing the name of the queue to be created.       */
/*****************************************************************************/
   CreateLocalQueue(hConn, argv[1]);

   /*****************************************************************************/
   /* Disconnect from the queue manager if not already connected                */
   /*****************************************************************************/
   if (connReason != MQRC_ALREADY_CONNECTED)
   {
      MQDISC(&hConn, &compCode, &reason);
      CheckCallResult("MQDISC", compCode, reason);
   }
   return 0;

}
/*****************************************************************************/
/*                                                                           */
/* Function:    CreateLocalQueue                                             */
/* Description: Create a local queue by sending a PCF command to the command */
/*              server.                                                      */
/*                                                                           */
/*****************************************************************************/
/*                                                                           */
/* Input Parameters:  Handle to the queue manager                            */
/*                    Name of the queue to be created                        */
/*                                                                           */
/* Output Parameters: None                                                   */
/*                                                                           */
/* Logic: The mqExecute call is executed with the command MQCMD_CREATE_Q.    */
/*        The call generates the correct PCF structure.                      */
/*        The default options to the call are used so that the command is sent*/
/*        to the SYSTEM.ADMIN.COMMAND.QUEUE.                                 */
/*        The reply from the command server is placed on a temporary dynamic */
/*        queue.                                                             */
/*        The reply is read from the temporary queue and formatted into the  */
/*        response bag.                                                      */
/*                                                                           */
/*        The completion code from the mqExecute call is checked and if there */
/*        is a failure from the command server then the code returned by the */
/*        command server is retrieved from the system bag that is            */
/*        embedded in the response bag to the mqExecute call.                */
/*                                                                           */
/*****************************************************************************/
void CreateLocalQueue(MQHCONN hConn, MQCHAR *qName)
{
   MQLONG reason;                         /* reason code                     */
   MQLONG compCode;                       /* completion code                 */
   MQHBAG commandBag = MQHB_UNUSABLE_HBAG; /* command bag for mqExecute      */
   MQHBAG responseBag = MQHB_UNUSABLE_HBAG;/* response bag for mqExecute     */
   MQHBAG resultBag;                      /* result bag from mqExecute       */
   MQLONG mqExecuteCC;                    /* mqExecute completion code       */
   MQLONG mqExecuteRC;                    /* mqExecute reason code           */

   printf("\nCreating Local Queue %s\n\n", qName);


   /*****************************************************************************/
   /* Create a command Bag for the mqExecute call. Exit the function if the   */
   /* create fails.                                                             */
   /*****************************************************************************/
   mqCreateBag(MQCBO_ADMIN_BAG, &commandBag, &compCode, &reason);
   CheckCallResult("Create the command bag", compCode, reason);
   if (compCode !=MQCC_OK)
```

```
   return;

/*****************************************************************************/
/* Create a response Bag for the mqExecute call, exit the function if the  */
/* create fails.                                                             */
/*****************************************************************************/
mqCreateBag(MQCBO_ADMIN_BAG, &responseBag, &compCode, &reason);
CheckCallResult("Create the response bag", compCode, reason);
if (compCode !=MQCC_OK)
   return;

/*****************************************************************************/
/* Put the name of the queue to be created into the command bag. This will */
/* be used by the mqExecute call.                                            */
/*****************************************************************************/
mqAddString(commandBag, MQCA_Q_NAME, MQBL_NULL_TERMINATED, qName, &compCode,
            &reason);
CheckCallResult("Add q name to command bag", compCode, reason);

/*****************************************************************************/
/* Put queue type of local into the command bag. This will be used by the  */
/* mqExecute call.                                                           */
/*****************************************************************************/
mqAddInteger(commandBag, MQIA_Q_TYPE, MQQT_LOCAL, &compCode, &reason);
CheckCallResult("Add q type to command bag", compCode, reason);

/*****************************************************************************/
/* Send the command to create the required local queue.                    */
/* The mqExecute call will create the PCF structure required, send it to   */
/* the command server and receive the reply from the command server into   */
/* the response bag.                                                         */
/*****************************************************************************/
mqExecute(hConn,                    /* IBM MQ connection handle            */
          MQCMD_CREATE_Q,           /* Command to be executed              */
          MQHB_NONE,                /* No options bag                      */
          commandBag,               /* Handle to bag containing commands   */
          responseBag,              /* Handle to bag to receive the response*/
          MQHO_NONE,                /* Put msg on SYSTEM.ADMIN.COMMAND.QUEUE*/
          MQHO_NONE,                /* Create a dynamic q for the response  */
          &compCode,           /* Completion code from the mqExecute   */
          &reason);            /* Reason code from mqExecute call      */

if (reason == MQRC_CMD_SERVER_NOT_AVAILABLE)
{
   printf("Please start the command server: <strmqcsv QMgrName>\n")
   MQDISC(&hConn, &compCode, &reason);
   CheckCallResult("MQDISC", compCode, reason);
   exit(98);
}


/*****************************************************************************/
/* Check the result from mqExecute call and find the error if it failed.   */
/*****************************************************************************/
if ( compCode == MQCC_OK )
   printf("Local queue %s successfully created\n", qName);
else
{
   printf("Creation of local queue %s failed: Completion Code = %d
           qName, compCode, reason);
   if (reason == MQRCCF_COMMAND_FAILED)
   {
      /*************************************************************************/
      /* Get the system bag handle out of the mqExecute response bag.      */
      /* This bag contains the reason from the command server why the      */
      /* command failed.                                                       */
      /*************************************************************************/
      mqInquireBag(responseBag, MQHA_BAG_HANDLE, 0, &resultBag, &compCode,
                   &reason);
      CheckCallResult("Get the result bag handle", compCode, reason);

      /*************************************************************************/
      /* Get the completion code and reason code, returned by the command  */
      /* server, from the embedded error bag.                                  */
      /*************************************************************************/
      mqInquireInteger(resultBag, MQIASY_COMP_CODE, MQIND_NONE, &mqExecuteCC,
                       &compCode, &reason);
      CheckCallResult("Get the completion code from the result bag",
                       compCode, reason);
      mqInquireInteger(resultBag, MQIASY_REASON, MQIND_NONE, &mqExecuteRC,
                       &compCode, &reason);
      CheckCallResult("Get the reason code from the result bag", compCode,
```

```
                          reason);
         printf("Error returned by the command server: Completion code = %d :
                 Reason = %d\n", mqExecuteCC, mqExecuteRC);
      }
   }
   /*****************************************************************************/
   /* Delete the command bag if successfully created.                           */
   /*****************************************************************************/
   if (commandBag != MQHB_UNUSABLE_HBAG)
   {
      mqDeleteBag(&commandBag, &compCode, &reason);
      CheckCallResult("Delete the command bag", compCode, reason);
   }

   /*****************************************************************************/
   /* Delete the response bag if successfully created.                          */
   /*****************************************************************************/
   if (responseBag != MQHB_UNUSABLE_HBAG)
   {
      mqDeleteBag(&responseBag, &compCode, &reason);
      CheckCallResult("Delete the response bag", compCode, reason);
   }
} /* end of CreateLocalQueue */


/*****************************************************************************/
/*                                                                           */
/* Function: CheckCallResult                                                 */
/*                                                                           */
/*****************************************************************************/
/*                                                                           */
/* Input Parameters:  Description of call                                    */
/*                    Completion code                                        */
/*                    Reason code                                            */
/*                                                                           */
/* Output Parameters: None                                                   */
/*                                                                           */
/* Logic: Display the description of the call, the completion code and the   */
/*        reason code if the completion code is not successful               */
/*                                                                           */
/*****************************************************************************/
void  CheckCallResult(char *callText, MQLONG cc, MQLONG rc)
{
   if (cc != MQCC_OK)
         printf("%s failed: Completion Code = %d :
                 Reason = %d\n", callText, cc, rc);

}
```

Multi Programa C de ejemplo para visualizar sucesos utilizando un supervisor de sucesos (amqsaiem.c)

El programa C de ejemplo `amqsaiem.c` muestra un supervisor de sucesos básico utilizando MQAI.

```
*****************************************************************************/
/*                                                                           */
/* Program name: AMQSAIEM.C                                                  */
/*                                                                           */
/* Description:  Sample C program to demonstrate a basic event monitor       */
/*               using the IBM MQ Admin Interface (MQAI).                    */
/* Licensed Materials - Property of IBM                                      */
/*                                                                           */
/* 63H9336                                                                   */
/* (c) Copyright IBM Corp. 1999, 2024. All Rights Reserved.                  */
/*                                                                           */
/* US Government Users Restricted Rights - Use, duplication or               */
/* disclosure restricted by GSA ADP Schedule Contract with                   */
/* IBM Corp.                                                                 */
/*****************************************************************************/
/*                                                                           */
/* Function:                                                                 */
/*    AMQSAIEM is a sample C program that demonstrates how to write a simple  */
/*    event monitor using the mqGetBag call and other MQAI calls.            */
/*                                                                           */
/*    The name of the event queue to be monitored is passed as a parameter    */
/*    to the program. This would usually be one of the system event queues:-  */
/*            SYSTEM.ADMIN.QMGR.EVENT        Queue Manager events            */
```

```
/*            SYSTEM.ADMIN.PERFM.EVENT      Performance events              */
/*            SYSTEM.ADMIN.CHANNEL.EVENT    Channel events                  */
/*            SYSTEM.ADMIN.LOGGER.EVENT     Logger events                   */
/*                                                                          */
/*    To monitor the queue manager event queue or the performance event queue,*/
/*    the attributes of the queue manager need to be changed to enable      */
/*    these events. For more information about this, see Part 1 of the      */
/*    Programmable System Management book. The queue manager attributes can */
/*    be changed using either MQSC commands or the MQAI interface.          */
/*    Channel events are enabled by default.                                */
/*                                                                          */
/* Program logic                                                            */
/*    Connect to the Queue Manager.                                         */
/*    Open the requested event queue with a wait interval of 30 seconds.    */
/*    Wait for a message, and when it arrives get the message from the queue*/
/*    and format it into an MQAI bag using the mqGetBag call.               */
/*    There are many types of event messages and it is beyond the scope of  */
/*    this sample to program for all event messages. Instead the program    */
/*    prints out the contents of the formatted bag.                         */
/*    Loop around to wait for another message until either there is an error*/
/*    or the wait interval of 30 seconds is reached.                        */
/*                                                                          */
/****************************************************************************/
/*                                                                          */
/* AMQSAIEM has 2 parameters - the name of the event queue to be monitored  */
/*                           - the queue manager name (optional)            */
/*                                                                          */
/****************************************************************************

/****************************************************************************/
/* Includes                                                                 */
/****************************************************************************/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>

#include <cmqc.h>                       /* MQI                              */
#include <cmqcfc.h>                     /* PCF                              */
#include <cmqbc.h>                      /* MQAI                             */

/****************************************************************************/
/* Macros                                                                   */
/****************************************************************************/
#if MQAT_DEFAULT == MQAT_WINDOWS_NT
  #define Int64 "I64"
#elif defined(MQ_64_BIT)
  #define Int64 "l"
#else
  #define Int64 "ll"
#endif

/****************************************************************************/
/* Function prototypes                                                      */
/****************************************************************************/
void CheckCallResult(MQCHAR *, MQLONG , MQLONG);
void GetQEvents(MQHCONN, MQCHAR *);
int PrintBag(MQHBAG);
int PrintBagContents(MQHBAG, int);

/****************************************************************************/
/* Function: main                                                           */
/****************************************************************************/
int main(int argc, char *argv[])
{
   MQHCONN hConn;                       /* handle to connection            */
   MQCHAR QMName[MQ_Q_MGR_NAME_LENGTH+1]=""; /* default QM name            */
   MQLONG reason;                       /* reason code                     */
   MQLONG connReason;                   /* MQCONN reason code              */
   MQLONG compCode;                     /* completion code                 */

   /***************************************************************************/
   /* First check the required parameters                                     */
   /***************************************************************************/
   printf("Sample Event Monitor (times out after 30 secs)\n");
   if (argc < 2)
   {
     printf("Required parameter missing - event queue to be monitored\n");
     exit(99);
   }

   /***************************************************************************/
```

```
   /* Connect to the queue manager                                      */
   /*********************************************************************/
   if (argc > 2)
     strncpy(QMName, argv[2], (size_t)MQ_Q_MGR_NAME_LENGTH);
   MQCONN(QMName, &hConn, &compCode, &connReason);
   /*********************************************************************/
   /* Report the reason and stop if the connection failed               */
   /*********************************************************************/
   if (compCode == MQCC_FAILED)
   {
      CheckCallResult("MQCONN", compCode, connReason);
      exit( (int)connReason);
   }

   /*********************************************************************/
   /* Call the routine to open the event queue and format any event messages  */
   /* read from the queue.                                              */
   /*********************************************************************/
   GetQEvents(hConn, argv[1]);

   /*********************************************************************/
   /* Disconnect from the queue manager if not already connected        */
   /*********************************************************************/
   if (connReason != MQRC_ALREADY_CONNECTED)
   {
      MQDISC(&hConn, &compCode, &reason);
      CheckCallResult("MQDISC", compCode, reason);
   }

   return 0;

}

/*********************************************************************/
/*                                                                   */
/* Function: CheckCallResult                                         */
/*                                                                   */
/*********************************************************************/
/*                                                                   */
/* Input Parameters:  Description of call                            */
/*                    Completion code                                */
/*                    Reason code                                    */
/*                                                                   */
/* Output Parameters: None                                           */
/*                                                                   */
/* Logic: Display the description of the call, the completion code and the    */
/*        reason code if the completion code is not successful       */
/*                                                                   */
/*********************************************************************/
void  CheckCallResult(char *callText, MQLONG cc, MQLONG rc)
{
   if (cc != MQCC_OK)
         printf("%s failed: Completion Code = %d : Reason = %d\n",
                callText, cc, rc);

}

/*********************************************************************/
/*                                                                   */
/* Function: GetQEvents                                              */
/*                                                                   */
/*********************************************************************/
/*                                                                   */
/* Input Parameters:  Handle to the queue manager                    */
/*                    Name of the event queue to be monitored        */
/*                                                                   */
/* Output Parameters: None                                           */
/*
/* Logic:   Open the event queue.                                    */
/*          Get a message off the event queue and format the message into     */
/*          a bag.                                                   */
/*          A real event monitor would need to be programmed to deal with     */
/*          each type of event that it receives from the queue. This is       */
/*          outside the scope of this sample, so instead, the contents of     */
/*          the bag are printed.                                     */
/*          The program waits for 30 seconds for an event message and then    */
/*          terminates if no more messages are available.            */
/*                                                                   */
/*********************************************************************/
void GetQEvents(MQHCONN hConn, MQCHAR *qName)
{
   MQLONG openReason;                      /* MQOPEN reason code            */
```

```
MQLONG reason;                          /* reason code                    */
MQLONG compCode;                        /* completion code                */
MQHOBJ eventQueue;                      /* handle to event queue          */

MQHBAG eventBag = MQHB_UNUSABLE_HBAG;   /* event bag to receive event msg */
MQOD   od = {MQOD_DEFAULT};             /* Object Descriptor              */
MQMD   md = {MQMD_DEFAULT};             /* Message Descriptor             */
MQGMO  gmo = {MQGMO_DEFAULT};           /* get message options            */
MQLONG bQueueOK = 1;                    /* keep reading msgs while true   */

/*****************************************************************************/
/* Create an Event Bag in which to receive the event.                        */
/* Exit the function if the create fails.                                    */
/*****************************************************************************/
mqCreateBag(MQCBO_USER_BAG, &eventBag, &compCode, &reason);
CheckCallResult("Create event bag", compCode, reason);
if (compCode !=MQCC_OK)
   return;

/*****************************************************************************/
/* Open the event queue chosen by the user                                   */
/*****************************************************************************/
strncpy(od.ObjectName, qName, (size_t)MQ_Q_NAME_LENGTH);
MQOPEN(hConn, &od, MQOO_INPUT_AS_Q_DEF+MQOO_FAIL_IF_QUIESCING, &eventQueue,
       &compCode, &openReason);
CheckCallResult("Open event queue", compCode, openReason);

/*****************************************************************************/
/* Set the GMO options to control the action of the get message from the     */
/* queue.                                                                    */
/*****************************************************************************/
gmo.WaitInterval = 30000;               /* 30 second wait for message      */
gmo.Options = MQGMO_WAIT + MQGMO_FAIL_IF_QUIESCING + MQGMO_CONVERT;
gmo.Version = MQGMO_VERSION_2;          /* Avoid need to reset Message ID  */
gmo.MatchOptions = MQMO_NONE;           /* and Correlation ID after every  */
                                 /* mqGetBag
/*****************************************************************************/
/* If open fails, we cannot access the queue and must stop the monitor.      */
/*****************************************************************************/
if (compCode != MQCC_OK)
   bQueueOK = 0;

/*****************************************************************************/
/* Main loop to get an event message when it arrives                         */
/*****************************************************************************/
while (bQueueOK)
{
  printf("\nWaiting for an event\n");

  /***************************************************************************/
  /* Get the message from the event queue and convert it into the event      */
  /* bag.                                                                    */
  /***************************************************************************/
  mqGetBag(hConn, eventQueue, &md, &gmo, eventBag, &compCode, &reason);

  /***************************************************************************/
  /* If get fails, we cannot access the queue and must stop the monitor.     */
  /***************************************************************************/
  if (compCode != MQCC_OK)
  {
     bQueueOK = 0;

     /************************************************************************/
     /* If get fails because no message available then we have timed out,    */
     /* so report this, otherwise report an error.                           */
     /************************************************************************/
     if (reason == MQRC_NO_MSG_AVAILABLE)
     {
        printf("No more messages\n");
     }
     else
     {
        CheckCallResult("Get bag", compCode, reason);
     }
  }

  /***************************************************************************/
  /* Event message read - Print the contents of the event bag                */
  /***************************************************************************/
  else
  {
    if ( PrintBag(eventBag) )
```

```
            printf("\nError found while printing bag contents\n");

      }  /* end of msg found */
   } /* end of main loop */
   /*****************************************************************************/
   /* Close the event queue if successfully opened                              */
   /*****************************************************************************/
   if (openReason == MQRC_NONE)
   {
      MQCLOSE(hConn, &eventQueue, MQCO_NONE, &compCode, &reason);
      CheckCallResult("Close event queue", compCode, reason);
   }

   /*****************************************************************************/
   /* Delete the event bag if successfully created.                             */
   /*****************************************************************************/
   if (eventBag != MQHB_UNUSABLE_HBAG)
   {
      mqDeleteBag(&eventBag, &compCode, &reason);
      CheckCallResult("Delete the event bag", compCode, reason);
   }

} /* end of GetQEvents */

/*****************************************************************************/
/*                                                                           */
/* Function: PrintBag                                                        */
/*                                                                           */
/*****************************************************************************/
/*                                                                           */
/* Input Parameters:  Bag Handle                                             */
/*                                                                           */
/* Output Parameters: None                                                   */
/*                                                                           */
/* Returns:           Number of errors found                                 */
/*                                                                           */
/* Logic: Calls PrintBagContents to display the contents of the bag.         */
/*                                                                           */
/*****************************************************************************

int PrintBag(MQHBAG dataBag)
{
    int errors;

    printf("\n");
    errors = PrintBagContents(dataBag, 0);
    printf("\n");

    return errors;
}

/*****************************************************************************/
/*                                                                           */
/* Function: PrintBagContents                                                */
/*                                                                           */
/*****************************************************************************/
/*                                                                           */
/* Input Parameters:  Bag Handle                                             */
/*                    Indentation level of bag                               */
/*                                                                           */
/* Output Parameters: None                                                   */
/*                                                                           */
/* Returns:           Number of errors found                                 */
/*                                                                           */
/* Logic: Count the number of items in the bag                               */
/*        Obtain selector and item type for each item in the bag.            */
/*        Obtain the value of the item depending on item type and display the */
/*        index of the item, the selector and the value.                     */
/*        If the item is an embedded bag handle then call this function again */
/*        to print the contents of the embedded bag increasing the           */
/*        indentation level.                                                 */
/*                                                                           */
/*****************************************************************************/
int PrintBagContents(MQHBAG dataBag, int indent)
{
   /*****************************************************************************/
   /* Definitions                                                               */
   /*****************************************************************************/
   #define LENGTH 500                     /* Max length of string to be read*/
   #define INDENT 4                       /* Number of spaces to indent     */
                                          /* embedded bag display           */
```

```
/*****************************************************************/
/* Variables                                                     */
/*****************************************************************/
MQLONG  itemCount;                      /* Number of items in the bag     */
MQLONG  itemType;                       /* Type of the item               */
int     i;                              /* Index of item in the bag       */
MQCHAR  stringVal[LENGTH+1];            /* Value if item is a string      */
MQBYTE  byteStringVal[LENGTH];          /* Value if item is a byte string */
MQLONG  stringLength;                   /* Length of string value         */
MQLONG  ccsid;                          /* CCSID of string value          */
MQINT32 iValue;                         /* Value if item is an integer    */
MQINT64 i64Value;                       /* Value if item is a 64-bit      */
                                        /* integer                        */
MQLONG  selector;                       /* Selector of item               */
MQHBAG  bagHandle;                      /* Value if item is a bag handle  */
MQLONG  reason;                         /* reason code                    */
MQLONG  compCode;                       /* completion code                */
MQLONG  trimLength;                     /* Length of string to be trimmed */
int     errors = 0;                     /* Count of errors found          */
char    blanks[] = "                      "; /* Blank string used to      */
                                             /* indent display            */

/*****************************************************************/
/* Count the number of items in the bag                          */
/*****************************************************************/
mqCountItems(dataBag, MQSEL_ALL_SELECTORS, &itemCount, &compCode, &reason);

if (compCode != MQCC_OK)
   errors++;
else
{
   printf("
   printf("
   printf("
}

/*****************************************************************/
/* If no errors found, display each item in the bag              */
/*****************************************************************/
if (!errors)
{
   for (i = 0; i < itemCount; i++)
   {

       /*************************************************************/
       /* First inquire the type of the item for each item in the bag */
       /*************************************************************/
       mqInquireItemInfo(dataBag,             /* Bag handle          */
                         MQSEL_ANY_SELECTOR,  /* Item can have any selector*/
                         i,                   /* Index position in the bag */
                         &selector,           /* Actual value of selector  */
                                              /* returned by call          */
                         &itemType,           /* Actual type of item       */
                                              /* returned by call          */
                         &compCode,           /* Completion code           */
                         &reason);            /* Reason Code               */

       if (compCode != MQCC_OK)
          errors++;

       switch(itemType)
       {
       case MQITEM_INTEGER:
            /*********************************************************/
            /* Item is an integer. Find its value and display its index,   */
            /* selector and value.                                   */
            /*********************************************************/
            mqInquireInteger(dataBag,         /* Bag handle          */
                             MQSEL_ANY_SELECTOR, /* Allow any selector     */
                             i,               /* Index position in the bag */
                             &iValue,         /* Returned integer value
                             &compCode,       /* Completion code           */
                             &reason);        /* Reason Code               */

            if (compCode != MQCC_OK)
               errors++;
            else
               printf("%.*s  %-2d     %-4d     (%d)\n",
                       indent, blanks, i, selector, iValue);
            break
```

```
         case MQITEM_INTEGER64:
              /*************************************************************/
              /* Item is a 64-bit integer. Find its value and display its  */
              /* index, selector and value.                                */
              /*************************************************************/
              mqInquireInteger64(dataBag,       /* Bag handle              */
                                 MQSEL_ANY_SELECTOR, /* Allow any selector */
                                 i,             /* Index position in the bag */
                                 &i64Value,     /* Returned integer value  */
                                 &compCode,     /* Completion code         */
                                 &reason);      /* Reason Code             */

              if (compCode != MQCC_OK)
                 errors++;
              else
                 printf("%.*s  %-2d     %-4d     (%"Int64"d)\n",
                         indent, blanks, i, selector, i64Value);
              break;


            case MQITEM_STRING:
              /*************************************************************/
              /* Item is a string. Obtain the string in a buffer, prepare  */
              /* the string for displaying and display the index, selector,*/
              /* string and Character Set ID.                              */
              /*************************************************************/
              mqInquireString(dataBag,          /* Bag handle              */
                              MQSEL_ANY_SELECTOR, /* Allow any selector    */
                              i,                /* Index position in the bag */
                              LENGTH,           /* Maximum length of buffer */
                              stringVal,        /* Buffer to receive string */
                              &stringLength,    /* Actual length of string */
                              &ccsid,           /* Coded character set ID  */
                              &compCode,        /* Completion code         */
                              &reason);         /* Reason Code             */

              /*************************************************************/
              /* The call can return a warning if the string is too long for */
              /* the output buffer and has been truncated, so only check   */
              /* explicitly for call failure.                              */
              /*************************************************************/
              if (compCode == MQCC_FAILED)
                  errors++;
              else
              {
                 /*********************************************************/
                 /* Remove trailing blanks from the string and terminate with*/
                 /* a null. First check that the string should not have been */
                 /* longer than the maximum buffer size allowed.          */
                 /*********************************************************/
                 if (stringLength > LENGTH)
                    trimLength = LENGTH;
                 else
                    trimLength = stringLength;
                 mqTrim(trimLength, stringVal, stringVal, &compCode, &reason);
                 printf("%.*s  %-2d     %-4d     '%s' %d\n",
                         indent, blanks, i, selector, stringVal, ccsid);
              }
              break;

         case MQITEM_BYTE_STRING:
              /*************************************************************/
              /* Item is a byte string. Obtain the byte string in a buffer,*/
              /* prepare the byte string for displaying and display the    */
              /* index, selector and string.                               */
              /*************************************************************/
              mqInquireByteString(dataBag,      /* Bag handle              */
                                  MQSEL_ANY_SELECTOR, /* Allow any selector */
                                  i,            /* Index position in the bag */
                                  LENGTH,       /* Maximum length of buffer */
                                  byteStringVal, /* Buffer to receive string */
                                  &stringLength, /* Actual length of string */
                                  &compCode,    /* Completion code         */
                                  &reason);     /* Reason Code

              /*************************************************************/
              /* The call can return a warning if the string is too long for */
              /* the output buffer and has been truncated, so only check   */
              /* explicitly for call failure.                              */
              /*************************************************************/
              if (compCode == MQCC_FAILED)
                  errors++;
```

```
               else
               {
                  printf("%.*s  %-2d     %-4d     X'",
                         indent, blanks, i, selector);

                  for (i = 0 ; i < stringLength ; i++)
                     printf("

                  printf("'\n");
               }
               break;

          case MQITEM_BAG:
               /*************************************************************/
               /* Item is an embedded bag handle, so call the PrintBagContents*/
               /* function again to display the contents.                   */
               /*************************************************************/
               mqInquireBag(dataBag,             /* Bag handle              */
                            MQSEL_ANY_SELECTOR,  /* Allow any selector      */
                            i,                   /* Index position in the bag */
                            &bagHandle,          /* Returned embedded bag hdle*/
                            &compCode,           /* Completion code         */
                            &reason);            /* Reason Code             */

               if (compCode != MQCC_OK)
                  errors++;
               else
               {
                  printf("%.*s  %-2d     %-4d     (%d)\n", indent, blanks, i,
                          selector, bagHandle);
                  if (selector == MQHA_BAG_HANDLE)
                     printf("
                  else
                     printf("
                  PrintBagContents(bagHandle, indent+INDENT);
               }
               break;

          default:
               printf("
          }
      }
   }
   return errors;
}
```

Multi Programa C de ejemplo para realizar consultas sobre objetos de canal (amqsaicl.c)

El programa C de ejemplo `amqsaicl.c` consulta los objetos de canal utilizando la MQAI.

```
/*****************************************************************************/
/*                                                                           */
/* Program name: AMQSAICL.C                                                  */
/*                                                                           */
/* Description:  Sample C program to inquire channel objects                 */
/*               using the IBM MQ Administration Interface (MQAI)            */
/*                                                                           */
/* <N_OCO_COPYRIGHT>                                                         */
/* Licensed Materials - Property of IBM                                      */
/*                                                                           */
/* 63H9336                                                                   */
/* (c) 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.                                                                 */
/* <NOC_COPYRIGHT>                                                           */
/*****************************************************************************/
/*                                                                           */
/* Function:                                                                 */
/*    AMQSAICL is a sample C program that demonstrates how to inquire        */
/*    attributes of the local queue manager using the MQAI interface. In     */
/*    particular, it inquires all channels and their types.                  */
/*                                                                           */
/*     - A PCF command is built from items placed into an MQAI administration */
/*       bag.                                                                */
```

```
/*       These are:-                                                          */
/*            - The generic channel name "*"                                  */
/*            - The attributes to be inquired. In this sample we just want    */
/*              name and type attributes                                      */
/*                                                                            */
/*     - The mqExecute MQCMD_INQUIRE_CHANNEL call is executed.                */
/*       The call generates the correct PCF structure.                        */
/*       The default options to the call are used so that the command is sent */
/*       to the SYSTEM.ADMIN.COMMAND.QUEUE.                                   */
/*       The reply from the command server is placed on a temporary dynamic   */
/*       queue.                                                               */
/*       The reply from the MQCMD_INQUIRE_CHANNEL is read from the            */
/*       temporary queue and formatted into the response bag.                 */
/*                                                                            */
/*     - The completion code from the mqExecute call is checked and if there  */
/*       is a failure from the command server, then the code returned by the  */
/*       command server is retrieved from the system bag that has been        */
/*       embedded in the response bag to the mqExecute call.                  */
/*                                                                            */
/* Note: The command server must be running.                                  */
/*                                                                            */
/******************************************************************************/
/*                                                                            */
/* AMQSAICL has 2 parameter - the queue manager name (optional)               */
/*                          - output file (optional) default varies           */
/******************************************************************************/

/******************************************************************************/
/* Includes                                                                   */
/******************************************************************************/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>
#if (MQAT_DEFAULT == MQAT_OS400)
#include <recio.h>
#endif

#include <cmqc.h>                       /* MQI                                */
#include <cmqcfc.h>                     /* PCF                                */
#include <cmqbc.h>                      /* MQAI                               */
#include <cmqxc.h>                      /* MQCD                               */

/******************************************************************************/
/* Function prototypes                                                        */
/******************************************************************************/
void CheckCallResult(MQCHAR *, MQLONG , MQLONG);

/******************************************************************************/
/* DataTypes                                                                  */
/******************************************************************************/
#if (MQAT_DEFAULT == MQAT_OS400)
typedef _RFILE OUTFILEHDL;
#else
typedef FILE OUTFILEHDL;
#endif

/******************************************************************************/
/* Constants                                                                  */
/******************************************************************************/
#if (MQAT_DEFAULT == MQAT_OS400)
const struct
{
  char name[9];
} ChlTypeMap[9] =
{
  "*SDR     ",    /* MQCHT_SENDER    */
  "*SVR     ",    /* MQCHT_SERVER    */
  "*RCVR    ",    /* MQCHT_RECEIVER  */
  "*RQSTR   ",    /* MQCHT_REQUESTER */
  "*ALL     ",    /* MQCHT_ALL       */
  "*CLTCN   ",    /* MQCHT_CLNTCONN  */
  "*SVRCONN ",    /* MQCHT_SVRCONN   */
  "*CLUSRCVR",    /* MQCHT_CLUSRCVR  */
  "*CLUSSDR "     /* MQCHT_CLUSSDR   */
};
#else
const struct
{
  char name[9];
} ChlTypeMap[9] =
{
```

```
  "sdr      ",    /* MQCHT_SENDER    */
  "svr      ",    /* MQCHT_SERVER    */
  "rcvr     ",    /* MQCHT_RECEIVER  */
  "rqstr    ",    /* MQCHT_REQUESTER */
  "all      ",    /* MQCHT_ALL       */
  "cltconn  ",    /* MQCHT_CLNTCONN  */
  "svrcn    ",    /* MQCHT_SVRCONN   */
  "clusrcvr ",    /* MQCHT_CLUSRCVR  */
  "clussdr  "     /* MQCHT_CLUSSDR   */
};
#endif

/********************************************************************************/
/* Macros                                                                       */
/********************************************************************************/
#if (MQAT_DEFAULT == MQAT_OS400)
  #define OUTFILE "QTEMP/AMQSAICL(AMQSAICL)"
  #define OPENOUTFILE(hdl, fname) \
    (hdl) = _Ropen((fname),"wr, rtncode=Y");
  #define CLOSEOUTFILE(hdl) \
    _Rclose((hdl));
  #define WRITEOUTFILE(hdl, buf, buflen) \
    _Rwrite((hdl),(buf),(buflen));

#elif (MQAT_DEFAULT == MQAT_UNIX)
  #define OUTFILE "/tmp/amqsaicl.txt"
  #define OPENOUTFILE(hdl, fname) \
    (hdl) = fopen((fname),"w");
  #define CLOSEOUTFILE(hdl) \
    fclose((hdl));
  #define WRITEOUTFILE(hdl, buf, buflen) \
    fwrite((buf),(buflen),1,(hdl)); fflush((hdl));

#else
  #define OUTFILE "amqsaicl.txt"
  #define OPENOUTFILE(fname) \
    fopen((fname),"w");
  #define CLOSEOUTFILE(hdl) \
    fclose((hdl));
  #define WRITEOUTFILE(hdl, buf, buflen) \
    fwrite((buf),(buflen),1,(hdl)); fflush((hdl));

#endif

#define ChlType2String(t) ChlTypeMap[(t)-1].name

/********************************************************************************/
/* Function: main                                                               */
/********************************************************************************/
int main(int argc, char *argv[])
{
   /*****************************************************************************/
   /* MQAI variables                                                            */
   /*****************************************************************************/
   MQHCONN hConn;                       /* handle to MQ connection         */
   MQCHAR qmName[MQ_Q_MGR_NAME_LENGTH+1]=""; /* default QMgr name             */
   MQLONG reason;                       /* reason code                     */
   MQLONG connReason;                   /* MQCONN reason code              */
   MQLONG compCode;                     /* completion code                 */
   MQHBAG adminBag = MQHB_UNUSABLE_HBAG;   /* admin bag for mqExecute         */
   MQHBAG responseBag = MQHB_UNUSABLE_HBAG;/* response bag for mqExecute      */
   MQHBAG cAttrsBag;                    /* bag containing chl attributes   */
   MQHBAG errorBag;                     /* bag containing cmd server error */
   MQLONG mqExecuteCC;                  /* mqExecute completion code       */
   MQLONG mqExecuteRC;                  /* mqExecute reason code           */
   MQLONG chlNameLength;                /* Actual length of chl name       */
   MQLONG chlType;                      /* Channel type                    */
   MQLONG i;                            /* loop counter                    */
   MQLONG numberOfBags;                 /* number of bags in response bag  */
   MQCHAR chlName[MQ_OBJECT_NAME_LENGTH+1];/* name of chl extracted from bag  */
   MQCHAR OutputBuffer[100];            /* output data buffer              */
   OUTFILEHDL *outfp = NULL;            /* output file handle              */

   /*****************************************************************************/
   /* Connect to the queue manager                                              */
   /*****************************************************************************/
   if (argc &gt; 1)
      strncpy(qmName, argv[1], (size_t)MQ_Q_MGR_NAME_LENGTH);
   MQCONN(qmName, &hConn;, &compCode;, &connReason;);

   /*****************************************************************************/
   /* Report the reason and stop if the connection failed.                      */
```

```
/******************************************************************************/
if (compCode == MQCC_FAILED)
{
   CheckCallResult("Queue Manager connection", compCode, connReason);
   exit( (int)connReason);
}

/******************************************************************************/
/* Open the output file                                                       */
/******************************************************************************/
if (argc &gt; 2)
{
  OPENOUTFILE(outfp, argv[2]);
}
else
{
  OPENOUTFILE(outfp, OUTFILE);
}

if(outfp == NULL)
{
  printf("Could not open output file.\n");
  goto MOD_EXIT;
}
/******************************************************************************/
/* Create an admin bag for the mqExecute call                                 */
/******************************************************************************/
mqCreateBag(MQCBO_ADMIN_BAG, &adminBag;, &compCode;, &reason;);
CheckCallResult("Create admin bag", compCode, reason);

/******************************************************************************/
/* Create a response bag for the mqExecute call                               */
/******************************************************************************/
mqCreateBag(MQCBO_ADMIN_BAG, &responseBag;, &compCode;, &reason;);
CheckCallResult("Create response bag", compCode, reason);

/******************************************************************************/
/* Put the generic channel name into the admin bag                            */
/******************************************************************************/
mqAddString(adminBag, MQCACH_CHANNEL_NAME, MQBL_NULL_TERMINATED, "*",
            &compCode;, &reason;);
CheckCallResult("Add channel name", compCode, reason);

/******************************************************************************/
/* Put the channel type into the admin bag                                    */
/******************************************************************************/
mqAddInteger(adminBag, MQIACH_CHANNEL_TYPE, MQCHT_ALL, &compCode;, &reason;);
CheckCallResult("Add channel type", compCode, reason);

/******************************************************************************/
/* Add an inquiry for various attributes                                      */
/******************************************************************************/
mqAddInquiry(adminBag, MQIACH_CHANNEL_TYPE, &compCode;, &reason;);
CheckCallResult("Add inquiry", compCode, reason);

/******************************************************************************/
/* Send the command to find all the channel names and channel types.          */
/* The mqExecute call creates the PCF structure required, sends it to         */
/* the command server, and receives the reply from the command server into    */
/* the response bag. The attributes are contained in system bags that are     */
/* embedded in the response bag, one set of attributes per bag.               */
/******************************************************************************/
mqExecute(hConn,                    /* MQ connection handle                   */
          MQCMD_INQUIRE_CHANNEL,    /* Command to be executed                 */
          MQHB_NONE,                /* No options bag                         */
          adminBag,                 /* Handle to bag containing commands      */
          responseBag,              /* Handle to bag to receive the response*/
          MQHO_NONE,                /* Put msg on SYSTEM.ADMIN.COMMAND.QUEUE*/
          MQHO_NONE,                /* Create a dynamic q for the response    */
          &compCode;,                /* Completion code from the mqexecute     */
          &reason;);                 /* Reason code from mqexecute call        */

/******************************************************************************/
/* Check the command server is started. If not exit.                          */
/******************************************************************************/
if (reason == MQRC_CMD_SERVER_NOT_AVAILABLE)
{
   printf("Please start the command server: <strmqcsv QMgrName="">\n");
   goto MOD_EXIT;
}

/******************************************************************************/
```

```
   /* Check the result from mqExecute call. If successful find the channel    */
   /* types for all the channels. If failed find the error.                   */
   /*************************************************************************/
   if ( compCode == MQCC_OK )                      /* Successful mqExecute    */
   {
     /***********************************************************************/
     /* Count the number of system bags embedded in the response bag from the */
     /* mqExecute call. The attributes for each channel are in separate bags. */
     /***********************************************************************/
     mqCountItems(responseBag, MQHA_BAG_HANDLE, &numberOfBags;,
                  &compCode;, &reason;);
     CheckCallResult("Count number of bag handles", compCode, reason);

     for ( i=0; i<numberOfbags; i++)
     {
       /*********************************************************************/
       /* Get the next system bag handle out of the mqExecute response bag.   */
       /* This bag contains the channel attributes                            */
       /*********************************************************************/
       mqInquireBag(responseBag, MQHA_BAG_HANDLE, i, &cAttrsbag,
                    &compCode, &reason);
       CheckCallResult("Get the result bag handle", compCode, reason);

       /*********************************************************************/
       /* Get the channel name out of the channel attributes bag              */
       /*********************************************************************/
       mqInquireString(cAttrsBag, MQCACH_CHANNEL_NAME, 0, MQ_OBJECT_NAME_LENGTH,
                       chlName, &chlNameLength, NULL, &compCode, &reason);
       CheckCallResult("Get channel name", compCode, reason);

       /*********************************************************************/
       /* Get the channel type out of the channel attributes bag              */
       /*********************************************************************/

    mqInquireInteger(cAttrsBag, MQIACH_CHANNEL_TYPE, MQIND_NONE, &chlType,
                        &compCode, &reason);
       CheckCallResult("Get type", compCode, reason);

       /*********************************************************************/
       /* Use mqTrim to prepare the channel name for printing.                */
       /* Print the result.                                                   */
       /*********************************************************************/
       mqTrim(MQ_CHANNEL_NAME_LENGTH, chlName, chlName, &compCode, &reason);
       sprintf(OutputBuffer, "%-20s%-9s", chlName, ChlType2String(chlType));
       WRITEOUTFILE(outfp,OutputBuffer,29)
     }
   }

   else                                             /* Failed mqExecute     */
   {
     printf("Call to get channel attributes failed: Cc = %ld : Rc = %ld\n",
                  compCode, reason);
     /*********************************************************************/
     /* If the command fails get the system bag handle out of the mqexecute   */
     /* response bag.This bag contains the reason from the command server     */
     /* why the command failed.                                             */
     /*********************************************************************/
     if (reason == MQRCCF_COMMAND_FAILED)
     {
       mqInquireBag(responseBag, MQHA_BAG_HANDLE, 0, &errorBag,
                    &compCode, &reason);
       CheckCallResult("Get the result bag handle", compCode, reason);

       /*********************************************************************/
       /* Get the completion code and reason code, returned by the command    */
       /* server, from the embedded error bag.                                */
       /*********************************************************************/
       mqInquireInteger(errorBag, MQIASY_COMP_CODE, MQIND_NONE, &mqExecuteCC,
                        &compCode, &reason );
       CheckCallResult("Get the completion code from the result bag",
                       compCode, reason);
       mqInquireInteger(errorBag, MQIASY_REASON, MQIND_NONE, &mqExecuteRC,
                         &compCode, &reason);
       CheckCallResult("Get the reason code from the result bag",
                       compCode, reason);
       printf("Error returned by the command server: Cc = %ld : Rc = %ld\n",
                mqExecuteCC, mqExecuteRC);
     }
   }

MOD_EXIT:
   /*************************************************************************/
```

```
   /* Delete the admin bag if successfully created.                          */
   /*************************************************************************/
   if (adminBag != MQHB_UNUSABLE_HBAG)
   {
      mqDeleteBag(&adminBag, &compCode, &reason);
      CheckCallResult("Delete the admin bag", compCode, reason);
   }

   /*************************************************************************/
   /* Delete the response bag if successfully created.                      */
   /*************************************************************************/
   if (responseBag != MQHB_UNUSABLE_HBAG)
   {
      mqDeleteBag(&responseBag, &compCode, &reason);
      CheckCallResult("Delete the response bag", compCode, reason);
   }

   /*************************************************************************/
   /* Disconnect from the queue manager if not already connected            */
   /*************************************************************************/
   if (connReason != MQRC_ALREADY_CONNECTED)
   {
      MQDISC(&hConn, &compCode, &reason);
      CheckCallResult("Disconnect from Queue Manager", compCode, reason);
   }

   /*************************************************************************/
   /* Close the output file if open                                         */
   /*************************************************************************/
   if(outfp != NULL)
      CLOSEOUTFILE(outfp);

   return 0;
}


/*************************************************************************/
/*                                                                       */
/* Function: CheckCallResult                                             */
/*                                                                       */
/*************************************************************************/
/*                                                                       */
/* Input Parameters:  Description of call                                */
/*                    Completion code                                    */
/*                    Reason code                                        */
/*                                                                       */
/* Output Parameters: None                                               */
/*                                                                       */
/* Logic: Display the description of the call, the completion code and the    */
/*        reason code if the completion code is not successful           */
/*                                                                       */
/*************************************************************************/
void  CheckCallResult(char *callText, MQLONG cc, MQLONG rc)
{
   if (cc != MQCC_OK)
         printf("%s failed: Completion Code = %ld : Reason = %ld\n", callText,
                cc, rc);
}
```

Multi Programa C de ejemplo para realizar consultas sobre colas e imprimir información (amqsailq.c)

El programa C de ejemplo `amqsailq.c` indaga la profundidad actual de las colas locales utilizando MQAI.

```
/*************************************************************************/
/*                                                                       */
/* Program name: AMQSAILQ.C                                              */
/*                                                                       */
/* Description:  Sample C program to inquire the current depth of the local   */
/*               queues using the IBM MQ Administration Interface (MQAI)     */
/*                                                                       */
/* Statement:    Licensed Materials - Property of IBM                    */
/*                                                                       */
/*               84H2000, 5765-B73                                       */
/*               84H2001, 5639-B42                                       */
/*               84H2002, 5765-B74                                       */
/*               84H2003, 5765-B75                                       */
```

```
/*              84H2004, 5639-B43                                             */
/*                                                                            */
/*              (C) Copyright IBM Corp. 1999, 2024                            */
/*                                                                            */
/******************************************************************************/
/*                                                                            */
/* Function:                                                                  */
/*    AMQSAILQ is a sample C program that demonstrates how to inquire         */
/*    attributes of the local queue manager using the MQAI interface. In      */
/*    particular, it inquires the current depths of all the local queues.     */
/*                                                                            */
/*     - A PCF command is built by placing items into an MQAI administration  */
/*       bag.                                                                 */
/*       These are:-                                                          */
/*            - The generic queue name "*"                                    */
/*            - The type of queue required. In this sample we want to         */
/*              inquire local queues.                                         */
/*            - The attribute to be inquired. In this sample we want the      */
/*              current depths.                                               */
/*                                                                            */
/*     - The mqExecute call is executed with the command MQCMD_INQUIRE_Q.     */
/*       The call generates the correct PCF structure.                        */
/*       The default options to the call are used so that the command is sent */
/*       to the SYSTEM.ADMIN.COMMAND.QUEUE.                                   */
/*       The reply from the command server is placed on a temporary dynamic   */
/*       queue.                                                               */
/*       The reply from the MQCMD_INQUIRE_Q command is read from the          */
/*       temporary queue and formatted into the response bag.                 */
/*                                                                            */
/*     - The completion code from the mqExecute call is checked and if there  */
/*       is a failure from the command server, then the code returned by      */
/*       command server is retrieved from the system bag that has been        */
/*       embedded in the response bag to the mqExecute call.                  */
/*                                                                            */
/*     - If the call is successful, the depth of each local queue is placed   */
/*       in system bags embedded in the response bag of the mqExecute call.   */
/*       The name and depth of each queue is obtained from each of the bags   */
/*       and the result displayed on the screen.                              */
/*                                                                            */
/* Note: The command server must be running.                                  */
/*                                                                            */
/******************************************************************************/
/*                                                                            */
/* AMQSAILQ has 1 parameter - the queue manager name (optional)               */
/*                                                                            */
/******************************************************************************/

/******************************************************************************/
/* Includes                                                                   */
/******************************************************************************/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>

#include <cmqc.h>                       /* MQI                                */
#include <cmqcfc.h>                     /* PCF                                */
#include <cmqbc.h>                      /* MQAI                               */

/******************************************************************************/
/* Function prototypes                                                        */
/******************************************************************************/
void CheckCallResult(MQCHAR *, MQLONG , MQLONG);

/******************************************************************************/
/* Function: main                                                             */
/******************************************************************************/
int main(int argc, char *argv[])
{
   /***************************************************************************/
   /* MQAI variables                                                          */
   /***************************************************************************/
   MQHCONN hConn;                       /* handle to IBM MQ connection     */
   MQCHAR qmName[MQ_Q_MGR_NAME_LENGTH+1]=""; /* default QMgr name          */
   MQLONG reason;                       /* reason code                     */
   MQLONG connReason;                   /* MQCONN reason code              */
   MQLONG compCode;                     /* completion code                 */
   MQHBAG adminBag = MQHB_UNUSABLE_HBAG;   /* admin bag for mqExecute      */
   MQHBAG responseBag = MQHB_UNUSABLE_HBAG;/* response bag for mqExecute    */
   MQHBAG qAttrsBag;                    /* bag containing q attributes     */
   MQHBAG errorBag;                     /* bag containing cmd server error */
   MQLONG mqExecuteCC;                  /* mqExecute completion code       */
```

```
MQLONG mqExecuteRC;                     /* mqExecute reason code           */
MQLONG qNameLength;                     /* Actual length of q name         */
MQLONG qDepth;                          /* depth of queue                  */
MQLONG i;                               /* loop counter                    */
MQLONG numberOfBags;                    /* number of bags in response bag  */
MQCHAR qName[MQ_Q_NAME_LENGTH+1];       /* name of queue extracted from bag*/


printf("Display current depths of local queues\n\n");

/*****************************************************************************/
/* Connect to the queue manager                                              */
/*****************************************************************************/
if (argc > 1)
   strncpy(qmName, argv[1], (size_t)MQ_Q_MGR_NAME_LENGTH);
MQCONN(qmName, &hConn, &compCode, &connReason);

/*****************************************************************************/
/* Report the reason and stop if the connection failed.                      */
/*****************************************************************************/
if (compCode == MQCC_FAILED)
{
   CheckCallResult("Queue Manager connection", compCode, connReason);
   exit( (int)connReason);
}

/*****************************************************************************/
/* Create an admin bag for the mqExecute call                                */
/*****************************************************************************/
mqCreateBag(MQCBO_ADMIN_BAG, &adminBag, &compCode, &reason);
CheckCallResult("Create admin bag", compCode, reason);
/*****************************************************************************/
/* Create a response bag for the mqExecute call                              */
/*****************************************************************************/
mqCreateBag(MQCBO_ADMIN_BAG, &responseBag, &compCode, &reason);
CheckCallResult("Create response bag", compCode, reason);

/*****************************************************************************/
/* Put the generic queue name into the admin bag                             */
/*****************************************************************************/
mqAddString(adminBag, MQCA_Q_NAME, MQBL_NULL_TERMINATED, "*",
            &compCode, &reason);
CheckCallResult("Add q name", compCode, reason);

/*****************************************************************************/
/* Put the local queue type into the admin bag                               */
/*****************************************************************************/
mqAddInteger(adminBag, MQIA_Q_TYPE, MQQT_LOCAL, &compCode, &reason);
CheckCallResult("Add q type", compCode, reason);

/*****************************************************************************/
/* Add an inquiry for current queue depths                                   */
/*****************************************************************************/
mqAddInquiry(adminBag, MQIA_CURRENT_Q_DEPTH, &compCode, &reason);
CheckCallResult("Add inquiry", compCode, reason);

/*****************************************************************************/
/* Send the command to find all the local queue names and queue depths.      */
/* The mqExecute call creates the PCF structure required, sends it to        */
/* the command server, and receives the reply from the command server into   */
/* the response bag. The attributes are contained in system bags that are    */
/* embedded in the response bag, one set of attributes per bag.              */
/*****************************************************************************/
mqExecute(hConn,                    /* IBM MQ connection handle              */
          MQCMD_INQUIRE_Q,          /* Command to be executed                */
          MQHB_NONE,                /* No options bag                        */
          adminBag,                 /* Handle to bag containing commands     */
          responseBag,              /* Handle to bag to receive the response*/
          MQHO_NONE,                /* Put msg on SYSTEM.ADMIN.COMMAND.QUEUE*/
          MQHO_NONE,                /* Create a dynamic q for the response   */
          &compCode,                /* Completion code from the mqExecute    */
          &reason);                 /* Reason code from mqExecute call       */


/*****************************************************************************/
/* Check the command server is started. If not exit.                         */
/*****************************************************************************/
if (reason == MQRC_CMD_SERVER_NOT_AVAILABLE)
{
   printf("Please start the command server: <strmqcsv QMgrName>\n");
   MQDISC(&hConn, &compCode, &reason);
   CheckCallResult("Disconnect from Queue Manager", compCode, reason);
```

```
    exit(98);
  }

  /*****************************************************************/
  /* Check the result from mqExecute call. If successful find the current    */
  /* depths of all the local queues. If failed find the error.               */
  /*****************************************************************/
  if ( compCode == MQCC_OK )                    /* Successful mqExecute    */
  {
    /***************************************************************/
    /* Count the number of system bags embedded in the response bag from the */
    /* mqExecute call. The attributes for each queue are in a separate bag.  */
    /***************************************************************/
    mqCountItems(responseBag, MQHA_BAG_HANDLE, &numberOfBags, &compCode,
                 &reason);
    CheckCallResult("Count number of bag handles", compCode, reason);

    for ( i=0; i<numberOfBags; i++)
    {
      /*************************************************************/
      /* Get the next system bag handle out of the mqExecute response bag.   */
      /* This bag contains the queue attributes                              */
      /*************************************************************/
      mqInquireBag(responseBag, MQHA_BAG_HANDLE, i, &qAttrsBag, &compCode,
                   &reason);
      CheckCallResult("Get the result bag handle", compCode, reason);

      /*************************************************************/
      /* Get the queue name out of the queue attributes bag                  */
      /*************************************************************/
      mqInquireString(qAttrsBag, MQCA_Q_NAME, 0, MQ_Q_NAME_LENGTH, qName,
                      &qNameLength, NULL, &compCode, &reason);
      CheckCallResult("Get queue name", compCode, reason);

      /*************************************************************/
      /* Get the depth out of the queue attributes bag                       */
      /*************************************************************/
      mqInquireInteger(qAttrsBag, MQIA_CURRENT_Q_DEPTH, MQIND_NONE, &qDepth,
                       &compCode, &reason);
      CheckCallResult("Get depth", compCode, reason);

      /*************************************************************/
      /* Use mqTrim to prepare the queue name for printing.                  */
      /* Print the result.                                                   */
      /*************************************************************/
      mqTrim(MQ_Q_NAME_LENGTH, qName, qName, &compCode, &reason);
      printf("%4d  %-48s\n", qDepth, qName);
    }
  }

  else                                          /* Failed mqExecute     */
  {
    printf("Call to get queue attributes failed: Completion Code = %d :
            Reason = %d\n", compCode, reason);

    /***************************************************************/
    /* If the command fails get the system bag handle out of the mqExecute   */
    /* response bag. This bag contains the reason from the command server    */
    /* why the command failed.                                             */
    /***************************************************************/
    if (reason == MQRCCF_COMMAND_FAILED)
    {
      mqInquireBag(responseBag, MQHA_BAG_HANDLE, 0, &errorBag, &compCode,
                   &reason);
      CheckCallResult("Get the result bag handle", compCode, reason);

     /*************************************************************/
     /* Get the completion code and reason code, returned by the command     */
     /* server, from the embedded error bag.                                */
     /*************************************************************/
     mqInquireInteger(errorBag, MQIASY_COMP_CODE, MQIND_NONE, &mqExecuteCC,
                      &compCode, &reason );
     CheckCallResult("Get the completion code from the result bag",
                      compCode, reason);
     mqInquireInteger(errorBag, MQIASY_REASON, MQIND_NONE, &mqExecuteRC,
                      &compCode, &reason);
     CheckCallResult("Get the reason code from the result bag",
                      compCode, reason);
     printf("Error returned by the command server: Completion Code = %d :
             Reason = %d\n", mqExecuteCC, mqExecuteRC);
   }
  }
```

```
  /******************************************************************************/
  /* Delete the admin bag if successfully created.                              */
  /******************************************************************************/
  if (adminBag != MQHB_UNUSABLE_HBAG)
  {
     mqDeleteBag(&adminBag, &compCode, &reason);
     CheckCallResult("Delete the admin bag", compCode, reason);
  }

  /******************************************************************************/
  /* Delete the response bag if successfully created.                           */
  /******************************************************************************/
  if (responseBag != MQHB_UNUSABLE_HBAG)
  {
     mqDeleteBag(&responseBag, &compCode, &reason);
     CheckCallResult("Delete the response bag", compCode, reason);
  }

  /******************************************************************************/
  /* Disconnect from the queue manager if not already connected                 */
  /******************************************************************************/
  if (connReason != MQRC_ALREADY_CONNECTED)
  {
     MQDISC(&hConn, &compCode, &reason);
      CheckCallResult("Disconnect from queue manager", compCode, reason);
  }
  return 0;
}

/******************************************************************************/
*                                                                             */
* Function: CheckCallResult                                                   */
*                                                                             */
/******************************************************************************/
*                                                                             */
* Input Parameters:  Description of call                                      */
*                    Completion code                                          */
*                    Reason code                                              */
*                                                                             */
* Output Parameters: None                                                     */
*                                                                             */
* Logic: Display the description of the call, the completion code and the     */
*        reason code if the completion code is not successful                 */
*                                                                             */
/******************************************************************************/
void  CheckCallResult(char *callText, MQLONG cc, MQLONG rc)
{
  if (cc != MQCC_OK)
        printf("%s failed: Completion Code = %d : Reason = %d\n",
                callText, cc, rc);
}
```

Multi Paquetes de datos y MQAI

Un paquete de datos es una manera de manejar propiedades o parámetros de objetos utilizando la Interfaz de administración de IBM MQ (MQAI).

Paquetes de datos

- El paquete de datos contiene cero o más *elementos de datos*. Estos elementos de datos se ordenan dentro del paquete cuando se colocan en el paquete. Esto se denomina el *orden de inserción*. Cada elemento de datos contiene un *selector* que identifica el elemento de datos y un *valor* de ese elemento de datos que puede ser un entero, un entero de 64 bits, un filtro de enteros, una serie, un filtro de texto, una serie de bytes, un filtro de serie de bytes, o un manejador de otro paquete. Los elementos de datos se describen en detalle en “Tipos de elemento de datos disponibles en la MQAI” en la página 68

  Existen dos tipos de selector; *selectores de usuario* y *selectores de sistema*. Éstos se describen en la sección Selectores de MQAI. Los selectores normalmente son exclusivos, pero es posible tener varios valores para el mismo selector. En este caso, un *índice* identifica la aparición concreta del selector que

es necesario. Los índices se describen en “Indexación en la interfaz de administración de IBM MQ” en la página 41.

Una jerarquía de estos conceptos se muestra en la Figura 1.

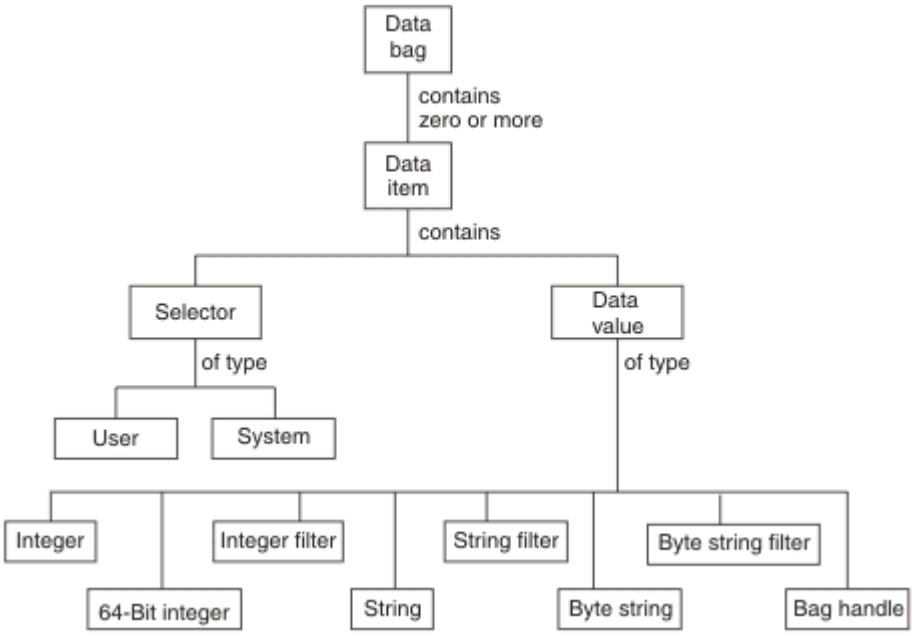


Figura 4. Jerarquía de los conceptos de MQAI

La jerarquía se ha explicado en un párrafo anterior.

Tipos de paquetes de datos

Puede elegir el tipo de paquete de datos que desea crear según la tarea que desee realizar:

paquete de usuario
Un paquete simple que se utiliza para datos de usuario.

paquete de administración
Un paquete creado para los datos utilizados para administrar objetos de IBM MQ enviando mensajes de administración a un servidor de mandatos. El paquete de administración supone automáticamente determinadas opciones tal como se describe en “Creación y supresión de paquetes de datos” en la página 67.

paquete de mandatos
También se crea un paquete para mandatos para administrar objetos de IBM MQ. Sin embargo, a diferencia del paquete de administración, el paquete de mandatos no implica automáticamente determinadas opciones aunque estas opciones están disponibles. Para obtener más información sobre las opciones, consulte “Creación y supresión de paquetes de datos” en la página 67.

paquete de grupo
Un paquete que se utiliza para mantener un conjunto de elementos de datos agrupados. Los paquetes de grupo no pueden utilizarse para administrar objetos de IBM MQ.

Además, el **paquete de sistema** lo crea la MQAI cuando se devuelve un mensaje de respuesta desde el servidor de mandatos y se coloca en el paquete de salida de un usuario. Un paquete de sistema no puede ser modificado por el usuario.

Utilización de paquetes de datos.En este tema se listan las distintas formas de utilizar paquetes de datos:

Utilización de paquetes de datos

En la siguiente lista se muestran las distintas formas de utilizar paquetes de datos:

- Puede crear y suprimir paquetes de datos “Creación y supresión de paquetes de datos” en la página 67.
- Puede enviar datos entre aplicaciones utilizando paquetes de datos “Colocación y recepción de paquetes de datos mediante la MQAI” en la página 68.
- Puede añadir elementos de datos a paquetes de datos “Adición de elementos de datos a paquetes con la MQAI” en la página 69.
- Puede añadir un mandato de consulta dentro de un paquete de datos de “Adición de un mandato de consulta a un paquete” en la página 70.
- Puede consultar dentro de paquetes de datos “Realizar consultas dentro de paquetes de datos” en la página 70.
- Puede contar los elementos de datos dentro de un paquete de datos “Recuento de elementos de datos” en la página 73.
- Puede cambiar información dentro de un paquete de datos “Cambiar información dentro de un paquete” en la página 71.
- Puede borrar un paquete de datos “Borrado de un paquete utilizando la llamada mqClearBag” en la página 72.
- Puede truncar un paquete de datos “Truncamiento de un paquete utilizando la llamada mqTruncateBag” en la página 72.
- Puede convertir paquetes y almacenamientos intermedios “Conversión de paquetes y almacenamientos intermedios” en la página 72.

Creación y supresión de paquetes de datos

Creación de paquetes de datos

Para utilizar la MQAI, primero debe crear un paquete de datos utilizando la llamada mqCreateBag. Como entrada para esta llamada, debe proporcionar una o más opciones para controlar la creación del paquete.

El parámetro **Options** de la llamada MQCreateBag le permite elegir si desea crear un paquete de usuario, un paquete de mandatos, un paquete de grupo o un paquete de administración.

Para crear un paquete de usuario, un paquete de mandatos, o un paquete de grupo, puede elegir una o más opciones adicionales para:

- Utilizar el formato de lista cuando haya dos o más apariciones adyacentes del mismo selector de un paquete.
- Volver a ordenar los elementos de datos cuando se añaden a un mensaje PCF para asegurarse de que los parámetros están en el orden correcto. Para obtener más información sobre los elementos de datos, consulte “Tipos de elemento de datos disponibles en la MQAI” en la página 68.
- Comprobar los valores de selectores de usuario para los elementos que se añaden al paquete.

Los paquetes de administración implican automáticamente estas opciones.

Un paquete de datos se identifica por su manejador. El manejador de paquete se devuelve de mqCreateBag y debe suministrarse en todas las otras llamadas que utilizan el paquete de datos.

Para obtener una descripción completa de la llamada mqCreateBag, consulte mqCreateBag.

Supresión de paquetes de datos

Cualquier paquete de datos creado por el usuario también debe suprimirse utilizando la llamada mqDeleteBag. Por ejemplo, si se crea un paquete en el código de usuario, también debe suprimirse en el código de usuario.

La MQAI crea y suprime automáticamente paquetes de sistema. Para obtener más información, consulte el apartado "Envío de mandatos de administración al servidor de mandatos qm utilizando la llamada mqExecute" en la página 74. El código de usuario no puede suprimir un paquete de sistema.

Para obtener una descripción completa de la llamada mqDeleteBag, consulte mqDeleteBag.

Multi

Colocación y recepción de paquetes de datos mediante la MQAI

Los datos también se pueden enviar entre aplicaciones transfiriendo y obteniendo paquetes de datos utilizando las llamadas mqPutBag y mqGetBag. Esto permite a la Interfaz administrativa de IBM MQ (MQAI) gestionar el almacenamiento intermedio en lugar de la aplicación.

La llamada mqPutBag convierte el contenido del paquete especificado en un mensaje PCF y envía el mensaje a la cola especificada y la llamada mqGetBag elimina el mensaje de la cola especificada y la vuelve a convertir en un paquete de datos. Por lo tanto, la llamada mqPutBag es equivalente a la llamada mqBagToBuffer seguida de MQPUT y la llamada mqGetBag es equivalente a la llamada MQGET seguida de mqBufferToBag.

Para obtener más información sobre el envío y recepción de mensajes PCF en una cola específica, consulte "Envío y recepción de mensajes PCF en una cola especificada" en la página 29

Nota: Si decide utilizar la llamada mqGetBag, los detalles de PCF dentro del mensaje deben ser correctos; si no, se produce el correspondiente error y no se devuelve el mensaje PCF.

Multi

Tipos de elemento de datos disponibles en la MQAI

La Interfaz administrativa de IBM MQ (MQAI) utiliza elementos de datos para llenar los paquetes de datos cuando estos se crean. Estos elementos de datos pueden ser elementos del usuario o del sistema.

Estos elementos del usuario contienen datos de usuario como atributos de objetos que se están administrando. Los elementos del sistema deben utilizarse para un tener mayor control sobre los mensajes generados: por ejemplo, la generación de las cabeceras de mensaje. Para obtener más información sobre elementos del sistema, consulte "Elementos del sistema y MQAI" en la página 69.

Tipos de elementos de datos

Cuando haya creado un paquete de datos, puede rellenarlo con elementos de enteros o elementos de series de caracteres. Puede consultar acerca de los tres tipos de elementos.

El elemento de datos puede ser un elemento de entero o un elemento de serie de caracteres. A continuación se muestran los tipos de elementos de datos disponibles dentro de la MQAI:

- Entero
- Entero de 64 bits
- Filtro de entero
- Serie de caracteres
- Filtro de texto
- Serie de bytes
- Filtro de serie de bytes
- Manejador de paquete

Utilización de elementos de datos

A continuación se muestran las diversas formas de utilización de elementos de datos:

- "Recuento de elementos de datos" en la página 73.
- "Supresión de elementos de datos" en la página 73.

- “Adición de elementos de datos a paquetes con la MQAI” en la página 69.
- “Filtrado y consulta de elementos de datos” en la página 70.

Multi *Elementos del sistema y MQAI*

La Interfaz de administración de IBM MQ (MQAI) puede utilizar elementos del sistema para:

- La generación de cabeceras PCF. Los elementos del sistema puede controlar el identificador de mandatos PCF, las opciones de control, el número de secuencia de mensaje y el tipo de mandato.
- Conversión de datos. Los elementos del sistema manejan el identificador de juego de caracteres para los elementos de serie de caracteres del paquete.

Al igual que todos los elementos de datos, los elementos del sistema constan de un selector y un valor. Para obtener información sobre estos selectores y para lo que sirven, consulte Selectores MQAI.

Los elementos del sistema son exclusivos. Uno o varios elementos del sistema se pueden identificar mediante un selector del sistema. Sólo hay una aparición de cada selector del sistema.

La mayoría de los elementos del sistema pueden modificarse (consulte “Cambiar información dentro de un paquete” en la página 71), pero el usuario no puede modificar las opciones de creación de paquete. No puede suprimir elementos del sistema. (Consulte el apartado “Supresión de elementos de datos” en la página 73).

Multi *Adición de elementos de datos a paquetes con la MQAI*

Cuando se crea un paquete de datos mediante la Interfaz administrativa de IBM MQ (MQAI), puede rellenarlo con elementos de datos. Estos elementos de datos pueden ser elementos del usuario o del sistema.

Para obtener más información sobre los elementos de datos, consulte “Tipos de elemento de datos disponibles en la MQAI” en la página 68.

La MQAI permite añadir elementos de entero, elementos de entero de 64 bits, elementos de filtro de enteros, elementos de serie de caracteres, un filtro de texto, elementos de serie de bytes y elementos de filtro de serie de bytes a paquetes y esto se muestra en la Figura 5 en la página 69. Los elementos se identifican mediante un selector. Normalmente un selector identifica un solo elemento, pero este no siempre es el caso. Si un elemento de datos con el selector especificado ya existe en el paquete, se añade una instancia adicional de ese selector al final del paquete.

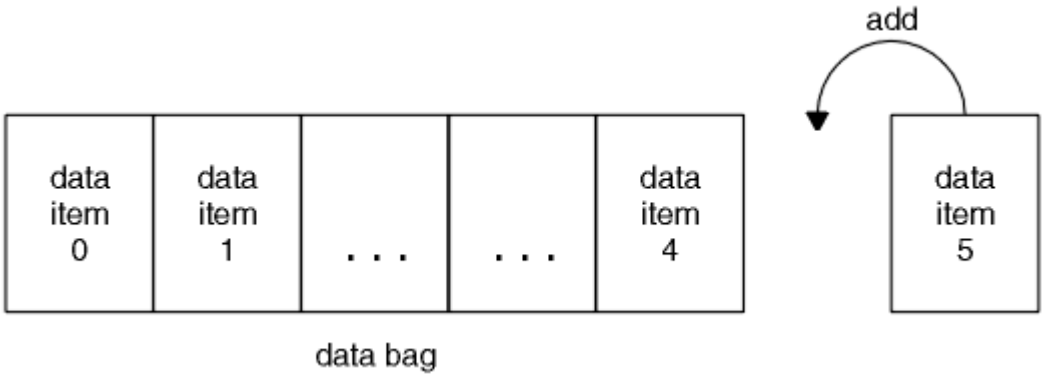


Figura 5. Adición de elementos de datos

Añadir elementos de datos a un paquete utilizando las llamadas mqAdd*:

- Para añadir elementos de entero, utilice la llamada mqAddInteger tal como se describe en mqAddInteger
- Para añadir elementos de entero de 64 bits, utilice la llamada mqAddInteger64 tal como se describe en mqAddInteger64
- Para añadir elementos de filtro de enteros, utilice la llamada mqAddIntegerFilter tal como se describe en mqAddIntegerFilter

- Para añadir elementos de serie de caracteres, utilice la llamada mqAddString tal como se describe en mqAddString
- Para añadir elementos de filtro de texto, utilice la llamada mqAddStringFilter tal como se describe en mqAddStringFilter
- Para añadir elementos de serie de bytes, utilice la llamada mqAddByteString tal como se describe en in mqAddByteString
- Para añadir elementos de filtro de serie de bytes, utilice la llamada mqAddByteStringFilter tal como se describe en mqAddByteStringFilter

Para obtener más información sobre la adición de elementos de datos a un paquete, consulte “Elementos del sistema y MQAI” en la página 69

Adición de un mandato de consulta a un paquete

La llamada mqAddInquiry se utiliza para añadir un mandato de consulta a un paquete. La llamada es específicamente para fines de administración, por lo que sólo puede utilizarse con paquetes de administración. Le permite especificar los selectores de los atributos que desea consultar desde IBM MQ.

Para obtener una descripción completa de la llamada mqAddInquiry, consulte mqAddInquiry.

Filtrado y consulta de elementos de datos

Cuando se utiliza la MQAI para consultar sobre los atributos de objetos de IBM MQ, puede controlar los datos que se devuelven al programa de dos formas.

- Puede ***filtrar*** los datos que se devuelven utilizando las llamadas mqAddInteger y mqAddString. Este enfoque permite especificar un par de *Selector* y *ItemValue*, por ejemplo:

  ```
  mqAddInteger(inputbag, MQIA_Q_TYPE, MQQT_LOCAL)
  ```

  Este ejemplo especifica que el tipo de cola (*Selector*) debe ser local (*ItemValue*) y esta especificación debe coincidir con los atributos del objeto (en este caso, una cola) sobre el que se realiza la consulta.

  Otros atributos que se pueden filtrar corresponden a las mandatos Inquire* de PCF que se pueden encontrar en “Introducción a los Formatos de mandato programable de IBM MQ” en la página 26. Por ejemplo, para consultar sobre los atributos de un canal, consulte el mandato Consultar canal en esta documentación del producto. Los "parámetros obligatorios" y los "parámetros opcionales" del mandato Consultar canal identifican los selectores que puede utilizar para el filtrado.
- Puede ***consultar*** atributos concretos de un objeto utilizando la llamada mqAddInquiry. Esto especifica el selector en los que está interesado. Si no especifica el selector, se devuelven todos los atributos del objeto.

A continuación se muestra un ejemplo del filtro y la consulta de los atributos de una cola:

```
/* Request information about all queues */
mqAddString(adminbag, MQCA_Q_NAME, "*")

/* Filter attributes so that local queues only are returned */
mqAddInteger(adminbag, MQIA_Q_TYPE, MQQT_LOCAL)

/* Query the names and current depths of the local queues */
mqAddInquiry(adminbag, MQCA_Q_NAME)
mqAddInquiry(adminbag, MQIA_CURRENT_Q_DEPTH)

/* Send inquiry to the command server and wait for reply */
mqExecute(MQCMD_INQUIRE_Q, ...)
```

Realizar consultas dentro de paquetes de datos

Puede realizar consultas acerca de:

- El valor de un elemento de entero utilizando la llamada mqInquireInteger. Consulte mqInquireInteger.
- El valor de un elemento de entero de 64 bits utilizando la llamada mqinquireinteger64. Consulte mqInquireInteger64.
- El valor de un elemento de filtro de enteros utilizando la llamada mqInquireIntegerFilter. Consulte mqInquireIntegerFilter.
- El valor de un elemento de serie de caracteres utilizando la llamada mqInquireString. Consulte mqInquireString.
- El valor de un elemento de filtro de texto utilizando la llamada mqInquireStringFilter. Consulte mqInquireStringFilter.
- El valor de un elemento de serie de bytes utilizando la llamada mqInquireByteString. Consulte mqInquireByteString.
- El valor de un elemento de filtro de serie de bytes utilizando la llamada mqInquireByteStringFilter. Consulte mqInquireByteStringFilter.
- El valor de un manejador de paquete utilizando la llamada mqInquireBag. Consulte mqInquireBag.

También puede consultar sobre el tipo (entero, entero de 64 bits, filtro de enteros, serie de caracteres, filtro de texto, serie de bytes, filtro de serie de bytes o manejador de paquete) de un elemento específico utilizando la llamada mqInquireItemInfo. Consulte mqInquireItemInfo.

Multi *Cambiar información dentro de un paquete*

La MQAI permite cambiar información dentro de un paquete utilizando las llamadas mqSet*. Puede:

1. Modificar elementos de datos dentro de un paquete. El índice permite sustituir una instancia individual de un parámetro identificando la aparición del elemento que se debe modificar (consulte Figura 6 en la página 71).

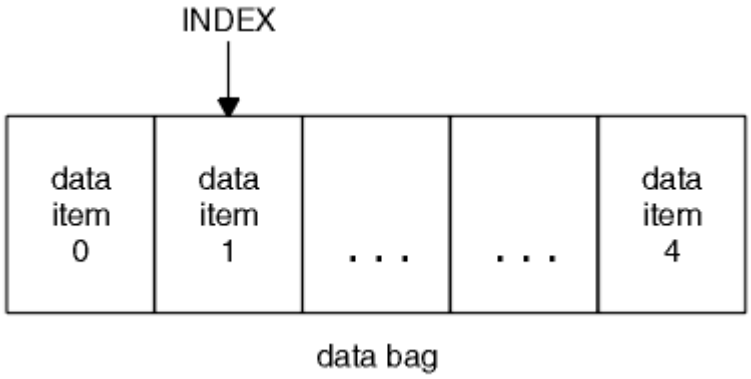


Figura 6. Modificación de un único elemento de datos

2. Suprimir todas las apariciones existentes del selector especificado y añadir una nueva aparición al final del paquete. (Consulte el apartado Figura 7 en la página 71). Un valor de índice especial permite sustituir ***todas*** las instancias de un parámetro.

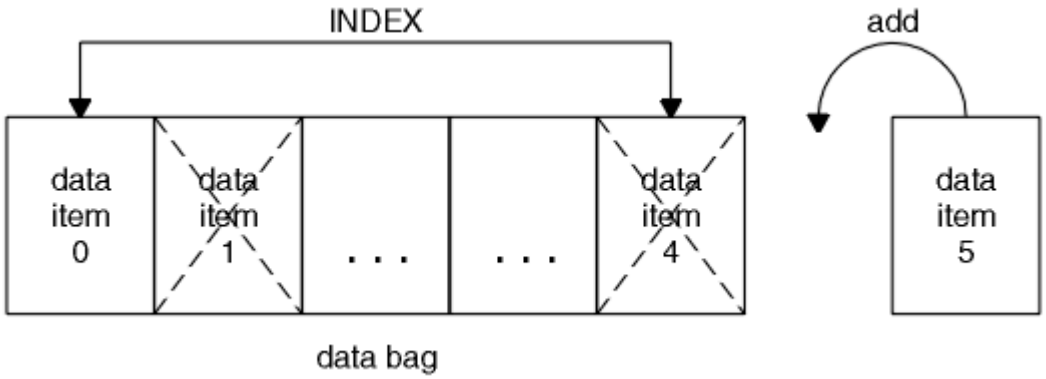


Figura 7. Modificación de todos los elementos de datos

Nota: El índice conserva el orden de inserción en el paquete aunque puede afectar a los índices de otros elementos de datos.

La llamada mqSetInteger permite modificar elementos de entero dentro de un paquete. La llamada mqSetInteger64 permite modificar elementos de enteros de 64 bits. La llamada mqSetIntegerFilter permite modificar elementos de filtro de enteros. La llamada mqSetString permite modificar elementos de serie de caracteres. La llamada mqSetStringFilter permite modificar elementos de filtro de texto. La llamada mqSetByteString permite modificar elementos de serie de bytes. La llamada mqSetByteStringFilter permite modificar elementos de filtro de serie de bytes. Como alternativa, puede utilizar estas llamadas para suprimir todas las apariciones existentes del selector especificado y añadir una nueva aparición al final del paquete. El elemento de datos puede ser un elemento de usuario o un elemento de sistema.

Para obtener una descripción completa de estas llamadas, consulte:

- mqSetInteger
- mqSetInteger64
- mqSetIntegerFilter
- mqSetSerie
- mqSetStringFilter
- mqSetByteString
- mqSetByteStringFiltro

Multi *Borrado de un paquete utilizando la llamada mqClearBag*

La llamada mqClearBag elimina todos los elementos de usuario de un paquete de usuario y restablece los elementos del sistema a sus valores iniciales. También se suprimen los paquetes de sistema incluidos dentro del paquete.

Para obtener una descripción completa de la llamada mqClearBag, consulte mqClearBag.

Multi *Truncamiento de un paquete utilizando la llamada mqTruncateBag*

La llamada mqTruncateBag reduce el número de elementos de usuario de un paquete de usuario suprimiendo los elementos del final del paquete, empezando por el elemento añadido más recientemente. Por ejemplo, puede usarse cuando se utiliza la misma información de cabecera para generar más de un mensaje.

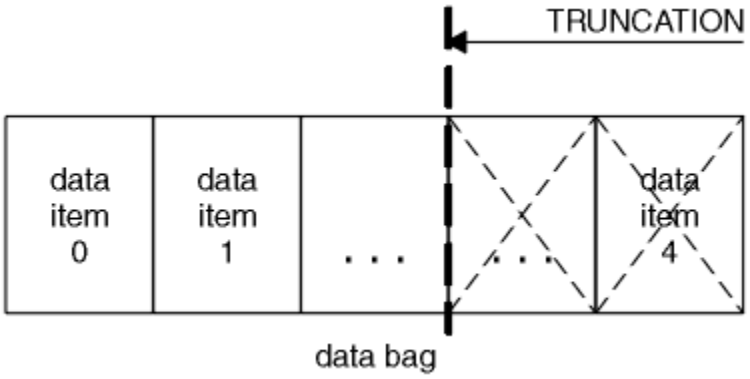


Figura 8. Truncar un paquete

Para obtener una descripción completa de la llamada mqTruncateBag, consulte mqTruncateBag.

Multi *Conversión de paquetes y almacenamientos intermedios*

Para enviar datos entre aplicaciones, primero los datos del mensaje se colocan en un paquete. A continuación, los datos del paquete se convierten en un mensaje PCF utilizando la llamada mqBagToBuffer. El mensaje PCF se envía a la cola necesaria utilizando la llamada MQPUT. Esto se

muestra en la Figura 9 en la página 73. Para obtener una descripción completa de la llamada mqBagToBuffer, consulte mqBagToBuffer.

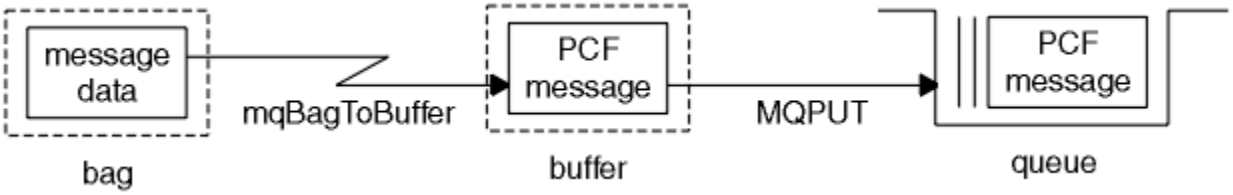


Figura 9. Conversión de paquetes en mensajes PCF

Para recibir datos, el mensaje se recibe en un almacenamiento intermedio utilizando la llamada MQGET. Los datos del almacenamiento intermedio a continuación se convierten en un paquete utilizando la llamada mqBufferToBag, siempre que el almacenamiento intermedio contenga un mensaje PCF válido. Esto se muestra en la Figura 10 en la página 73. Para obtener una descripción completa de la llamada mqBufferToBag, consulte mqBufferToBag.

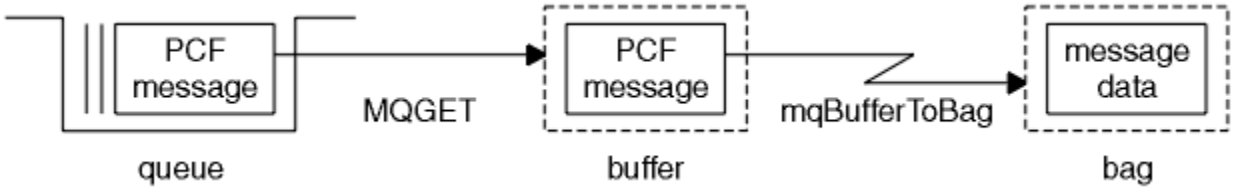


Figura 10. Conversión de mensajes PCF a formato de paquete

Multi *Recuento de elementos de datos*

La llamada mqCountItems cuenta el número de elementos de usuario, elementos del sistema, o ambos, que están almacenados en un paquete de datos, y devuelve este número. Por ejemplo, `mqCountItems( Bag, 7, ...)`, devuelve el número de elementos del paquete con un selector de 7. Puede contar elementos por selector individual, por selectores de usuario, por selectores del sistema o por todos los selectores.

Nota: Esta llamada cuenta el número de elementos de datos, no el número de selectores exclusivos del paquete. Un selector puede darse varias veces, por lo que puede haber menos selectores exclusivos en el paquete que elementos de datos.

Para obtener una descripción completa de la llamada mqCountItems, consulte mqCountItems.

Multi *Supresión de elementos de datos*

Puede suprimir elementos de paquetes de varias formas. Puede:

- Eliminar uno o más elementos de usuario de un paquete. Para obtener información detallada, consulte “Supresión de elementos de datos de un paquete utilizando la llamada mqDeleteItem” en la página 73.
- Suprimir todos los elementos de usuario de un paquete, es decir, borrar un paquete. Para obtener información detallada, consulte “Borrado de un paquete utilizando la llamada mqClearBag” en la página 72.
- Suprimir todos los elementos de usuario del final de un paquete, es decir, truncar un paquete. Para obtener información detallada, consulte “Truncamiento de un paquete utilizando la llamada mqTruncateBag” en la página 72.

Multi *Supresión de elementos de datos de un paquete utilizando la llamada mqDeleteItem*

La llamada mqDeleteItem elimina uno o más elementos de usuario de un paquete. El índice se utiliza para suprimir uno de los siguientes valores:

1. Una única aparición del selector especificado. (Consulte Figura 11 en la página 74.)

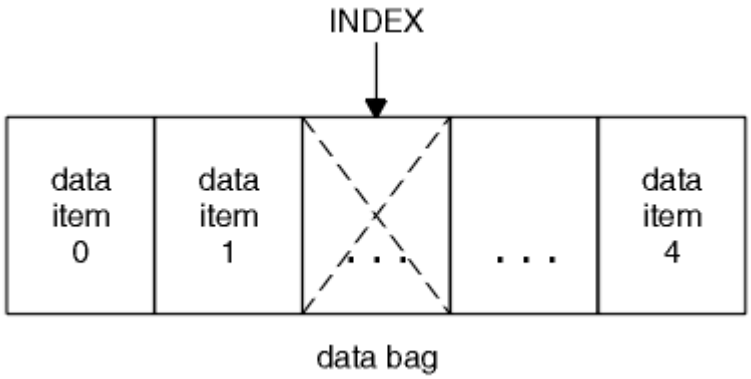


Figura 11. Supresión de un único elemento de datos

o

2. Todas las apariciones del selector especificado. (Consulte Figura 12 en la página 74.)

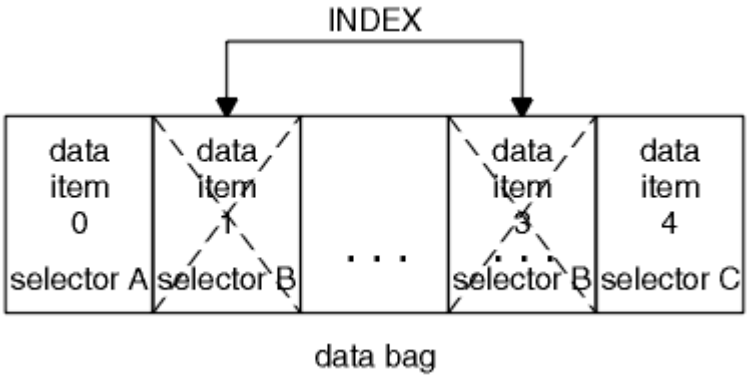


Figura 12. Supresión de todos los elementos de datos

Nota: El índice conserva el orden de inserción en el paquete aunque puede afectar a los índices de otros elementos de datos. Por ejemplo, la llamada mqDeleteItem no conserva los valores de índice de los elementos de datos que siguen el elemento suprimido porque los índices se reorganizan para rellenar el hueco que permanece del elemento suprimido.

Para obtener una descripción completa de la llamada mqDeleteItem, consulte mqDeleteItem.

Multi Envío de mandatos de administración al servidor de mandatos qm utilizando la llamada mqExecute

Una vez que se ha creado y llenado un paquete de datos, puede enviarse un mensaje de mandato administrativo al servidor de mandatos de un gestor de colas utilizando la llamada mqExecute. Esto maneja el intercambio con el servidor de mandatos y devuelve respuestas en un paquete.

Después de haber creado y llenado los paquetes de datos, puede enviar un mensaje de mandato de administración al servidor de mandatos de un gestor de colas. La forma más fácil de hacerlo es utilizando la llamada mqExecute. La llamada mqExecute envía un mensaje de mandato de administración como mensaje no persistente y espera las respuestas. Las respuestas se devuelven en un paquete de respuesta. Por ejemplo, estas pueden incluir información sobre los atributos relacionados con varios objetos de IBM MQ o una serie de mensajes de respuesta de error PCF. Por lo tanto, el paquete de respuesta podría contener un código de retorno o sólo *paquetes anidados*.

Los mensajes de respuesta se colocan en paquetes del sistema que crea el sistema. Por ejemplo, para consultas sobre los nombres de los objetos, se crea un paquete de sistema para mantener esos nombres de objeto y el paquete se inserta en el paquete de usuario. Los manejadores de estos paquetes se insertarán en el paquete de respuesta y el selector MQHA_BAG_HANDLE podrá acceder al paquete anidado. El paquete de sistema permanece en el almacenamiento, si no se suprime, hasta que se suprime el paquete de respuestas.

El concepto de *anidamiento* es muestra en la Figura 13 en la página 75.

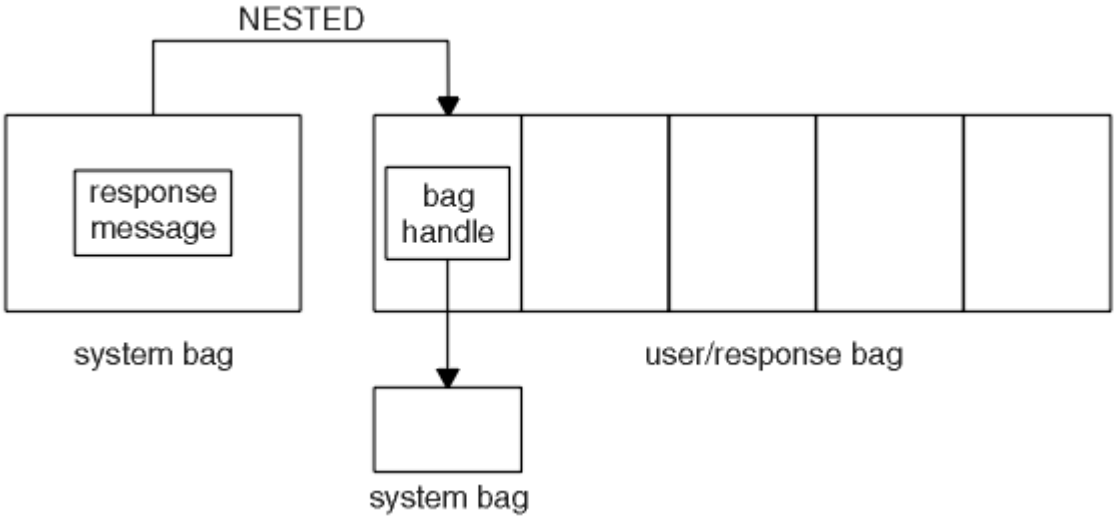


Figura 13. Anidamiento

Como entrada para la llamada mqExecute debe proporcionar:

- Un manejador de conexión MQI.
- El mandato que se va a ejecutar. Este debe ser uno de los valores MQCMD_*.

  Nota: Si la MQAI no reconoce este valor, el valor se sigue aceptando. Sin embargo, si se utilizó la llamada mqAddInquiry para insertar valores en el paquete, este parámetro debe ser un mandato INQUIRE reconocido por la MQAI. Es decir, el parámetro debe tener el formato MQCMD_INQUIRE_*.
- Opcionalmente, un manejador del paquete que contiene las opciones que controlan el proceso de la llamada. Aquí también es donde puede especificar el tiempo máximo en milisegundos que la MQAI debe esperar para cada mensaje de respuesta.
- Un manejador del paquete de administración que contiene detalles del mandato de administración que va a emitirse.
- Un manejador del mensaje de respuesta que recibe los mensajes de respuesta.

Los siguientes manejadores son opcionales:

- Un manejador de objeto de la cola donde se colocará el mandato de administración.

  Si no se especifica ningún manejador de objeto, el mandato de administración se coloca en la cola SYSTEM.ADMIN.COMMAND.QUEUE que pertenece al gestor de colas conectado actualmente. Este es el valor predeterminado.
- Un manejador de objeto de la cola donde se colocarán los mensajes de respuesta.

  Puede elegir colocar los mensajes de respuesta en una cola dinámica que la MQAI crea automáticamente. La cola creada sólo existe durante la llamada y la MQAI la suprime al salir de la llamada mqExecute.

Para obtener ejemplos de usos de la llamada mqExecute, consulte el apartado Código de ejemplo

Administración utilizando REST API

Puede utilizar la administrative REST API para administrar objetos IBM MQ como, por ejemplo, gestores de colas y colas, y agentes y transferencias Managed File Transfer. La información se envía a y se recibe de administrative REST API en formato JSON. Estas API RESTful pueden ayudarle a incorporar la administración de IBM MQ en las herramientas populares de automatización y DevOps.

Antes de empezar

Nota: V 9.4.0 administrative REST API no está disponible en una instalación autónoma de IBM MQ Web Server . Para obtener más información sobre las opciones de instalación para el componente IBM MQ que ejecuta administrative REST API, consulte IBM MQ Console y REST API.

Para ver la información de referencia sobre los recursos REST disponibles, consulte Referencia de la administrative REST API.

Procedimiento

Iniciación a administrative REST API

Empiece rápidamente con la administrative REST API y pruebe unas pocas solicitudes de ejemplo utilizando cURL para crear, actualizar, ver y suprimir una cola.

Antes de empezar

Para empezar a utilizar la administrative REST API, los ejemplos de esta tarea tienen los requisitos siguientes:

- Los ejemplos utilizan cURL para realizar solicitudes REST para mostrar información sobre gestores de colas en el sistema, y para crear una cola, actualizar, ver y suprimir una cola. Por lo tanto, para completar estar, necesita tener el cURL instalado en el sistema.
- Para completar esta tarea, debe ser un usuario con determinados privilegios para que pueda utilizar el mandato **dspmqweb**:
  - z/OS En z/OS, debe tener autorización para ejecutar el mandato **dspmqweb** y acceso de escritura en el archivo `mqwebuser.xml`.
  - Multi En todos los demás sistemas operativos, debe ser un usuario con privilegios.

IBM i En IBM i, los mandatos deben estar en ejecución en QSHELL.

Procedimiento

1. Asegúrese de que ha configurado el servidor mqweb para que lo utilice administrative REST API, administrative REST API para MFT, messaging REST APIo IBM MQ Console.

   Para obtener más información sobre cómo configurar el servidor mqweb con un registro básico, consulte Configuración básica para el servidor mqweb.

2. z/OS

   En z/OS, establezca la variable de entorno WLP_USER_DIR para que pueda utilizar el mandato **dspmqweb** . Establezca la variable para que apunte a la configuración del servidor mqweb especificando el mandato siguiente.

   ```
   export WLP_USER_DIR=WLP_user_directory
   ```

donde *WLP_user_directory* es el nombre del directorio que se pasa a `crtmqweb`. Por ejemplo:

```
export WLP_USER_DIR=/var/mqm/web/installation1
```

Si desea más información, consulte Creación del servidor mqweb.

3. Determina elREST API URL ingresando el siguiente comando:

```
dspmqweb status
```

Los ejemplos de los siguientes pasos suponen que suREST API La URL es la URL predeterminada.`https://localhost:9443/ibmmq/rest/v1/` . Si el URL es diferente al valor predeterminado, sustituya el URL en los pasos siguientes.

4. Pruebe una solicitud GET en el recurso `qmgr` utilizando autenticación básica con el usuario `mqadmin`:

```
curl -k https://localhost:9443/ibmmq/rest/v2/admin/qmgr -X GET -u mqadmin:mqadmin
```

5. Cree, vea, modifique y suprima una cola utilizando el recurso `mqsc`:

Este ejemplo utiliza un gestor de colas `QM1`. Cree un gestor de colas con el mismo nombre o sustituya un gestor de colas existentes en el sistema.

a) Realice una solicitud POST en el recurso `mqsc` para crear la cola local:

En el cuerpo de la solicitud, el nombre de la nueva cola se establece en `Q1`. Se utiliza la autenticación básica y una cabecera HTTP `ibm-mq-rest-csrf-token` con un valor arbitrario se establece en la solicitud REST cURL. Esta cabecera adicional es necesaria para solicitudes POST, PATCH y DELETE:

```
curl -k https://localhost:9443/ibmmq/rest/v2/admin/action/qmgr/QM1/mqsc -X POST -u
mqadmin:mqadmin -H "ibm-mq-rest-csrf-token: value" -H "Content-Type: application/json" --
data "{\"type\": \"runCommandJSON\", \"command\": \"define\", \"qualifier\": \"qlocal\",
\"name\": \"Q1\"}"
```

b) Realice una solicitud POST en el recurso `mqsc` para ver la cola local creada en el paso “5.a” en la página 77:

```
curl -k https://localhost:9443/ibmmq/rest/v2/admin/action/qmgr/QM1/mqsc -X POST -u
mqadmin:mqadmin -H "ibm-mq-rest-csrf-token: value" -H "Content-Type: application/json" --
data "{\"type\": \"runCommandJSON\", \"command\": \"display\", \"qualifier\": \"qlocal\",
\"name\": \"Q1\"}"
```

c) Realice una solicitud POST en el recurso `mqsc` al recurso para actualizar la descripción de la cola:

```
curl -k https://localhost:9443/ibmmq/rest/v2/admin/action/qmgr/QM1/mqsc -X POST -u
mqadmin:mqadmin -H "ibm-mq-rest-csrf-token: value" -H "Content-Type: application/json" --
data "{\"type\": \"runCommandJSON\", \"command\": \"alter\", \"qualifier\": \"qlocal\",
\"name\": \"Q1\", \"parameters\": {\"descr\": \"new description\" }}"
```

d) Realice una solicitud POST en el recurso `mqsc` para ver la nueva descripción de cola. Especifique el atributo **`responseParameters`** en el cuerpo de la solicitud de modo que la respuesta incluya el campo de descripción:

```
curl -k https://localhost:9443/ibmmq/rest/v2/admin/action/qmgr/QM1/mqsc -X POST -u
mqadmin:mqadmin -H "ibm-mq-rest-csrf-token: value" -H "Content-Type: application/json" --
data "{\"type\": \"runCommandJSON\", \"command\": \"display\", \"qualifier\": \"qlocal\",
\"name\": \"Q1\", \"responseParameters\" : [\"descr\"]}"
```

e) Realice una solicitud POST en el recurso `mqsc` para suprimir la cola:

```
curl -k https://localhost:9443/ibmmq/rest/v2/admin/action/qmgr/QM1/mqsc -X POST -u
mqadmin:mqadmin -H "ibm-mq-rest-csrf-token: value" -H "Content-Type: application/json" --
data "{\"type\": \"runCommandJSON\", \"command\": \"delete\", \"qualifier\": \"qlocal\",
\"name\": \"Q1\"}"
```

f) Realice una solicitud POST en el recurso `mqsc` para probar que la cola se ha suprimido:

```
curl -k https://localhost:9443/ibmmq/rest/v2/admin/action/qmgr/QM1/mqsc -X POST -u
mqadmin:mqadmin -H "ibm-mq-rest-csrf-token: value" -H "Content-Type: application/json" --
data "{\"type\": \"runCommandJSON\", \"command\": \"display\", \"qualifier\": \"qlocal\",
\"name\": \"Q1\"}"
```

Qué hacer a continuación

- Los ejemplos utilizan la autenticación básica para proteger la solicitud. En su lugar, puede utilizar la autenticación basada en señal o la autenticación basada en cliente. Si desea más información, consulte Utilización de autenticación de certificado de cliente con la REST API y IBM MQ Console, y Utilización de la autenticación basada en señal con la REST API.
- Obtenga más información sobre cómo utilizar administrative REST API y construir los URL con parámetros de consulta: “Utilización de administrative REST API” en la página 79.
- Examine la información de referencia para los recursos de administrative REST API disponibles y todos los parámetros de consulta opcionales disponibles: Referencia de administrative REST API.
- Obtenga información sobre cómo utilizar la administrative REST API para administrar objetos IBM MQ en sistemas remotos: “Administración remota mediante REST API” en la página 81.
- Obtenga más información sobre cómo utilizar la administrative REST API con MFT:“Cómo empezar con REST API para MFT” en la página 78.
- Descubra messaging REST API, una interfaz RESTful para la mensajería IBM MQ: Mensajería utilizando REST API.
- Descubra IBM MQ Console, una GUI basada en navegador: “Administración utilizando IBM MQ Console” en la página 93.

Cómo empezar con REST API para MFT

Empiece rápidamente con administrative REST API para Managed File Transfer y pruebe algunas solicitudes de ejemplo para ver el estado del agente MFT y para ver una lista de transferencias.

Antes de empezar

- Los ejemplos utilizan cURL para enviar solicitudes REST para ver una lista de transferencias y ver el estado del agente MFT. Por lo tanto, para completar estar, necesita tener el cURL instalado en el sistema.
- Para completar esta tarea, debe ser un usuario con determinados privilegios para que pueda utilizar el mandato **dspmqweb**:
  - z/OS En z/OS, debe tener autorización para ejecutar el mandato **dspmqweb** y acceso de escritura en el archivo `mqwebuser.xml`.
  - Multi En todos los demás sistemas operativos, debe ser un usuario con privilegios.

Procedimiento

1. Asegúrese de que el servidor mqweb se ha configurado para administrative REST API para MFT:
   - Asegúrese de que ha configurado el servidor mqweb para que lo utilice administrative REST API, administrative REST API para MFT, messaging REST APIo IBM MQ Console.Para obtener más información sobre cómo configurar el servidor mqweb con un registro básico, consulte Configuración básica para el servidor mqweb.
   - Si el servidor mqweb está configurado, asegúrese de que el paso 8 de la Configuración básica para el servidor mqweb se ha completado para habilitar administrative REST API para MFT.
2. z/OS

En z/OS, establezca la variable de entorno WLP_USER_DIR para que pueda utilizar el mandato **dspmqweb** . Establezca la variable para que apunte a la configuración del servidor mqweb especificando el mandato siguiente.

```
export WLP_USER_DIR=WLP_user_directory
```

donde *WLP_user_directory* es el nombre del directorio que se pasa a `crtmqweb`. Por ejemplo:

```
export WLP_USER_DIR=/var/mqm/web/installation1
```

Si desea más información, consulte Creación del servidor mqweb.

3. Determina elREST API URL ingresando el siguiente comando:

```
dspmqweb status
```

Los ejemplos de los siguientes pasos suponen que suREST API La URL es la URL predeterminada.`https://localhost:9443/ibmmq/rest/v1/` . Si el URL es diferente al valor predeterminado, sustituya el URL en los pasos siguientes.

4. Realice una solicitud GET en el recurso `agent` para volver a detalles básicos sobre todos los agentes, incluyendo el nombre, tipo y estado:

```
curl -k https://localhost:9443/ibmmq/rest/v2/admin/mft/agent/ -X GET -u mftadmin:mftadmin
```

5. Cree algunas transferencias para mostrar utilizando el mandato **`fteCreateTransfer`**.

El servidor mqweb almacena en la memoria caché información sobre transferencias y devuelve esta información cuando se realiza una solicitud. Esta memoria caché se restablece cuando se reinicia el servidor mqweb. Puede ver si el servidor se ha reiniciado visualizando los archivos `console.log` y `messages.log`, o en z/OS, buscando la salida de la tarea iniciada.

6. Realice una solicitud GET en el recurso `transfer` para devolver detalles de hasta cuatro transferencias que se han realizado desde que se inició el servidor mqweb:

```
curl -k https://localhost:9443/ibmmq/rest/v2/admin/mft/transfer?limit=4 -X GET -u
mftadmin:mftadmin
```

Qué hacer a continuación

- Los ejemplos utilizan la autenticación básica para proteger la solicitud. En su lugar, puede utilizar la autenticación basada en señal o la autenticación basada en cliente. Si desea más información, consulte Utilización de la autenticación basada en señal con el REST API y Utilización de la autenticación de certificado de cliente con REST API y IBM MQ Console.
- Obtenga más información sobre cómo utilizar administrative REST API y construir los URL con parámetros de consulta: “Utilización de administrative REST API” en la página 79.
- Examine la información de referencia para los recursos de administrative REST API para MFT disponibles y todos los parámetros de consulta opcionales disponibles: Referencia de administrative REST API.
- Descubra messaging REST API, una interfaz RESTful para la mensajería IBM MQ: Mensajería utilizando REST API.
- Descubra IBM MQ Console, una GUI basada en navegador: “Administración utilizando IBM MQ Console” en la página 93.

Utilización de administrative REST API

Cuando utiliza administrative REST API, se invocan métodos HTTP en los URL que representan los diversos objetos de IBM MQ, como gestores de colas o colas. El método HTTP, por ejemplo, POST representa el tipo de acción que debe realizarse en el objeto que está representado por el URL. Es posible que se proporcione más información acerca de la acción en JSON como parte de la carga útil del método

HTTP o codificada en los parámetros de consulta. La información sobre el resultado de realizar la acción puede devolverse como el cuerpo de la respuesta HTTP.

Antes de empezar

Tenga en cuenta estos puntos antes de utilizar administrative REST API:

- Debe autenticarse con el servidor mqweb para poder utilizar administrative REST API. Puede autenticarse mediante la autenticación básica HTTP, la autenticación de certificados de cliente o la autenticación basada en señal. Para obtener más información sobre cómo utilizar estos métodos de autenticación, consulte Seguridad deIBM MQ Console y REST API.
- REST API es sensible a las mayúsculas y minúsculas. Por ejemplo un HTTP GET sobre el URL siguiente no muestra información si el gestor de colas se llama `qmgr1`.

```
/ibmmq/rest/v1/admin/qmgr/QMGR1
```

- No todos los caracteres que se pueden utilizar en nombres de objeto de IBM MQ se pueden codificar directamente en un URL. Para codificar estos caracteres correctamente, debe utilizar la codificación de URL apropiada:
  - Una barra inclinada, /, debe codificarse como %2F.
  - Un signo de porcentaje, %, debe codificarse como %25.
- Debido al comportamiento de algunos navegadores, no nombre objetos utilizando solo un punto o una barra inclinada.

Acerca de esta tarea

Cuando se utiliza REST API para realizar una acción en un objeto, primero es necesario construir un URL para representar ese objeto. Cada URL empieza con un prefijo, que describe a qué nombre de host y puerto se debe enviar la solicitud. El resto del URL describe un objeto particular o un conjunto de objetos conocido como un recurso.

La acción que se debe realizar en el recurso define si el URL necesita parámetros de consulta o no. También define el método HTTP que se utiliza y si se envía información adicional al URL, o se devuelve del mismo, en formato JSON. La información adicional puede formar parte de la solicitud HTTP o devolverse como parte de la respuesta HTTP.

Después de construir el URL y de crear una carga útil JSON opcional para enviar en la solicitud HTTP, puede enviar la solicitud HTTP a IBM MQ. Puede enviar la solicitud utilizando la implementación de HTTP que se base en el lenguaje de programación de su elección. También puede enviar las solicitudes utilizando herramientas de línea de mandatos como cURL, o un navegador web o complemento de navegador web.

Importante: Como mínimo, hay que seguir los pasos “1.a” en la página 80 y “1.b” en la página 80.

Procedimiento

1. Construya el URL:
   a) Determine el URL del prefijo especificando el mandato siguiente:

      ```
      dspmqweb status
      ```

      El URL que desea utilizar incluye la frase `/ibmmq/rest/`.
   b) Añada el recurso a la vía de acceso de URL.

      Están disponibles los recursos IBM MQ:

      - /admin/instalación
      - /admin/gestor_colas
      - /admin/cola

- /admin/suscripción
- /admin/canal
- /action/qmgr/{qmgrname}/mqsc

Están disponibles los recursos Managed File Transfer:

- /admin/agente
- /admin/transferencia
- /admin/monitor

Por ejemplo, para interactuar con gestores de colas, añada `/qmgr` al URL de prefijo para crear el URL siguiente:

```
https://localhost:9443/ibmmq/rest/v2/admin/qmgr
```

c) Opcional: Añada los segmentos de vía de acceso opcionales adicionales al URL.

En la información de referencia para cada tipo de objeto, los segmentos opcionales pueden identificarse en el URL por las llaves que los incluyen { }.

Por ejemplo, añada el nombre de gestor de colas `QM1` al URL para crear el URL siguiente:

```
https://localhost:9443/ibmmq/rest/v2/admin/qmgr/QM1
```

d) Opcional: Añada un parámetro de consulta opcional al URL.

Añada un signo de interrogación,?, nombre de variable, signo igual =, y un valor o lista de valores para el URL.

Por ejemplo, para solicitar todos los atributos del gestor de colas QM1, cree el URL siguiente:

```
https://localhost:9443/ibmmq/rest/v2/admin/qmgr/QM1?attributes=*
```

e) Añada parámetros de consulta opcionales adicionales al URL.

Añada un ampersand, &, al URL y, a continuación, repita el paso d.

2. Invoque el método HTTP correspondiente en el URL. Especifique cualquier carga útil JSON opcional y proporcione las credenciales de seguridad apropiadas para la autenticación. Por ejemplo:
   - Utilice la implementación de HTTP/REST del lenguaje de programación elegido.
   - Utilice una herramienta como un complemento de navegador de cliente REST o cURL.

Administración remota mediante REST API

Puede utilizar REST API para administrar los gestores de colas remotos y los objetos de IBM MQ asociados con dichos gestores de colas. Esta administración remota incluye los gestores de colas que están en el mismo sistema, pero no en la misma instalación IBM MQ que el servidor mqweb. Por lo tanto, puede utilizar REST API para administrar toda la red de IBM MQ sólo con una única instalación que ejecuta el servidor mqweb. Para administrar los gestores de colas remotos, debe configurar la pasarela de administrative REST API de manera que al menos un gestor de colas de la misma instalación que el servidor mqweb actúe como un gestor de colas de pasarela. Por lo tanto, puede especificar el gestor de colas remoto en el URL de recursos de REST API para realizar la acción administrativa especificada.

Antes de empezar

Puede impedir la administración remota inhabilitando la pasarela de administrative REST API. Para obtener más información, consulte Configuración de la pasarela de administrative REST API.

Para utilizar la pasarela de administrative REST API, se deben cumplir las condiciones siguientes:

- El servidor mqweb tiene que estar configurado e iniciado. Para obtener más información sobre cómo configurar e iniciar el servidor mqweb, consulte “Iniciación a administrative REST API” en la página 76.

- El gestor de colas que desee configurar como gestor de colas de pasarela tiene que estar en la misma instalación que el servidor mqweb.
- El gestor de colas remoto que desee administrar tiene que ser IBM MQ 8.0 o posteriores.
- Hay que asegurarse de que todos los atributos especificados en la solicitud sean válidos para el sistema al que se envía dicha solicitud. Por ejemplo, si el gestor de colas de pasarela está en Windows y el gestor de colas remoto está en z/OS, no puede solicitar que se devuelva el atributo `dataCollection.statistics` en una solicitud HTTP GET en el recurso `queue`.
- Hay que asegurarse de que todos los atributos especificados en la solicitud sean válidos para el nivel de IBM MQ al que se envía dicha solicitud. Por ejemplo, si el gestor de colas remoto está ejecutando en IBM MQ 8.0, no puede solicitar que se devuelva el atributo `extended.enableMediaImageOperations` en una solicitud HTTP GET en el recurso `queue`.
- Hay que utilizar uno de estos recursos REST soportados:
  - `/queue`
  - `/subscription`
  - `/channel`
  - `/mqsc`
  - `/qmgr`

    El recurso `/qmgr` solo devuelve un subconjunto de los atributos cuando se consulta un gestor de colas remoto: `name`, `status.started`, `status.channelInitiatorState`, `status.ldapConnectionState`, `status.connectionCount` y `status.publishSubscribeState`.

Acerca de esta tarea

Para utilizar la pasarela de administrative REST API para administrar los gestores de colas remotos, debe preparar los gestores de colas para la administración remota. Es decir, hay que configurar colas de transmisión, escuchas y canales emisor y receptor entre el gestor de colas de pasarela y el gestor de colas remoto. A continuación, se puede enviar una solicitud REST al gestor de colas remoto especificando el gestor de colas en el URL de recurso. EL gestor de colas de pasarela se especifica utilizando el mandato **`setmqweb`** para establecer el atributo `mqRestGatewayQmgr` en el nombre del gestor de colas de pasarela, o enviando el nombre del gestor de colas de pasarela en una cabecera que se envía con la solicitud. La solicitud se envía a través del gestor de colas de pasarela del gestor de colas remoto. La respuesta se devuelve en una cabecera que indica el gestor de colas usado como gestor de colas de pasarela.

Procedimiento

1. Configure las comunicaciones entre el gestor de colas de pasarela y los gestores de colas remotos que desee administrar. Estos pasos de configuración son los mismos pasos que son necesarios para configurar la administración remota con runmqsc y PCF.

   Para obtener más información sobre estos pasos, consulte "Configuración de gestores de colas para la administración remota" en la página 201.
2. Configure la seguridad en los gestores de colas remotos:

   a) Asegúrese de que existan los correspondientes ID de usuario en el sistema en que ejecuta el gestor de colas remoto. El ID de usuario debe existir en el sistema remoto que depende del rol del usuario de REST API:

   - Si el usuario de REST API está en el grupo MQWebAdmin o MQWebAdminRO, el ID de usuario que ha iniciado el servidor mqweb debe existir en el sistema remoto. En IBM MQ Appliance, el usuario que inicia el servidor mqweb es `mqsystem`.
   - Si el usuario de REST API está en el grupo MQWebUser, dicho ID de usuario de REST API debe existir en el sistema remoto.

b) Asegúrese de que se otorguen a los ID de usuario correspondientes los niveles de autorización necesarios para acceder a los recursos de REST API adecuados en el gestor de colas remoto:

- Autoridad para poner mensajes en `SYSTEM.ADMIN.COMMAND.QUEUE`.
- Autoridad para poner mensajes en `SYSTEM.REST.REPLY.QUEUE`.
- Autoridad para acceder a las colas de transmisión definidas para la administración remota.
- Autoridad para visualizar los atributos del gestor de colas.
- Autoridad para realizar peticiones REST. Para obtener más información, consulte la sección de requisitos de seguridad de los temas de referencia de los recursos de REST API.

3. Configure qué gestor de colas local se utiliza como pasarela. Se puede configurar un gestor de colas de pasarela predeterminado, especificar el gestor de colas de pasarela en una cabecera HTTP o utilizar una combinación de ambos enfoques:
   - Configurar un gestor de colas de pasarela predeterminado con el mandato **setmqweb**:

     ```
     setmqweb properties -k mqRestGatewayQmgr -v qmgrName
     ```

     donde *qmgrName* es el nombre del gestor de colas de pasarela.

     Este gestor de colas de pasarela se utiliza cuando se cumplen las dos sentencias siguientes:

     – No se ha especificado un gestor de colas en la cabecera `ibm-mq-rest-gateway-qmgr` de una solicitud REST.
     – El gestor de colas especificado en el URL del recurso de REST API no es un gestor de colas local.
   - Configure el gestor de colas de pasarela en cada solicitud REST estableciendo la cabecera HTTP `ibm-mq-rest-gateway-qmgr` al nombre del gestor de colas de pasarela.
4. Incluya en el URL del recurso el nombre del gestor de colas remoto que desee administrar.

   Por ejemplo, para obtener la lista de colas del gestor de colas remoto `remoteQM`, utilice el URL siguiente:

   ```
   https://localhost:9443/ibmmq/rest/v1/admin/qmgr/remoteQM/queue
   ```

Resultados

Se devolverá una cabecera `ibm-mq-rest-gateway-qmgr` con la respuesta REST. Esta cabecera especifica qué gestor de colas se utiliza como pasarela.

Si tiene dificultades con el uso de administrative REST API para administrar gestores de colas remotos.

- Compruebe que se está ejecutando el gestor de colas remoto.
- Compruebe que el servidor de mandatos se está ejecutando en el sistema remoto.
- Compruebe que el intervalo de desconexión del canal no ha caducado. Por ejemplo, si se ha iniciado un canal, pero se ha concluido al cabo de un tiempo. Esto es especialmente importante si inicia manualmente los canales.

Ejemplo

En el ejemplo siguiente, hay tres instalaciones de IBM MQ en dos máquinas. En `Machine 1`, hay un `Installation 1` y un `Installation 2`. En `Machine 2`, hay un `Installation 3`. Se ha configurado un servidor mqweb para `Installation 1`. Hay un único gestor de colas en cada instalación y dichos gestores de colas están configurados para la administración remota. Es decir, se configuran e inician los siguientes escuchas, canales y colas:

- En el gestor de colas QM1, en `Installation 1`, en `Machine 1`:
  – Canal emisor QM1.to.QM2
  – Canal receptor QM2.to.QM1
  – Canal emisor QM1.to.QM3

  - Canal receptor QM3.to.QM1
  - Cola de transmisión QM2
  - Cola de transmisión QM3
  - Un escucha configurado en el puerto 1414
- En el gestor de colas QM2, en `Installation 2`, en `Machine 1`:
  - Canal emisor QM2.to.QM1
  - Canal receptor QM1.to.QM2
  - Cola de transmisión QM1
  - Un escucha configurado en el puerto 1415
- En el gestor de colas QM3, en `Installation 3`, en `Machine 2`:
  - Canal emisor QM3.to.QM1
  - Canal receptor QM1.to.QM3
  - Cola de transmisión QM1
  - El escucha predeterminado

En QM2 se define la cola `Qon2` y en QM3 se define la cola `Qon3`.

Al usuario `mquser`, definido en ambas máquinas, se le otorga el rol MQWebAdmin en REST API y la autoridad de acceder a las correspondientes colas en cada gestor de colas.

El mandato setmqweb se utiliza para configurar el gestor de colas QM1 como gestor de colas de pasarela predeterminado.

El diagrama siguiente muestra esta configuración:

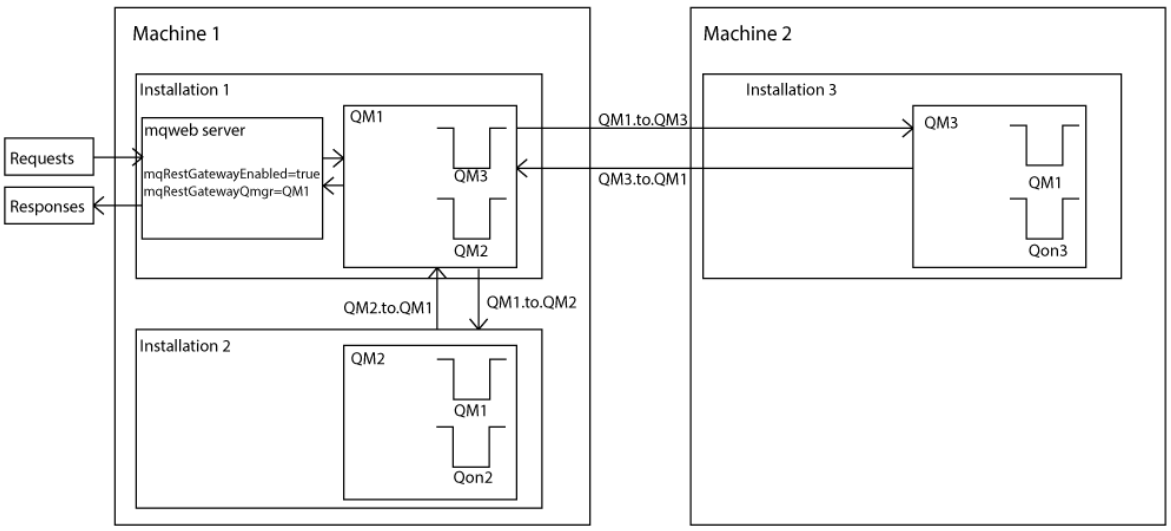


Figura 14. Diagrama de configuración de ejemplo de una administración remota mediante REST API.

Se envía la siguiente solicitud REST al servidor mqweb:

```
GET https://localhost:9443/ibmmq/rest/v1/admin/qmgr/QM2/queue?
attributes=general.isTransmissionQueue
```

Se recibe la siguiente respuesta:

```
{
   "queue" :
   [{
```

```
        "general": {
            "isTransmissionQueue": true
        },
        "name": "QM1",
        "type": "local"
    },
   {
        "general": {
            "isTransmissionQueue": false
        },
        "name" : "Qon2",
        "type" : "local"
    }]
}
```

Se envía la siguiente solicitud REST al servidor mqweb:

```
GET https://localhost:9443/ibmmq/rest/v1/admin/qmgr/QM3/queue?
attributes=general.isTransmissionQueue,general.description
```

Se recibe la siguiente respuesta:

```
{
   "queue" :
   [{
        "general": {
            "isTransmissionQueue": true,
            "description": "Transmission queue for remote admin."
        },
        "name": "QM1",
        "type": "local"
    },
   {
        "general": {
            "isTransmissionQueue": false,
            "description": "A queue on QM3."
        },
        "name" : "Qon3",
        "type" : "local"
    }]
}
```

Indicaciones de fecha y hora de REST API

Cuando administrative REST API devuelve información de fecha y hora, esta se devuelve como Hora universal coordinada (UTC) con un formato fijo.

La fecha y hora se devuelven con el siguiente el formato de indicación de fecha y hora:

```
YYYY-MM-DDTHH:mm:ss:sssZ
```

Por ejemplo, `2012-04-23T18:25:43.000Z`, donde la Z indica que el huso horario es la Hora universal coordinada (UTC).

La precisión de esta indicación de fecha y hora no está garantizada. Por ejemplo, si el servidor mqweb no se inicia en el mismo huso horario que el gestor de colas que se especifica en el URL del recurso, puede que la indicación de fecha y hora no sea exacta. Asimismo, si se requieren ajustes de horario de verano, puede que la indicación de fecha y hora no sea precisa.

Manejo de errores de REST API

REST API informa de los errores devolviendo un código de respuesta HTTP correspondiente, por ejemplo 404 (No encontrado), y una respuesta JSON. Cualquier código de respuesta HTTP que no esté en el rango 200 - 299 se considera un error.

Formato de respuesta de error

La respuesta está en formato JSON y en codificación UTF-8. Contiene objetos JSON anidados:

- Un objeto JSON externo que contiene una matriz JSON única denominada `error`.

- Cada elemento de la matriz es un objeto JSON que representa información sobre un error. Cada objeto JSON contiene las siguientes propiedades:

  Tipo
  Serie.

  Tipo de error.

  messageId
  Serie.

  Identificador exclusivo del mensaje en formato MQWBnnnnX. Este identificador tiene los siguientes elementos:

  MQWB
  Prefijo que muestra que el mensaje se ha originado en el API REST de MQ.

  nnnn
  Número exclusivo que identifica el mensaje.

  X
  Letra única que indica la gravedad del mensaje:

  - I si el mensajes es meramente informativo.
  - W si el mensaje es un aviso de un problema.
  - E si el mensaje indica que se ha producido un error.
  - S si el mensaje indica que se ha producido un error.

  mensaje
  Serie.

  Una descripción del error.

  explanation
  Serie.

  Una explicación del error.

  acción
  Serie.

  Una descripción de los pasos que se pueden realizar para resolver el error.

  qmgrName
  z/OS Este campo solo está disponible para z/OS, donde el gestor de colas es miembro del grupo de compartición de colas. Debe haber especificado el parámetro de consulta opcional **commandScope** o el atributo **queueSharingGroupDisposition**.

  Serie.

  El nombre del gestor de colas que ha experimentado el error.

  Este campo no se aplica para messaging REST API.

  completionCode
  Este campo solo está disponible cuando **type** es pcf, java o rest.

  Número.

  Código de terminación MQ asociado al error.

  reasonCode
  Este campo solo está disponible cuando **type** es pcf, java o rest.

  Número.

  Código de razón MQ asociado al error.

  exceptions
  Este campo solo está disponible cuando **type** es java.

  Matriz.

Una matriz de excepciones Java o JMS de cadena. Cada elemento de la matriz de excepciones contiene una matriz de series **stackTrace**.

La matriz de series **stackTrace** contiene los detalles de cada excepción divididos en líneas.

Errores con los grupos de compartición de colas

z/OS

En un grupo de compartición de colas, se puede especificar un parámetro de consulta opcional de **commandScope** en determinados mandatos. Este parámetro permite propagar el mandato a otros gestores de colas del grupo de compartición de colas. Cualquiera de estos mandatos puede fallar independientemente, por lo que algunos mandatos terminarán satisfactoriamente y otros fallarán para el grupo de compartición de colas.

En los casos en los que un mandato falla parcialmente, se devuelve un código de error HTTP 500. Para cada gestor de colas que genera un error, se devuelve información sobre ese error como un elemento en la matriz JSON `error`. Para cada gestor de colas que ejecuta correctamente el mandato, se devuelve el nombre del gestor de colas como un elemento en una matriz JSON `success`.

Ejemplos

- El ejemplo siguiente muestra la respuesta de error a un intento de obtener información sobre un gestor de colas que no existe:

```
"error": [
   {
      "type": "rest",
      "messageId": "MQWB0009E",
      "message": "MQWB0009E: Could not query the queue manager 'QM1'",
      "explanation": "The MQ REST API was invoked specifying a queue manager name which
cannot be located.",
      "action": "Resubmit the request with a valid queue manager name or no queue manager
name, to retrieve a list of queue managers. "
   }
]
```

- z/OS El ejemplo siguiente muestra la respuesta de error a un intento de suprimir una cola en un grupo de compartición de colas que no existe para algunos gestores de colas:

```
"error" : [
  {
    "type": "rest",
    "messageId": "MQWB0037E",
    "message": "MQWB0037E: Could not find the queue 'missingQueue' - the queue manager reason
code is 3312 : 'MQRCCF_UNKNOWN_OBJECT_NAME'",
    "explanation": "The MQ REST API was invoked specifying a queue name which cannot be
located.",
    "action": "Resubmit the request with the name of an existing queue, or with no queue name
to retrieve a list of queues.",
    "qmgrName": "QM1"
  },
  {
    "type": "rest",
    "messageId": "MQWB0037E",
    "message": "MQWB0037E: Could not find the queue 'missingQueue' - the queue manager reason
code is 3312 : 'MQRCCF_UNKNOWN_OBJECT_NAME'",
    "explanation": "The MQ REST API was invoked specifying a queue name which cannot be
located.",
    "action": "Resubmit the request with the name of an existing queue, or with no queue name
to retrieve a list of queues.",
    "qmgrName": "QM2"
  }
],
"success" : [{"qmgrName": "QM3"}, {"qmgrName": "QM4"}]
```

Errores con solicitudes MFT

Si los servicios MFT REST API no están habilitados, e invoca MFT REST API, recibirá la excepción siguiente:

```
{"error": [{
  "action": "Enable the Managed File Transfer REST API and resubmit the request.",
  "completionCode": 0,
  "explanation": "Managed File Transfer REST calls are not permitted as the service is
disabled.",
  "message": "MQWB0400E: Managed File Transfer REST API is not enabled.",
  "msgId": "MQWB0400E",
  "reasonCode": 0,
  "type": "rest"
}]}
```

Si los servicios MFT REST API están habilitados y el gestor de colas de coordinación no está establecido en el archivo `mqwebuser.xml`, recibirá la siguiente excepción:

```
{"error": [{
  "action": "Set the coordination queue manager name and restart the mqweb server.",
  "completionCode": 0,
  "explanation": "Coordination queue manager name must be set before using Managed File
Transfer REST services.",
  "message": "MQWB0402E: Coordination queue manager name is not set.",
  "msgId": "MQWB0402E",
  "reasonCode": 0,
  "type": "rest"
}]}
```

Descubrimiento de REST API

La documentación de REST API está disponible en IBM Documentation y en formato Swagger. Swagger es un enfoque utilizado comúnmente para documentar las API REST. La documentación de Swagger para REST API se puede ver habilitando la característica de descubrimiento de API (apiDiscovery) en el servidor mqweb.

Antes de empezar

Stabilized

Importante: La característica apiDiscovery se ha estabilizado. Todavía puede utilizar esta característica. En la actualidad, IBM MQ no da soporte al uso de la característica `mpOpenAPI` .

Debe habilitar la seguridad para el servidor mqweb para ver la documentación Swagger utilizando el descubrimiento de API. Si desea más información sobre los pasos que son necesarios para habilitar la seguridad, consulte Seguridad de IBM MQ Console y REST API.

Procedimiento

1. Localice el archivo `mqwebuser.xml` en uno de los directorios siguientes:
   - ALW *MQ_DATA_PATH*/web/installations/*installationName*/servers/mqweb
   - z/OS *WLP_user_directory*/servers/mqweb

     Donde *WLP_user_directory* es el directorio que se especificó cuando se ejecutó el script **crtmqweb** para crear la definición del servidor mqweb.
2. Añada el XML adecuado al archivo `mqwebuser.xml`:
   - Si las etiquetas `<featureManager>` existen en el archivo `mqwebuser.xml`, añada el XML siguiente dentro de las etiquetas `<featureManager>`:

     ```
     <feature>apiDiscovery-1.0</feature>
     ```

- Si las etiquetas <featureManager> no existen en el archivo mqwebuser.xml, añada el XML siguiente dentro de las etiquetas <server>:

```
<featureManager>
     <feature>apiDiscovery-1.0</feature>
</featureManager>
```

3. Vea la documentación Swagger utilizando uno de los métodos siguientes:
   - Visualice una página web que pueda examinar y pruebe REST API especificando el URL siguiente en un navegador:

     https://*host*:*port*/ibm/api/explorer

     Además de autenticar cada solicitud, debe incluir una cabecera ibm-mq-rest-csrf-token para cada solicitud POST, PATCH o DELETE. El contenido de esta cabecera puede ser cualquier serie, incluido el espacio en blanco.

     Esta cabecera de solicitud se utiliza para confirmar que las credenciales para autenticar la solicitud las utiliza el propietario de las credenciales. Es decir, se utiliza la señal para impedir ataques de falsificación de solicitudes entre sitios.
   - Recupere un único documento Swagger 2 que describe completamente REST API emitiendo HTTP GET al URL siguiente:

     https://*host*:*port*/ibm/api/docs

     Este documento puede utilizarse para aplicaciones donde desea navegar mediante programa en las API disponibles.

   host
   Especifica el nombre de host o la dirección IP en la que está disponible REST API.

   El valor predeterminado es localhost.

   port
   Especifica el número de puerto HTTPS que utiliza administrative REST API.

   El valor predeterminado es 9443.

   Si el nombre de host o número de puerto cambia respecto al valor predeterminado, puede determinar los valores correctos a partir del URL de REST API. Utilice el mandato **dspmqweb status** para ver el URL.

Información relacionada
Estado de dspmqweb (visualizar estado de servidor mqweb)

Soporte multilingüístico de REST API

REST API soporta, con determinadas cualificaciones, la capacidad de especificar idiomas nacionales como parte de una solicitud HTTP.

En segundo plano

Las cabeceras HTTP permiten que se especifique un comportamiento concreto en solicitudes y que se proporcione información adicional en respuestas.

Se incluye en las cabeceras HTTP la capacidad de solicitar que dicha información se devuelva en un idioma nacional. REST API respeta esta cabecera siempre que sea posible.

Especificar un idioma nacional

En la cabecera ACCEPT-LANGUAGE HTTP, se puede proporcionar uno o más códigos de idioma. De forma opcional, puede asociar una clasificación con los códigos, lo que le permite la especificación de una lista ordenada por preferencia. Esta página tiene un debate útil del principio.

REST API respeta esta cabecera, seleccionando un idioma de la cabecera ACCEPT-LANGUAGE y devolviendo mensajes en dicho idioma. Cuando la cabecera ACCEPT-LANGUAGE no contiene ningún idioma que REST API pueda soportar, los mensajes se devuelven en un idioma predeterminado. Este idioma predeterminado corresponde al entorno local predeterminado del servidor web de REST API.

La sección "¿Qué datos se traducen?" en la página 90 explica qué datos se traducen.

Indicar el idioma aplicable en las respuestas

La cabecera HTTP CONTENT-LANGUAGE en las respuestas de REST API indica el idioma en el cual se devuelven los mensajes.

¿Qué datos se traducen?

Los mensajes de error e informativos se traducen, otro texto no.

- Los datos que se devuelven de un gestor de colas no se traducen, por ejemplo, en el caso de ejecutar un mandato MQSC mediante REST API, las respuestas del gestor de colas están en el entorno local del gestor de colas.
- La documentación generada (Swagger) para REST API, tal como se expone mediante la característica `apiDiscovery`, está en inglés.

¿Qué idiomas están soportados?

Además del inglés, los mensajes de error e informativos de REST API están traducidos a los idiomas siguientes.

Chino (Simplificado)
Indicado por el código de idioma `zh_CN`

Chino (Tradicional)
Indicado por el código de idioma `zh_TW`

Checo
Indicado por el código de idioma `cs`

Francés
Indicado por el código de idioma `fr`

Húngaro
Indicado por el código de idioma `hu`

Italiano
Indicado por el código de idioma `it`

Japonés
Indicado por el código de idioma `ja`

Coreano
Indicado por el código de idioma `ko`

Polaco
Indicado por el código de idioma `pl`

Portugués (Brasil)
Indicado por el código de idioma `pt_BR`

Ruso
Indicado por el código de idioma `ru`

Español
Indicado por el código de idioma `es`

Ejemplos

En los ejemplos, el servidor web tiene un entorno local inglés.

Especificación de un único idioma soportado

En las cabeceras de petición, ACCEPT-LANGUAGE está establecido en `fr`. Este valor especifica que el francés es el idioma preferido para el texto traducible.

En las cabeceras de respuesta, CONTENT-LANGUAGE está establecido en `fr`. Este valor indica que los mensajes de error e información en la respuesta están en francés.

Especificación de una lista de idiomas

En las cabeceras de petición, ACCEPT-LANGUAGE está establecido en `am, fr`. Este valor especifica que el amhárico y el francés son idiomas aceptables para texto traducible y que el amhárico es el idioma preferido para el texto traducible.

En las cabeceras de respuesta, CONTENT-LANGUAGE está establecido en `fr`. Este valor indica que los mensajes de error e informativos en la respuesta están en francés, ya que REST API no soporta el amhárico.

Especificación de un único idioma no soportado

En las cabeceras de petición, ACCEPT-LANGUAGE está establecido en `am`. Este valor especifica que el amhárico es el idioma preferido para el texto traducible.

En las cabeceras de respuesta, CONTENT-LANGUAGE está establecido en en. Este valor indica que los mensajes de error e informativos en la respuesta están en inglés, ya que REST API no soporta el amhárico.

Versiones de REST API

El número de versión de REST API forma parte del URL base para solicitudes REST. Por ejemplo, `https://localhost:9443/ibmmq/rest/`**`v2`**`/admin/installation`. El número de versión se utiliza para aislar los clientes de los cambios en la REST API que se podrían introducir en futuros releases.

IBM MQ 9.2.0 introduce la versión 2 de REST API. Este aumento de versión se aplica a administrative REST API, messaging REST API y MFT REST API. Este aumento de versión cambia el URL de recurso que se utiliza para la REST API. El prefijo de URL para los URL de recurso de la versión 2 es el URL siguiente:

```
https://host:port/ibmmq/rest/v2/
```

Stabilized Algunos cambios que se han introducido en la REST API podrían cambiar la función de la REST API existente como, por ejemplo, que los clientes que utilizan la REST API podrían tener que actualizarse. Para impedir que estos cambios fuercen la actualización de los clientes, el número de versión de REST API se aumenta y la función existente se estabiliza en el número anterior. La nueva función que podría cambiar la función existente se añade a la REST API en el nuevo número de versión. Por lo tanto, los clientes pueden seguir utilizando la REST API en la versión anterior sin actualizarse.

Los cambios de REST API que podrían dar como resultado que sea necesario una actualización de cliente incluyen los cambios siguientes:

- La eliminación del soporte para un atributo existente en el JSON que se envía a, o devuelve de, la REST API.
- La eliminación de un URL, verbo HTTP o cabecera. Por ejemplo, si se renombra un URL o una cabecera, o si se utiliza un verbo diferente.
- La adición de un nuevo atributo JSON obligatorio a los datos que se envían a un URL existente.
- La adición de una nueva cabecera HTTP obligatoria a los datos que se envían a un URL existente.
- La adición de un nuevo parámetro de consulta obligatorio a un URL existente.

Cuando este tipo de cambio se introduce en la función de REST API que existía en un release de Long Term Support (LTS), el número de versión de la REST API se aumenta para el primero de estos cambios. Cualquier cambio posterior que se haya realizado en un release de Continuous Delivery (CD) que pueda requerir cambios en los clientes que utilizan la REST API utiliza el nuevo número de versión.

Este número de versión sigue siendo el mismo a lo largo de todos los releases posteriores de CD hasta el siguiente release de LTS. Por lo tanto, el número de versión aumenta como máximo una vez entre los releases de LTS.

Stabilized Cuando el número de versión se aumenta, la función de REST API existente se estabiliza en el número de versión antiguo. Es decir, la función REST API existente que estaba disponible en el release de LTS sigue estando disponible en el número de versión antigua, pero no se realizan más cambios en esa versión. Cualquier función nueva que se añada a la REST API se añade a la nueva versión de la REST API. Sin embargo, no se garantiza que las adiciones que se realizan en la REST API en los releases de CD antes del aumento de versión estén incluidas en la versión más antigua de la REST API.

Deprecated Los clientes existentes pueden seguir utilizando la REST API en el número de versión antiguo sin necesitar ningún cambio. Las versiones más antiguas de la REST API podrían estar en desuso y, finalmente, haber sido eliminadas.

Algunos cambios no requieren cambios en los clientes que utilizan la REST API. Estos cambios no generan un aumento del número de versión. Por lo tanto, asegúrese de que cualquier cliente que utiliza la REST API no es necesario que se actualice cuando se introduzcan estos tipos de cambios. Estos cambios en la REST API podrían incluir los cambios siguientes:

- Adición de un nuevo atributo JSON a los datos existentes que se devuelven de la REST API.
- Adición de un nuevo URL.
- Adición de un nuevo verbo HTTP a un URL existente.
- Adición de un nuevo código de estado a un URL existente.
- Adición de nuevos atributos JSON opcionales a los datos que se envían a un URL existente.
- Adición de nuevos parámetros de consulta en un URL existente.
- Adición de nuevas cabeceras a los datos que se envían a un URL existente.
- Devolución de nuevas cabeceras de la REST API.

Cambios en la nueva función de la API REST Continuous Delivery

Para la nueva función de REST API que se ha añadido en un release de CD, los cambios realizados en esta nueva función que podrían necesitar cambios en los clientes de REST API no aumentan el número de versión. Es decir, la nueva función puede cambiar antes del siguiente release de LTS sin aumentar el número de versión. Cuando la función está incluida en un release de LTS, los cambios posteriores que podrían requerir cambios en clientes de REST API no aumentan el número de versión.

Ejemplo

1. En el release X de LTS, la REST API está en la versión 1.
2. En el release CD X.0.1, se ha añadido el soporte para un nuevo URL. Este cambio no requiere cambios en clientes que utilizan la REST API. Por lo tanto, la REST API permanece en la versión 1.
3. En CD X.0.2, se ha añadido soporte para un nuevo URL. Este cambio no requiere cambios en los clientes que utilizan la API REST. Por lo tanto, la REST API permanece en la versión 1.
4. En LTS release Y, la REST API está en la versión 1.
5. En CD release Y.0.1, se renombra un URL existente. Este cambio podría requerir cambios en los clientes que utilizan la REST API. Por consiguiente, se crea una nueva versión de REST API como versión 2. El URL renombrado se incluye en la versión 2 de REST API, junto con toda la función existente. Cualquier nueva función que se añada a REST API se añade a la versión 2. La versión 1 permanece estabilizada en el nivel del release Y de LTS.
6. En CD release Y.0.2, se renombra otro URL existente. Como la versión ya se ha incrementado en CD release Y, REST API permanece en la versión 2. La versión 1 permanece estabilizada en el nivel del release Y de LTS.
7. En LTS release Z, REST API permanece en la versión 2. La versión 1 permanece estabilizada en el nivel del release Y de LTS.

Administración utilizando IBM MQ Console

Puede realizar tareas de administración básicas utilizando la IBM MQ Console.

Nota: No inhabilite el servidor de mandatos en ninguno de los gestores de colas cuando utilice IBM MQ Console. Si el servidor de mandatos está inhabilitado para un gestor de colas:

- El IBM MQ Console deja de responder, con largos retardos en el proceso de los mandatos
- Los mandatos que se emiten al gestor de colas agotan el tiempo de espera.

Tareas relacionadas

Rastreo de IBM MQ Console

Iniciación a IBM MQ Console

Configure el servidor mqweb; determine el URI del IBM MQ Console; conéctese a la consola; inicie sesión en la consola.

Antes de empezar

Para completar esta tarea, debe ser un usuario con determinados privilegios para que pueda utilizar el mandato **dspmqweb**:

- z/OS En z/OS, debe tener autorización para ejecutar el mandato **dspmqweb** y acceso de escritura en el archivo `mqwebuser.xml`.
- Multi En todos los demás sistemas operativos, debe ser un usuario con privilegios.

IBM i En IBM i, los mandatos deben estar en ejecución en QSHELL.

Acerca de esta tarea

Tenga en cuenta las restricciones siguientes:

- z/OS
  - No pueden crearse, suprimirse, iniciarse ni detenerse gestores de colas en z/OS.
  - Los iniciadores de canal en z/OS no pueden iniciarse ni detenerse, y no se muestra el estado del iniciador de canal.
  - Los escuchas no pueden visualizarse ni administrarse.
  - Los mandatos para iniciar, ejecutar ping, resolver y restablecer el canal solo pueden emitirse con CHLDISP(DEFAULT).
  - Los objetos definidos con QSGDISP(GROUP) no pueden visualizarse ni gestionarse.
  - La seguridad del gestor de colas no puede gestionarse.
  - El uso de recursos de sistema no se puede supervisar.
- Multi
  - No puede utilizar IBM MQ Console para trabajar con canales AMQP.
  - No puede utilizar IBM MQ Console para trabajar con canales MQTT .

Procedimiento

1. Si el servidor mqweb todavía no está configurado para ser utilizado por IBM MQ Console, configure el servidor mqweb.

   Para obtener más información sobre cómo configurar el servidor mqweb con un registro básico, consulte Configuración básica para el servidor mqweb.

2. z/OS
   En z/OS, establezca la variable de entorno WLP_USER_DIR para que pueda utilizar el mandato **dspmqweb** . Establezca la variable para que apunte a la configuración del servidor mqweb especificando el mandato siguiente.

   ```
   export WLP_USER_DIR=WLP_user_directory
   ```

   donde *WLP_user_directory* es el nombre del directorio que se pasa a `crtmqweb`. Por ejemplo:

   ```
   export WLP_USER_DIR=/var/mqm/web/installation1
   ```

   Si desea más información, consulte Creación del servidor mqweb.
3. Determine el URI para IBM MQ Console especificando el mandato siguiente:

   ```
   dspmqweb status
   ```

   El mandato genera una salida similar a la siguiente:

   ```
   MQWB1124I: Server 'mqweb' is running.
      URLS:
      https://localhost:9443/ibmmq/rest/v1/
      https://localhost:9443/ibmmq/console/
   ```

   El URI para la IBM MQ Console finaliza con el sufijo `console/`.
4. Conéctese a IBM MQ Console especificando el URL del paso anterior en un navegador.

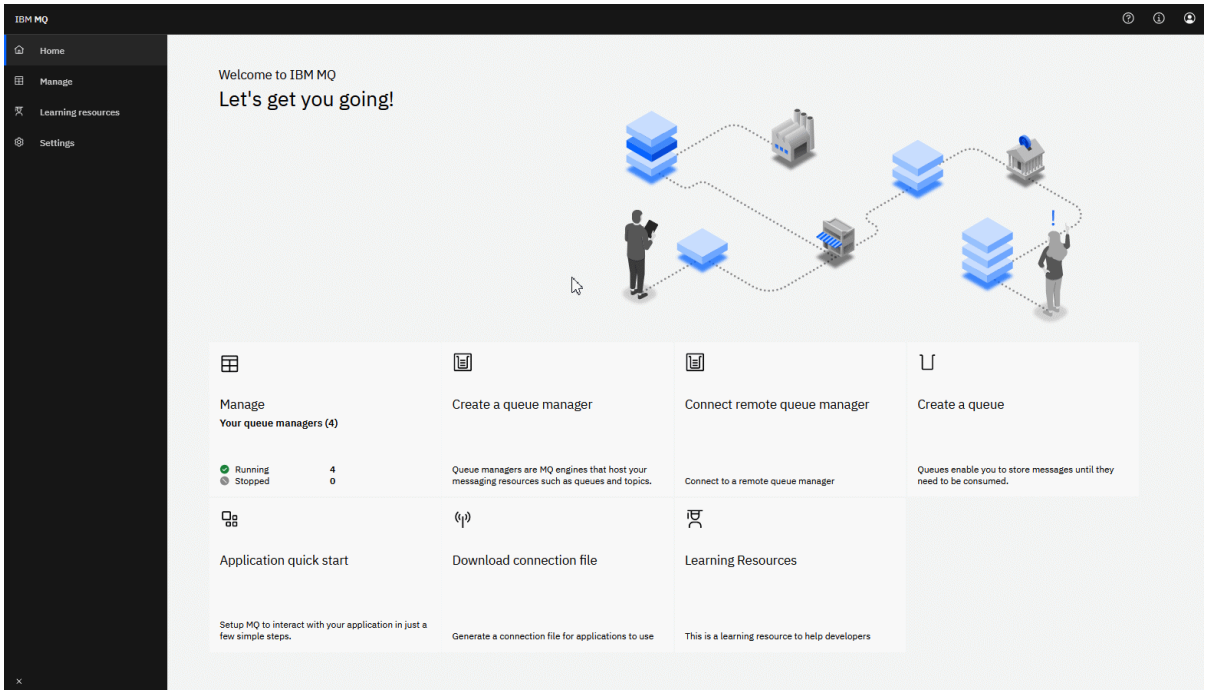
   El navegador podría generar una excepción de seguridad porque el certificado predeterminado que se proporciona con el servidor no es un certificado de confianza. Elija continuar con la IBM MQ Console.
5. Inicie la sesión en IBM MQ Console. Utilice el nombre de usuario `mqadmin` y la contraseña `mqadmin`.

Qué hacer a continuación

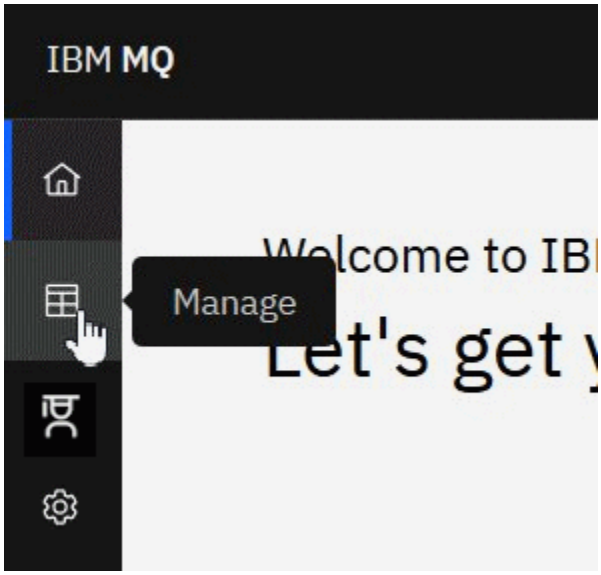
De forma predeterminada, la IBM MQ Console utiliza la autenticación basada en señales para autenticar usuarios. También puede utilizar la autenticación de certificado de cliente. Para obtener más información, consulte Utilización de la autenticación de certificados de cliente con REST API y IBM MQ Console.

V 9.4.0 Visita rápida de la IBM MQ Console

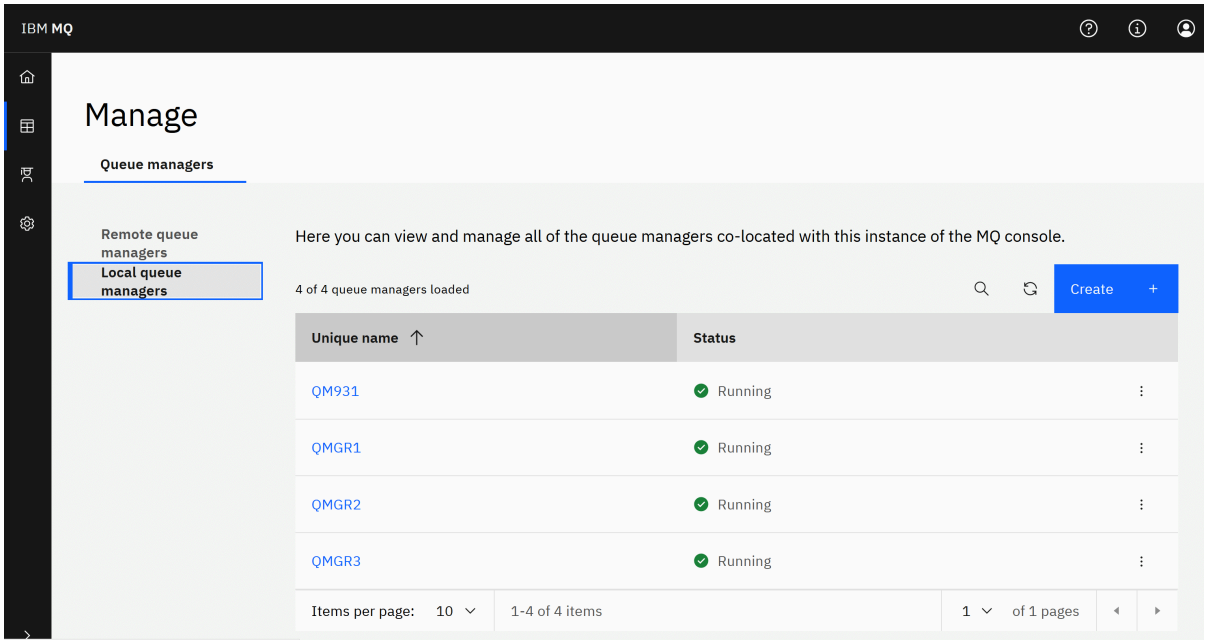
La primera vez que inicia sesión en IBM MQ Console se le dirige a la página de destino. Desde aquí puede elegir gestionar gestores de colas existentes, crear un gestor de colas o una cola, navegar a algunos temas de formación o abrir la información del producto IBM MQ en IBM Documentation. También puede iniciar el inicio rápido de la aplicación, que le guía a través del proceso de configuración rápida y sencilla de la mensajería entre gestores de colas y aplicaciones nuevos o existentes.



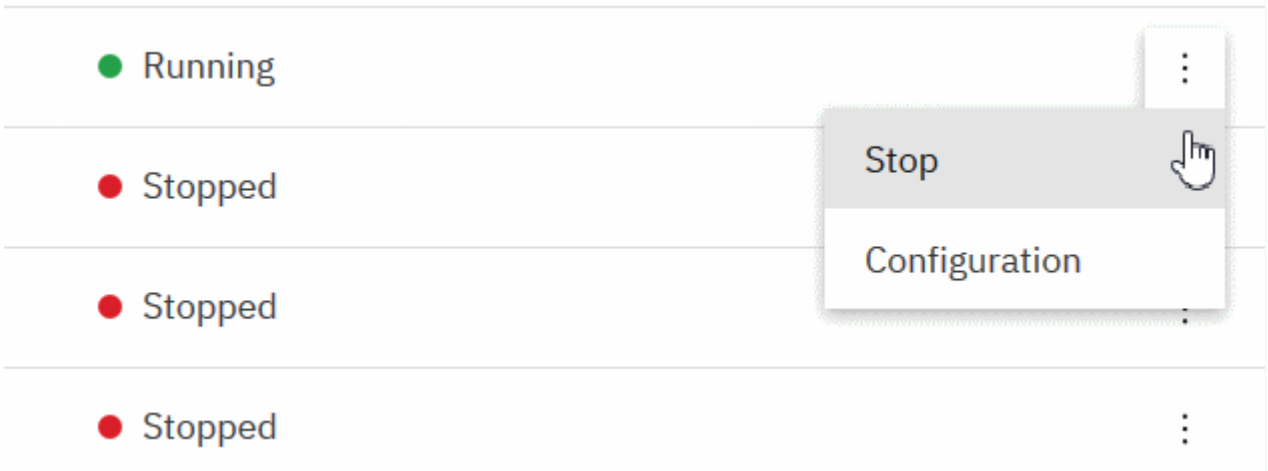
Como alternativa, puede simplemente pulsar el icono Gestionar para empezar a gestionar directamente los objetos de IBM MQ.



La vista de gestión muestra inicialmente los gestores de colas y su estado actual. También puede crear nuevos gestores de colas y conectar gestores de colas remotos.

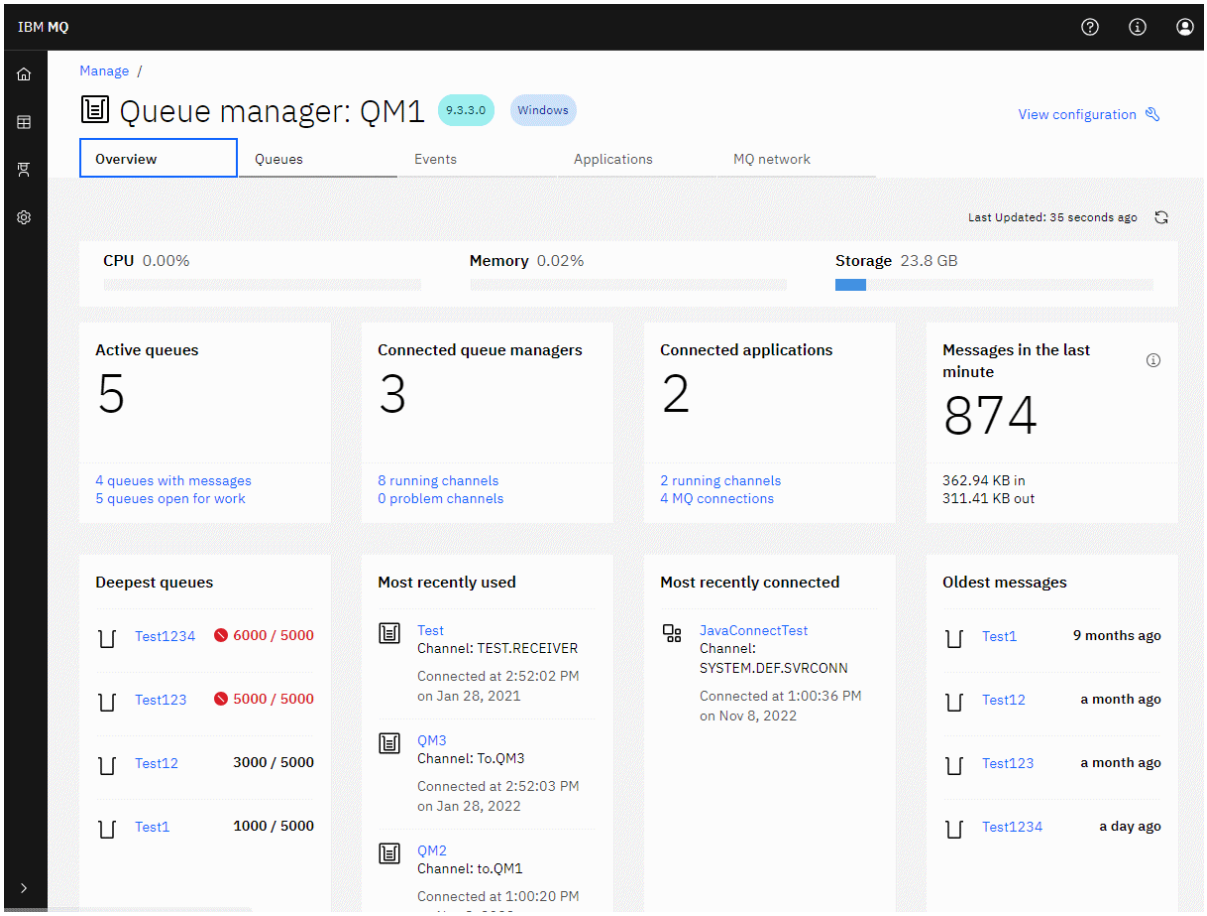


Cada gestor de colas tiene un menú que le permite detener o configurar un gestor de colas en ejecución, o iniciar o suprimir un gestor de colas detenido.



Los registros de autorizaciones, los objetos de información de autenticación y los registros de autenticación de canal para el gestor de colas se pueden encontrar en la pestaña **Seguridad** de la página **Configuración** del gestor de colas, donde puede crear y añadir unos nuevos.

Pulse el nombre de un gestor de colas en ejecución para abrir su panel de instrumentos.



Desde el panel de instrumentos del gestor de colas, puede completar las acciones siguientes:

V 9.4.0 En la pestaña **Visión general** , vea la información siguiente:

CPU
Estimación porcentual del uso de CPU por parte del gestor de colas. (No aplicable en z/OS.)

Memoria
Estimación porcentual del uso de memoria por parte del gestor de colas. (No aplicable en z/OS o Windows.)

Almacenamiento
Estimación porcentual del espacio libre del disco en el que reside el gestor de colas. (No aplicable en z/OS.)

Colas activas
Recuento de colas que tienen mensajes o que están abiertas para entrada o salida.

Gestores de colas conectados
Recuento de gestores de colas conectados actualmente derivados de canales activos.

Aplicaciones conectadas
Recuento de aplicaciones conectadas actualmente.

Mensajes en el último minuto
Muestra un resumen de los temas del sistema PUT/GET que muestran el rendimiento de los mensajes cada 10 segundos. (No aplicable en z/OS.)

Suscripciones
Muestra un recuento de suscripciones. Solo está visible en z/OS y en otras plataformas donde la supervisión de temas del sistema está inhibida (consulte propiedades setmqweb).

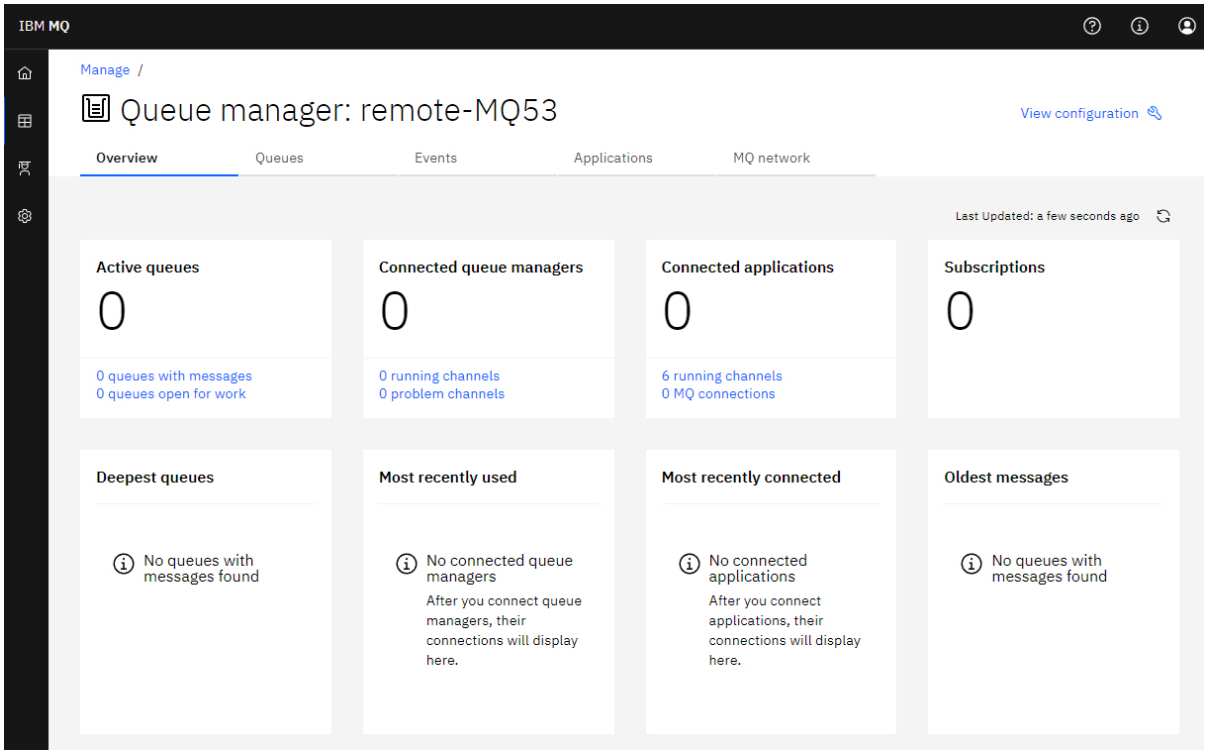
Colas más profundas
Lista las colas en orden de profundidad. Muestra la profundidad de cola actual y la profundidad de cola máxima.

Utilizado más recientemente
Lista los gestores de colas conectados actualmente, ordenados por fecha del último mensaje.

Conectado más recientemente
Lista las aplicaciones conectadas actualmente como derivadas de canales de conexión de servidor activos, ordenados por fecha y hora de inicio de canal.

Mensajes más antiguos
Lista las colas ordenadas por la fecha y hora del mensaje más antigua.

La información que se muestra en la pestaña **Visión general** se deriva de la supervisión de temas del sistema (consulte Métricas publicadas en los temas del sistema). z/OS no da soporte a la supervisión de temas del sistema y la supervisión para la visualización de la consola se puede inhabilitar en otras plataformas (consulte propiedades setmqweb). En estos casos, la pestaña **Visión general** muestra información más limitada y su aspecto es similar al del ejemplo siguiente:



En la pestaña **Colas**:

- Crear colas nuevas
- Pulse en un nombre de cola para ver los mensajes existentes y crear otros nuevos, y para configurar la cola.

En la pestaña **Sucesos** :

Temas

- Crear temas nuevos
- Configurar los temas existentes
- Pulse un nombre de tema para ver las suscripciones coincidentes

Suscripciones

- Crear suscripciones nuevas gestionadas o sin gestionar

- Configurar las suscripciones existentes 

V 9.4.0 En la pestaña **Aplicaciones** :

Visión general
Contiene mosaicos que ofrecen una visión general de las estadísticas siguientes:

Aplicaciones conectadas
Muestra un recuento del número de aplicaciones conectadas. Proporciona enlaces a las pestañas siguientes:

- **Instancias de aplicación**
- **Conexiones**

Ejecución de instancias de canal
Muestra un recuento del número de instancias de canal SVRCONN y desde ese enlace a las definidas o detenidas en la pestaña **Canal de aplicación** .

Conexiones
Muestra un recuento del número de conexiones. Proporciona enlaces a la información siguiente en el separador **Conexiones** :

- Conexiones locales (aquellas sin un nombre de canal)
- Conexiones remotas (aquellas con un nombre de canal)

Aplicaciones más comunes
Muestra una lista de aplicaciones frecuentes, ordenadas por el número de conexiones que están utilizando.

Canales más comunes
Muestra una lista de canales frecuentes, ordenados por el número de instancias que están activas.

Transacciones más antiguas
Muestra una lista de las transacciones más antiguas por nombre de aplicación. Estas transacciones tienen conexiones con unidades de trabajo abiertas y se ordenan por fecha y hora de inicio de UOW.

Versiones conectadas remotas
Muestra una lista de las versiones de IBM MQ conectadas comunes, es decir, las instancias de canal que tienen una REMOTE_VERSION especificada.

Seguridad de canal de aplicación
Muestra una lista de protocolos de seguridad de canal conectados comunes, es decir, las instancias de canal que tienen un SECURITY_PROTOCOL especificado.

Velocidades de transferencia de canal
Muestra una lista de canales comunes ordenados por las tasas de transferencia de mensajes y bytes. Utiliza la fecha y hora de inicio del canal para calcular la duración y utiliza MSGS y MQIACH_BYTES_SENT/ MQIACH_BYTES_RCVD para calcular la tasa.

Aplicaciones
Ver información sobre las aplicaciones que están conectadas al gestor de colas.

Canales
Ver actividad en canales conectados a aplicaciones.

Canales de aplicación

- Iniciar, detener, hacer ping y configurar canales
- Crear nuevos canales
- Restablecer canales

Instancias de canal de aplicación

- Ver estado de instancias de canal de aplicación
- Resolver mensajes pendientes en canales

V 9.4.0 En la pestaña **Red de MQ** :

Visión general

Contiene mosaicos que ofrecen una visión general de las estadísticas siguientes:

Ejecución de instancias de canal de gestor de colas

Muestra un recuento del número de instancias de canal no SVRCONN. Muestra enlaces a los siguientes tipos de instancias de canal en la pestaña **Gestores de colas conectados** :

- Canales definidos
- Canales detenidos

Gestores de colas conectados

Muestra un recuento de los gestores de colas conectados mediante MQCA_REMOTE_Q_MGR_NAME. También proporciona un recuento de gestores de colas devueltos por MQCMD_INQUIRE_CLUSTER_Q_MGR.

Pertenencia al clúster

Si sólo hay un clúster de gestores de colas, muestra el nombre del clúster y si el gestor de colas es un repositorio completo o parcial. Muestra cuántos gestores de colas están visibles en el clúster. Si hay más de un clúster, muestra el número de clústeres más un recuento de gestores de colas de repositorio completo y parcial en cada clúster.

Canales de gestor de colas anómalos

Muestra una lista de canales que están en un estado de reintento (no detenido/en ejecución). Calcula el número de reintentos restantes si están en estado de reintento. La lista contiene canales con los siguientes tipos de estado:

- MQCHS_PAUSED
- MQCHS_RETRYING

Retardos de mensajes más largos

Muestra una lista de canales que tienen un indicador de tiempo XMIT (periodo largo).

Colas de transmisión desatendidas

Muestra una lista de colas de transmisión que tienen una profundidad de cola distinta de cero y ningún descriptor de contexto asociado.

Versiones de conexión remota

Muestra una lista de las versiones de IBM MQ conectadas comunes, es decir, las instancias de canal que tienen una REMOTE_VERSION especificada.

Seguridad de canal de gestor de colas

Muestra una lista de protocolos de seguridad de canal conectados comunes, es decir, las instancias de canal que tienen un SECURITY_PROTOCOL especificado.

Estado del clúster

Muestra una serie de estadísticas independientes relacionadas con el estado del clúster. El estado incluye:

- Número de objetos de clúster (colas, temas, gestores de colas).
- Número de gestores de colas suspendidos (MQIACF_SUSPEND establecido en YES).
- La profundidad del SYSTEM.CLUSTER.COMMAND.QUEUE .
- Número de entradas de gestor de colas de clúster que empiezan por SYSTEM.TEMP.

Si todos estos valores son cero, este mosaico no se visualiza y, en su lugar, se muestra el mosaico **Escuchas** .

Escuchas
Muestra una lista de escuchas y si están en un estado de ejecución. Sólo se muestra si no se visualiza el mosaico **Estado del clúster** .

Gestores de colas conectados
Ver detalles de los gestores de colas conectados actualmente a este gestor de colas.

Canales de gestores de colas

- Iniciar, detener, hacer ping y configurar canales
- Crear nuevos canales
- Restablecer canales

Instancias de canal de gestor de colas

- Ver estado de instancias de canal de gestor de colas
- Resolver mensajes pendientes en canales

V 9.4.0 IBM MQ Console: Trabajar con gestores de colas locales

Puede crear, configurar y controlar los gestores de colas locales desde el nivel superior de la vista Gestionar Manage.

Acerca de esta tarea

Multi La vista Gestionar enumera los gestores de colas locales añadidos a la instalación de IBM MQ desde la que se está ejecutando IBM MQ Console. Los gestores de colas asociados a distintas instalaciones de IBM MQ en el mismo sistema no se enumeran.

z/OS En z/OS, la vista Gestionar lista los gestores de colas que están en la misma versión que IBM MQ Console y se definen en el sistema en el que se ejecuta IBM MQ Console. Los gestores de colas de una versión diferente a la de IBM MQ Console no se enumeran.

Puede seleccionar gestores de colas individuales en la lista para trabajar con ellos.

Nota: El IBM MQ Console puede conectarse a un gestor de colas RDQM local cuando está activo (es decir, tiene el rol primario), pero no ofrece ninguna característica específica de RDQM.

Procedimiento

- Para crear un nuevo gestor de colas local:
  a) Pulse el botón Crear Create + en la vista de lista del gestor de colas.
  b) Escriba un nombre para el nuevo gestor de colas. El nombre puede tener hasta 48 caracteres. Los caracteres válidos son letras y números y los caracteres ".", "/", "_" y "%".
  c) Opcional: Especifique un puerto TCP/IP disponible para la escucha del gestor de colas. El número de puerto no debe exceder 65535.
  d) Pulse **Crear**. Se crea y se inicia el nuevo gestor de colas.
- Para iniciar un gestor de colas local:
  a) Localice el gestor de colas que desea iniciar en la lista.
  b) Seleccione **Iniciar** en el menú .
- Para detener un gestor de colas local:

a) Seleccione el gestor de colas que desea detener de la lista en el widget de gestor de colas local.

b) Seleccione **Detener** en el menú .

- Para suprimir un gestor de colas local:
  a) Si el gestor de colas está en ejecución, deténgalo.

  b) Seleccione **Ver configuración** en el menú y seleccione **Suprimir gestor de colas**.

  c) Confirme que desea suprimir el gestor de colas especificando su nombre en la ventana de confirmación. Se suprimen el gestor de colas y todos los objetos asociados.
- Para ver y editar las propiedades de un gestor de colas local:
  a) Asegúrese de que el gestor de colas esté en ejecución y localícelo en la lista de gestores de colas.

  b) Seleccione **Ver configuración** en el menú .

  c) Asegúrese de que la pestaña **Propiedades** está seleccionada. Consulte las propiedades y edítelas según sea necesario. Si el recuadro de texto de propiedad está inhabilitado, la propiedad es de sólo lectura o solo se puede editar desde la línea de mandatos. Para obtener información sobre una propiedad, puede ver la información de propiedades en Propiedades del gestor de colas.
- Para trabajar con la configuración de seguridad del gestor de colas local:
  a) Asegúrese de que el gestor de colas está en ejecución, y selecciónelo en la lista del gestor de colas.

  b) Seleccione **Ver configuración** en el menú .

  c) Asegúrese de que la pestaña **Seguridad** esté seleccionada.

  d) Puede trabajar con objetos de autenticación, registros de autorizaciones u objetos de autenticación de canal. Visite los temas siguientes para obtener más información:
     - “IBM MQ Console: Trabajar con objetos de información de autenticación” en la página 102
     - “IBM MQ Console: Trabajar con registros de autorización de gestor de colas” en la página 104
     - “IBM MQ Console: Trabajar con registros de autenticación de canal” en la página 105

V 9.4.0 *IBM MQ Console: Trabajar con objetos de información de autenticación*

Puede utilizar la consola para añadir y suprimir objetos de información de autenticación en un gestor de colas. También puede ver y establecer las propiedades y gestionar los registros de autorizaciones del canal.

Acerca de esta tarea

La vista de información de autenticación lista la información de autenticación que existe para un gestor de colas específico. Puede seleccionar información de autenticación individual de la lista para trabajar con ella.

La información de autenticación del gestor de colas forma parte del soporte de IBM MQ para la seguridad de la capa de transporte (TLS). Estos objetos contienen las definiciones necesarias para realizar la comprobación de revocación de certificados mediante OCSP o listas de revocación de certificados (CRL) en los servidores LDAP y las definiciones necesarias para habilitar la comprobación de ID de usuario y contraseña.

Procedimiento

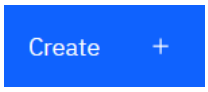
- Para ver la información de autenticación de un gestor de colas:
  a) Asegúrese de que el gestor de colas está en ejecución, y selecciónelo en la lista del gestor de colas.

b) Seleccione **Ver configuración** en el menú .

c) Asegúrese de que la pestaña **Seguridad** esté seleccionada.

d) Seleccione **Información de autenticación** en el panel de navegación.

- Para añadir un objeto de información de autenticación:

a) Pulse el botón Crear 

en la vista de lista de información de autenticación.

b) Especifique el nombre del objeto de información de autentificación. Los caracteres válidos son letras y números y los caracteres ".", "/", "_" y "%".

c) Especifique el tipo de objeto de información de autenticación.

d) Especifique información adicional adecuada para el tipo de objeto:

  - Para **CRL LDAP**, especifique el **Nombre del servidor LDAP**. Se trata del nombre de host, la dirección IPv4 en notación decimal con puntos o IPv6 en notación hexadecimal del host en el que se está ejecutando el servidor LDAP, con un número de puerto opcional. Opcionalmente, puede especificar un nombre de usuario y una contraseña para el usuario que accede al servidor LDAP.
  - Para **OCSP**, especifique el **URL del programa de respuesta OCSP**. Este URL es el URL del programa de respuesta OCSP (respondedor OCSP) que se utiliza para comprobar la revocación de certificados. Este valor debe ser un URL de HTTP que contenga el nombre de host y el número de puerto del respondedor OCSP. Si el programa de respuesta OCSP está utilizando el puerto 80, que es el valor predeterminado para HTTP, entonces el número de puerto se puede omitir. Los URL de HTTP se definen en la RFC 1738.
  - Para **IDPW OS**, no hay requisitos adicionales aunque puede especificar opciones adicionales para este tipo de autenticación.
  - Para **IDPW LDAP**, especifique el **nombre de servidor LDAP** y el nombre **Usuario abreviado** . El nombre del servidor LDAP es el nombre de host, la dirección IPv4 en notación decimal con puntos o IPv6 en notación hexadecimal del host en el que se está ejecutando el servidor LDAP, con un número de puerto opcional. El nombre de usuario abreviado es el campo del registro de usuario de LDAP que se utiliza como nombre abreviado para la conexión. Opcionalmente, puede especificar más opciones para este tipo de autenticación.

e) Pulse **Añadir**.

- Para suprimir un objeto de información de autenticación:

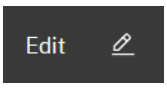
a) Seleccione el icono de llave  para el objeto de información de autenticación que desea suprimir de la lista.

b) En la vista de propiedades de objeto, pulse **Suprimir objeto de información de autenticación**.

c) Confirme que desea suprimir el objeto de información de autenticación pulsando **Suprimir**. Se suprime el objeto.

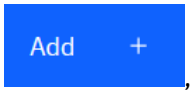
- Para ver y editar las propiedades de un objeto de información de autenticación:

a) Seleccione el icono de llave  para el objeto de información de autenticación que desea ver en la lista.

b) Para editar las propiedades visualizadas, pulse el botón Editar 

c) Edite las propiedades que sean necesarias. Si el recuadro de texto de propiedad está inhabilitado, la propiedad es de sólo lectura o solo se puede editar desde la línea de mandatos.

d) Pulse **Guardar** para guardar los cambios.

- Para ver y editar los registros de autorizaciones de un objeto de información de autenticación:
  a) Seleccione el icono de llave  para el objeto de información de autenticación para el que desea ver el registro de autorización de la lista.
  b) Seleccione el separador **Seguridad**.
  c) Para editar o suprimir un registro de autorización existente, seleccione **Editar** o **Suprimir** en el menú .
  d) Para añadir un nuevo registro de autorización, pulse el botón **Añadir** 

, proporcione los detalles del nuevo registro de autorizaciones y pulse **Crear**.

V 9.4.0 *IBM MQ Console: Trabajar con registros de autorización de gestor de colas*

Puede controlar el acceso que los usuarios y grupos tienen a los gestores de colas especificando un registro de autorizaciones para dicho usuario o grupo.

Acerca de esta tarea

Puede ajustar el acceso que un usuario o grupo de usuarios de mensajería tiene a un determinado gestor de colas utilizando los registros de autorizaciones. Existen dos tipos de registros de autorización: los registros de **acceso de gestor de colas** que controlan las autorizaciones generales y el permiso **para crear** registros que controlan qué usuarios y grupos pueden crear objetos para el gestor de colas.

Procedimiento

- Para ver los registros de autorizaciones de un gestor de colas:
  a) Asegúrese de que el gestor de colas está en ejecución, y selecciónelo en la lista del gestor de colas.
  b) Seleccione **Ver configuración** en el menú .
  c) Asegúrese de que la pestaña **Seguridad** esté seleccionada.
  d) Seleccione **Registros de autorizaciones** en el panel de navegación. La vista muestra los registros de autorizaciones en dos paneles, lo que le permite trabajar con registros de autorizaciones generales y con registros de autorizaciones de creación.
- Para añadir un registro de autorizaciones general:
  a) Pulse el botón Añadir 

en la vista de lista **Acceso al gestor de colas** .
  b) Elija si va a añadir un registro de autorizaciones para un usuario o grupo.
  c) Especifique el nombre del usuario o grupo para el que está añadiendo un registro de autorización (el registro de autorización lo toma como su nombre).
  d) Seleccione las autorizaciones que desea otorgar.
  e) Pulse **Crear**.
- Para añadir un registro de autorizaciones de creación:
  a) Pulse el botón Añadir 

en la vista de lista **Permiso para crear** .
  b) Elija si va a añadir un registro de autorizaciones para un usuario o grupo.
  c) Especifique el nombre del usuario o grupo para el que está añadiendo un registro de autorización (el registro de autorización lo toma como su nombre).

d) Seleccione los tipos de objeto a los que va a otorgar la autorización de creación.

e) Pulse **Crear**.

- Para suprimir un registro de autorizaciones:
  a) Seleccione el registro de autorización que desea suprimir y seleccione **Suprimir**.
  b) Confirme que desea suprimir el objeto de información de autenticación pulsando **Suprimir**. Se suprime el objeto.
- Para ver y editar las propiedades de un registro de autorizaciones:
  a) Pulse el registro de autorización que desea ver.
  b) Cambie los valores según sea necesario y pulse **Guardar** para guardar los cambios.

V 9.4.0 *IBM MQ Console: Trabajar con registros de autenticación de canal*

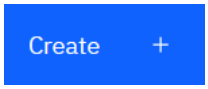
Puede utilizar IBM MQ Console para añadir y suprimir registros de autenticación de canal en un gestor de colas. También puede ver y establecer las propiedades de los registros de autenticación de canal.

Acerca de esta tarea

Para ejercer un control más preciso sobre el acceso que se otorga a los sistemas de conexión en un nivel de canal, puede utilizar registros de autenticación de canal.

Para garantizar la seguridad, puede utilizar registros de autenticación de canal de bloqueo para bloquear el acceso a los canales. También puede utilizar registros de autenticación de canal de correlación de direcciones para permitir el acceso a usuarios especificados. Para obtener más información sobre los registros de autenticación de canal, consulte Registros de autenticación de canal

Procedimiento

- Para ver la información de autenticación de canal para un gestor de colas:
  a) Asegúrese de que el gestor de colas está en ejecución, y selecciónelo en la lista del gestor de colas.
  b) Seleccione **Ver configuración** en el menú ⋮.
  c) Asegúrese de que la pestaña **Seguridad** esté seleccionada.
  d) Seleccione **Autenticación de canal** en el panel de navegación.
- Para añadir un registro de autenticación de canal:
  a) Pulse el botón Crear 

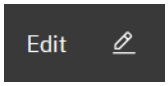
en la vista de lista de información de autenticación de canal.
  b) Elija el tipo de regla que desea utilizar. Seleccione una **Allow**, **Block**o **Warn**.
  c) Elija el tipo de identidad para el que está configurando una regla de autenticación de canal. Hay diferentes tipos de identidad disponibles, en función del tipo de regla que haya seleccionado.
  d) Proporcione la información necesaria para la identidad que está especificando. De forma predeterminada, se muestran las propiedades mínimas recomendadas para que pueda proporcionar valores. Puede ver todas las propiedades disponibles seleccionando **Crear personalizado**.
  e) Pulse **Crear** para crear el registro de autenticación de canal.

  Para obtener más información sobre los valores disponibles para los registros de autenticación de canal, consulte Registros de autenticación de canal y SET CHLAUTH
- Para suprimir un registro de autenticación de canal:
  a) Pulse el icono de llave  junto al registro de autenticación de canal que desea suprimir.
  b) En la vista Editar autenticación de canal, pulse **Suprimir objeto de autenticación de canal**.

c) Confirme que desea suprimir el registro de autenticación de canal pulsando **Suprimir**. El registro de autenticación de canal se suprime.

- Para ver y editar las propiedades de un registro de autenticación de canal:

  a) Pulse el icono de llave  junto al registro de autenticación de canal que desea editar o visualizar. Se muestran las propiedades.

  b) Pulse el botón Editar 

  c) Edite las propiedades que sean necesarias. Si el recuadro de texto de propiedad está inhabilitado, la propiedad es de sólo lectura o solo se puede editar desde la línea de mandatos.

  d) Pulse **Guardar** para guardar los cambios.

Multi V 9.4.0 *IBM MQ Console: Trabajar con escuchas*

Puede utilizar IBM MQ Console para añadir y suprimir escuchas, iniciar y detener escuchas, ver y establecer propiedades del escucha y gestionar los registros de autorizaciones de un escucha.

Acerca de esta tarea

La vista de escuchas enumera los escuchas existentes para un gestor de colas específico. Puede seleccionar escuchas individuales con los que desee trabajar.

Procedimiento

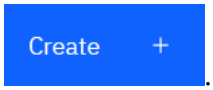
- Para ver los escuchas de un gestor de colas:

  a) Asegúrese de que el gestor de colas está en ejecución, y selecciónelo en la lista del gestor de colas.

  b) Seleccione **Ver configuración** en el menú .

  c) Seleccione la pestaña **Escuchas** .

- Para crear un escucha:

  a) Pulse el botón Crear 

.

  b) Proporcione la información necesaria para el escucha que está creando.

  c) Pulse **Crear**. Se crea el nuevo escucha.

- Para iniciar un escucha:

  a) Localice el escucha que desea iniciar en la lista.

  b) Seleccione **Iniciar** en el menú .

- Para detener un escucha:

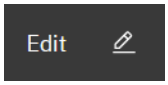
  a) Localice el escucha que desea iniciar en la lista.

  b) Seleccione **Detener** en el menú .

- Para ver y editar las propiedades de un escucha:

  a) Localice el escucha en la lista.

  b) Seleccione **Ver configuración** en el menú .

c) Asegúrese de que la pestaña **Propiedades** está seleccionada. Para editar las propiedades, pulse el botón Editar 

d) Edite las propiedades que sean necesarias. Si el recuadro de texto de propiedad está inhabilitado, la propiedad es de sólo lectura o solo se puede editar desde la línea de mandatos. Para obtener más información sobre las propiedades, consulte Propiedades de escucha en la documentación de MQ Explorer.

e) Pulse **Guardar** para guardar los cambios.

- Para ver y editar registros de autorizaciones para una escucha:
  a) Localice el escucha en la lista.
  b) Seleccione **Ver configuración** en el menú ⋮.
  c) Pulse la pestaña **Seguridad**.
  d) Trabaje con los registros de autorizaciones tal como se describe para los registros de autorizaciones de gestor de colas. Consulte "IBM MQ Console: Trabajar con registros de autorización de gestor de colas" en la página 104.
- Para suprimir un escucha:
  a) Localice el escucha en la lista.
  b) Seleccione **Ver configuración** en el menú ⋮.
  c) Pulse **Suprimir escucha**.

V 9.4.0 IBM MQ Console: Adición de un gestor de colas remoto

Puede utilizar IBM MQ Console para administrar un gestor de colas que se ejecute en un sistema remoto.

Antes de empezar

- Debe preparar el gestor de colas en el sistema remoto para que se pueda administrar de forma remota, consulte el paso "1" en la página 109, "2" en la página 109, "3" en la página 110y "4" en la página 110 de "Adición de un gestor de colas remoto al IBM MQ Console utilizando la línea de mandatos" en la página 109.
- También debe habilitar las conexiones remotas desde IBM MQ Console. Para obtener más información, consulte Configuración del comportamiento de conexión del gestor de colas remoto.

Acerca de esta tarea

Utilice una tabla de definición de conexión de cliente (CCDT) en formato JSON para especificar los detalles de conexión remota. Puede crear una CCDT JSON utilizando un editor de texto (consulte el paso "5" en la página 110 de "Adición de un gestor de colas remoto al IBM MQ Console utilizando la línea de mandatos" en la página 109) o puede crear una utilizando IBM MQ Console.

De forma alternativa, puede crear la CCDT desde IBM MQ Console especificando los detalles de conexión directamente al añadir el gestor de colas remoto.

También puede conectar un gestor de colas remoto a IBM MQ Console utilizando la línea de mandatos para todas las tareas necesarias (además de preparar el gestor de colas remoto y crear una CCDT). Consulte "Adición de un gestor de colas remoto al IBM MQ Console utilizando la línea de mandatos" en la página 109.

Atención: si recibe los mensajes siguientes:

```
MQWB2026E: The request to connect to the remote queue manager 'rqmgr-qmgr_name' failed
with the error message:
```

```
'JMSCC0051: The property 'JMS_IBM_MQMD_AccountingToken' should be set using type '[B',
not 'java.lang.Object'.'
```

está intentando pasar un `java.lang.Object` a la señal de contabilidad, cuando se espera un tipo de objeto java `byte[]` .

Procedimiento

- Para añadir un gestor de colas remoto, especifique una CCDT existente:
  a) En la página de inicio, pulse **Conectar gestor de colas remoto**.
  b) Especifique el nombre del gestor de colas remoto.
  c) Opcionalmente, especifique un nombre exclusivo para el gestor de colas. Si no especifica un nombre exclusivo, se utiliza el nombre real con el prefijo "remote-" añadido.
  d) Asegúrese de que **Conectar utilizando una CCDT JSON** esté seleccionado.
  e) Pulse **Examinar** y seleccione el archivo que contiene la CCDT JSON que desea utilizar.
  f) Pulse **Siguiente** para desplazarse a la página de usuario y, opcionalmente, especificar un nombre de usuario y una contraseña para conectarse al gestor de colas remoto. Si no especifica esta información, la información de autenticación se toma del archivo de configuración de conexión remota.
  g) Pulse **Siguiente** para desplazarse a la página Certificado. Si la CCDT especifica información de "transmissionSecurity", se utiliza esta información. Opcionalmente, puede pegar un certificado (como una clave pública codificada en base64) y esto se añade al almacén de confianza global.

     El certificado se almacena temporalmente en *WLP_USER_DIR*/`generated.crts/`*uniqueName*-*qmgrName*`.crt` antes de que se añada al almacén de confianza. Cuando se añade correctamente la conexión, el certificado se suprime de esta ubicación.
  h) Pulse **Siguiente** para ver la página de resumen. Puede utilizar el botón **Atrás** para revisar páginas anteriores y realizar correcciones. Si está satisfecho con la información, pulse **Conectar** para conectarse al gestor de colas remoto.
- Para añadir un gestor de colas remoto y especificar la información de conexión manualmente:
  a) En la página de inicio, pulse **Conectar gestor de colas remoto**.
  b) Especifique el nombre del gestor de colas remoto.
  c) Opcionalmente, especifique un nombre exclusivo para el gestor de colas. Si no especifica un nombre exclusivo, se utiliza el nombre real con el prefijo "remote-" añadido.
  d) Seleccione **Entrada manual**.
  e) Especifique el nombre del canal de conexión de cliente que utilizará la conexión.
  f) Especifique el nombre del host donde se ejecuta el gestor de colas remoto. Si se detectan instalaciones de MQ remotas, se muestran los nombres de host y puede seleccionar el host del gestor de colas remoto al que desea conectarse. En algunas configuraciones de red, no es posible detectar instancias de MQ remotas. En este caso, añada el nombre de host y el puerto manualmente.
  g) Pulse **Siguiente** para desplazarse a la página de usuario y, opcionalmente, especificar un nombre de usuario y una contraseña para conectarse al gestor de colas remoto. Si no especifica esta información, la información de autenticación se toma del archivo de configuración de conexión remota.
  h) Pulse **Siguiente** para desplazarse a la página Certificado. Puede seleccionar una CipherSpec SSL en la lista desplegable. Opcionalmente, puede pegar un certificado (como una clave pública codificada en base64) y esto se añade al almacén de confianza global.

     El certificado se almacena temporalmente en *WLP_USER_DIR*/`generated.crts/`*uniqueName*-*qmgrName*`.crt` antes de que se añada al almacén de confianza. Cuando se añade correctamente la conexión, el certificado se suprime de esta ubicación.

i) Pulse **Siguiente** para ver la página de resumen. Puede utilizar el botón **Atrás** para revisar páginas anteriores y realizar correcciones. Si está satisfecho con la información, pulse **Conectar** para conectarse al gestor de colas remoto.

La información de conexión que ha especificado se graba en el archivo CCDT en el directorio web. La vía de acceso es *WLP_USER_DIR*/generated.ccdt/ccdt-*uniqueName*.

Resultados

El gestor de colas remoto aparece en la lista de gestores de colas remotos en IBM MQ Console. Siempre que la conexión sea correcta, puede administrar los objetos del gestor de colas remoto de la misma forma que trabaja con los objetos de un gestor de colas local.

V 9.4.0 *Adición de un gestor de colas remoto al IBM MQ Console utilizando la línea de mandatos*

Puede añadir un gestor de colas remoto al IBM MQ Console utilizando el mandato **setmqweb remote** en la línea de mandatos. Un gestor de colas remoto puede ser un gestor de colas que se ejecuta en una instalación distinta en el mismo sistema que IBM MQ Console o un gestor de colas que se ejecuta en un sistema distinto.

Antes de empezar

Nota: Los pasos de esta tarea requieren que ejecute mandatos MQSC:

- ALW EnAIX, Linux, and Windows , emite comandos MQSC desde un**runmqsc** símbolo del sistema. Ver Ejecutar comandos MQSC de forma interactiva en**runmqsc** y Ejecutar comandos MQSC desde archivos de texto en**runmqsc** . Para esta tarea, si está ejecutando en AIX, Linux, and Windows, abra un indicador de mandatos runmqsc que utilice QM1:

  ```
  runmqsc QM1
  ```

- IBM i EnIBM i , crea una lista de comandos en un archivo de secuencia de comandos y luego ejecuta el archivo usando el**STRMQMMQSC** dominio. Ver Administración mediante comandos MQSC enIBM i .
- z/OS Enz/OS , los comandos MQSC se pueden emitir desde varias fuentes, dependiendo del comando. Ver Fuentes desde las que puede emitir comandos MQSC y PCF enIBM MQ for z/OS .

Asegúrese de que el servidor mqweb está configurado para permitir conexiones de gestor de colas remoto a IBM MQ Console. Para obtener más información, consulte Configuración del comportamiento de conexiones del gestor de colas.

Procedimiento

1. En el gestor de colas remoto, cree un canal de conexión de servidor para permitir la administración remota del gestor de colas utilizando el mandato MQSC de **DEFINE CHANNEL** .
   Por ejemplo, para crear un canal de conexión de servidor QM1.SVRCONN para el gestor de colas QM1, especifique el siguiente mandato MQSC:

   ```
   DEFINE CHANNEL(QM1.SVRCONN) CHLTYPE(SVRCONN) TRPTYPE(TCP)
   ```

   Para obtener más información sobre **DEFINE CHANNEL** y las opciones disponibles, consulte DEFINE CHANNEL.
2. Asegúrese de que un usuario adecuado esté autorizado para administrar el gestor de colas y los objetos de MQ asociados con el gestor de colas.
   - ALW En AIX, Linux, and Windows , utilice el mandato de control **setmqaut** en una línea de mandatos estándar.

- z/OS En z/OS, defina perfiles RACF para otorgar al usuario autorizado acceso al gestor de colas.

Por ejemplo, en AIX, Linux, and Windows, para autorizar al usuario `exampleUser` a acceder al gestor de colas QM1, especifique el mandato de control siguiente:

```
setmqaut -m QM1 -t qmgr -p exampleUser +connect +inq +setall +dsp
```

Este usuario autorizado puede ser uno de los siguientes usuarios:

- Un ID de usuario que es el mismo que el ID de usuario que inicia el servidor mqweb que ejecuta el IBM MQ Console en el sistema desde el que desea administrar de forma remota este gestor de colas.
- Un ID de usuario que coincide con un ID de usuario y una contraseña que se incluyen en el mandato **setmqweb remote** en el paso "7" en la página 111. Al incluir el ID de usuario y la contraseña en el mandato **setmqweb remote** , este ID de usuario y esta contraseña se utilizan para la autenticación cuando el IBM MQ Console se conecta al gestor de colas.
- ID de usuario determinado por las reglas de seguridad de canal. Por ejemplo, puede establecer una regla de autenticación de canal en el canal de conexión de servidor para permitir conexiones desde la dirección IP desde la que utiliza IBM MQ Console para la administración remota y correlacionar todas estas conexiones con un ID de usuario específico que esté autorizado para utilizar el gestor de colas. Para obtener más información, consulte Creación de nuevas reglas CHLAUTH para canales.

3. ALW
Si no hay ningún escucha en ejecución en el gestor de colas remoto, cree un escucha para aceptar conexiones de red entrantes utilizando el mandato MQSC de **DEFINE LISTENER** .
Por ejemplo, para crear un escucha `REMOTE.LISTENER` en el puerto 1414 para el gestor de colas remoto QM1, especifique el siguiente mandato MQSC:

```
runmqsc QM1
DEFINE LISTENER(REMOTE.LISTENER) TRPTYPE(TCP) PORT(1414)
end
```

4. Asegúrese de que el escucha se está ejecutando utilizando el mandato MQSC de **START LISTENER** .

ALW Por ejemplo, en AIX, Linux, and Windows para iniciar el escucha `REMOTE.LISTENER` para el gestor de colas QM1, especifique el siguiente mandato MQSC:

```
runmqsc QM1
START LISTENER(REMOTE.LISTENER)
end
```

z/OS Por ejemplo, en z/OS, para iniciar el escucha, especifique el siguiente mandato MQSC:

```
/cpf START LISTENER TRPTYPE(TCP) PORT(1414)
```

Tenga en cuenta que el espacio de direcciones del iniciador de canal debe estar iniciado para poder iniciar un escucha en z/OS.

5. Cree un archivo CCDT JSON que contenga la información de conexión del gestor de colas remoto:

- Genere un archivo CCDT utilizando el IBM MQ Console que está asociado con la misma instalación que el gestor de colas al que desea conectarse de forma remota.

  En el panel **Inicio**, pulse el mosaico **Descargar archivo de conexión** .
- Cree un archivo CCDT de formato JSON que defina la conexión. Para obtener más información sobre la creación de una CCDT de formato JSON, consulte Configuración de una CCDT de formato JSON.

  El archivo CCDT debe incluir la información de `name`, `clientConnection` y `type` . Opcionalmente, puede incluir información adicional como, por ejemplo, la información de

`transmissionSecurity`. Para obtener más información sobre todas las definiciones de atributos de canal de CCDT, consulte la Lista completa de definiciones de atributos de canal CCDT.

El ejemplo siguiente muestra un archivo CCDT JSON básico para una conexión de gestor de colas remoto. Establece el nombre del canal en el mismo nombre que el canal de conexión de servidor de ejemplo creado en el paso “1” en la página 109 y el puerto de conexión con el mismo valor que el puerto que utiliza el escucha. El host de conexión se establece en el nombre de host del sistema en el que se ejecuta el gestor de colas remoto de ejemplo, QM1:

```
{
  "channel": [{
      "name": "QM1.SVRCONN",
      "clientConnection": {
        "connection": [{
            "host": "example.com",
            "port": 1414
          }],
        "queueManager": "QM1"
      },
      "type": "clientConnection"
    }]
}
```

6. Copie el archivo CCDT JSON en el sistema donde se ejecuta IBM MQ Console .
7. En la instalación que ejecuta IBM MQ Console, utilice el mandato **setmqweb remote** para añadir la información del gestor de colas remoto a la configuración de IBM MQ Console .

   Como mínimo, para añadir un gestor de colas remoto a IBM MQ Console debe proporcionar el nombre del gestor de colas, un nombre exclusivo para el gestor de colas (para diferenciar entre otros gestores de colas remotos que pueden tener el mismo nombre de gestor de colas) y el URL de CCDT para el gestor de colas. El nombre exclusivo es el nombre de visualización en IBM MQ Console, por lo tanto, especifique un nombre que deje claro que se trata de un gestor de colas remoto, por ejemplo, `"remote-QM2"`. Puede especificar varias opciones adicionales, como el nombre de usuario y la contraseña que se deben utilizar para la conexión del gestor de colas remoto, o los detalles del almacén de confianza y del almacén de claves. Para obtener una lista completa de los parámetros que se pueden especificar con el mandato **setmqweb remote** , consulte setmqweb remote.

   Por ejemplo, para añadir el gestor de colas remoto de ejemplo QM1, utilizando el archivo CCDT de ejemplo, especifique el mandato siguiente:

```
setmqweb remote add -uniqueName "MACHINEAQM1" -qmgrName "QM1" -ccdtURL "c:\myccdts\ccdt.json"
```

Resultados

El gestor de colas remoto aparece en la lista de gestores de colas remotos en la IBM MQ Console cuando se renueva la siguiente lista de conexiones remotas. Siempre que la conexión sea correcta, puede administrar los objetos del gestor de colas remoto de la misma forma que trabaja con los objetos de un gestor de colas local.

Ejemplo

El ejemplo siguiente configura la conexión del gestor de colas remoto para un gestor de colas QM1. El IBM MQ Console está autorizado para administrar el gestor de colas basándose en la autorización otorgada al usuario `exampleUser`. Las credenciales de este usuario se proporcionan al IBM MQ Console cuando se utiliza el mandato **setmqweb remote** para configurar la información de conexión del gestor de colas remoto.

1. En el sistema donde está el gestor de colas remoto QM1 , se crean un canal de conexión de servidor y un escucha. Se inicia el escucha y se otorga autorización al usuario `exampleUser` para administrar el gestor de colas. Por ejemplo, en AIX, Linux, and Windows, ejecute los mandatos siguientes:

```
runmqsc QM1
#Define the server connection channel that will accept connections from the Console
DEFINE CHANNEL(QM1.SVRCONN) CHLTYPE(SVRCONN) TRPTYPE(TCP)
# Define the listener to use for the connection from the Console
DEFINE LISTENER(REMOTE.LISTENER) TRPTYPE(TCP) PORT(1414)
```

```
# Start the listener
START LISTENER(REMOTE.LISTENER)
end

#Set mq authorization for exampleUser to access the queue manager
setmqaut -m QM1 -t qmgr -p exampleUser +connect +inq +setall +dsp
```

2. En el sistema donde se ejecuta IBM MQ Console , se crea un archivo `QM1_ccdt.json` con la siguiente información de conexión:

```
{
  "channel": [{
      "name": "QM1.SVRCONN",
      "clientConnection": {
        "connection": [{
            "host": "example.com",
            "port": 1414
          }],
        "queueManager": "QM1"
      },
      "type": "clientConnection"
    }]
}
```

3. En el sistema donde se ejecuta IBM MQ Console , la información de conexión del gestor de colas remoto para el gestor de colas QM1 se añade al servidor mqweb. Las credenciales para `exampleUser` se incluyen en la información de conexión:

```
setmqweb remote add -uniqueName "remote-QM1" -qmgrName "QM1" -ccdtURL
"c:\myccdts\QM1_ccdt.json" -username "exampleUser" -password "password"
```

4. La IBM MQ Console muestra el gestor de colas remoto QM1.

V 9.4.0 IBM MQ Console: Trabajar con objetos

Cada gestor de colas de IBM MQ tiene varios tipos distintos de objetos asociados.

Acerca de esta tarea

Puede utilizar la consola para trabajar con los distintos tipos de objeto de IBM MQ:

- Colas
- Objetos de sucesos:
  - Temas
  - Suscripciones
- Objetos de aplicaciones:
  - Conexiones
  - Canales de aplicación
  - Instancias de canal de aplicación
- Objetos de red de MQ :
  - Gestores de colas conectados
  - Canales de gestores de colas
  - Instancias de canal de gestor de colas

Procedimiento

Para trabajar con un objeto de IBM MQ:

1. En la vista de lista de gestores de colas, pulse en el gestor de colas propietario de los objetos con los que desea trabajar.

2. Pulse en la pestaña de red Colas, Sucesos, Aplicaciones o MQ para seleccionar el tipo de objeto con el que desea trabajar.
3. Consulte uno de los temas siguientes para obtener instrucciones detalladas sobre cómo trabajar con los objetos.

V 9.4.0 *IBM MQ Console: Trabajar con colas*

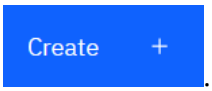
Puede ver las colas existentes para un gestor de colas específico en la pestaña **Colas**. Puede añadir y suprimir colas, añadir y borrar mensajes en una cola, examinar mensajes, ver y establecer las propiedades de una cola y gestionar los registros de autorizaciones de una cola.

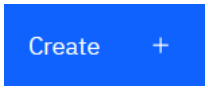
Acerca de esta tarea

La vista de colas enumera las colas existentes para un gestor de colas específico. Puede acceder a la lista de colas pulsando en un gestor de colas y seleccionando la pestaña **Colas**. Puede seleccionar colas individuales de la lista con las que trabajar.

z/OS No puede ver ni editar registros de autorizaciones de colas en z/OS.

Procedimiento

- Para añadir una cola:
  a) En la pestaña **Colas**, pulse el botón Crear 

.
  b) Seleccione el tipo de cola que desea crear:
    – Cola local: los mensajes se almacenan en el gestor de colas al que pertenecen.
    – Cola alias: puntero a otra cola en el mismo gestor de colas.
    – Cola remota: puntero a otra cola en un gestor de colas distinto.
    – Cola modelo: plantilla de una cola utilizada cuando se crea un gestor de colas dinámico.
  c) Proporcione la información necesaria para el tipo de cola que está creando. De forma predeterminada, se muestran las propiedades mínimas recomendadas para que pueda proporcionar valores. Puede ver todas las propiedades disponibles seleccionando **Crear personalizado**.
  d) Pulse **Crear**. Se crea la nueva cola.
- Para poner mensajes en una cola:
  a) Pulse la cola a la que desea añadir mensajes en la lista de la vista de lista de colas. No puede seleccionar una cola de modelo.
  b) Pulse el botón Crear 

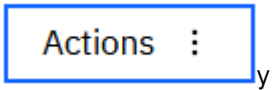
  c) Especifique el mensaje que desea transferir a la cola.
  d) Pulse **Crear**.
- Para borrar mensajes de una cola:
  a) Pulse la cola local de la que desea borrar los mensajes en la lista de colas.
  b) Pulse el icono Borrar cola 
  c) Confirme que desea borrar la cola pulsando **Borrar cola**.
- V 9.4.0
  Para suprimir un mensaje individual de una cola:
  a) Localice el mensaje que desea suprimir.

b) Pulse el icono de supresión situado junto al mensaje  .

c) Confirme que desea borrar el mensaje pulsando **Suprimir**.

- Para examinar los mensajes de una cola, pulse en la cola en la vista de lista de colas. Se muestra una lista de los mensajes en esa cola.
- Para suprimir una cola:

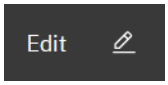
  a) Pulse en la cola local que desea suprimir en la lista de colas.

  b) Pulse el botón Acciones 

y seleccione **Suprimir cola**.

  c) Confirme que desea suprimir la cola pulsando **Suprimir**. Se suprime la cola.
- Para ver y editar las propiedades de una cola:

  a) Seleccione **Ver configuración** en el menú junto a la cola que desea editar.

  b) Pulse el botón Editar 

  c) Edite las propiedades que sean necesarias. Si el recuadro de texto de propiedad está inhabilitado, la propiedad es de sólo lectura o solo se puede editar desde la línea de mandatos. Para obtener información sobre las propiedades, consulte Propiedades de cola en la documentación de IBM MQ Explorer

  d) Pulse **Guardar** para guardar los cambios.
- Para ver y editar registros de autorizaciones para una cola:

  a) Seleccione **Ver configuración** en el menú junto a la cola que desea editar.

  b) Pulse la pestaña **Seguridad**.

  c) Trabaje con los registros de autorizaciones tal como se describe para los registros de autorizaciones de gestor de colas. Consulte "IBM MQ Console: Trabajar con registros de autorización de gestor de colas" en la página 104.
- V 9.4.0

  Para ver los objetos de IBM MQ asociados a una cola:

  a) Seleccione **Ver objetos asociados** en el menú junto a la cola que desea ver.

  b) Ver los objetos en el panel que aparece. Pulse los enlaces para ver más detalles sobre cada uno de los objetos listados.

  Puede utilizar el panel para ver qué aplicaciones están colocando mensajes en colas y ver las relaciones entre distintas colas. Esto puede ayudarle a identificar y resolver problemas.

V 9.4.0 *IBM MQ Console: Trabajar con temas*

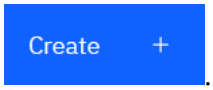
Puede utilizar IBM MQ Console para añadir y suprimir temas y para ver y establecer las propiedades de un tema.

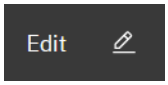
Acerca de esta tarea

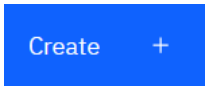
La vista de temas enumera los temas existentes para un gestor de colas específico. Puede acceder a los temas desde la pestaña **Sucesos** del gestor de colas. Puede seleccionar temas individuales de la lista con los que trabajar.

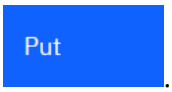
z/OS No puede ver ni editar registros de autorizaciones de un tema en z/OS.

Procedimiento

- Para añadir un tema:
  a) En la vista del gestor de colas, abra el separador **Sucesos** y pulse **Temas**.
  b) Pulse el botón Crear 

.
  c) Proporcione la información necesaria para el tema que está creando. De forma predeterminada, se muestran las propiedades mínimas recomendadas para que pueda proporcionar valores. Puede ver todas las propiedades disponibles seleccionando **Crear personalizado**.
  d) Pulse **Crear**. Se crea el nuevo tema.
- Para suprimir un tema:
  a) Pulse el icono de llave  junto al tema que desea suprimir.
  b) En la vista Editar cola, pulse **Suprimir tema**.
  c) Confirme que desea suprimir el tema pulsando **Suprimir**. Se suprime el tema.
- Para ver y editar las propiedades de un tema:
  a) Pulse el icono de llave  junto al tema que desea editar.
  b) Pulse el botón Editar 

  c) Edite las propiedades que sean necesarias. Si el recuadro de texto de propiedad está inhabilitado, la propiedad es de sólo lectura o solo se puede editar desde la línea de mandatos. Para obtener más información sobre las propiedades, consulte Propiedades de tema en la documentación de MQ Explorer.
  d) Pulse **Guardar** para guardar los cambios.
- Para publicar un mensaje en un tema, debe tener al menos una suscripción coincidente.
  a) Pulse el tema en el que desea publicar en la lista de temas.
  b) Pulse el nombre de suscripción coincidente.
  c) Pulse el botón Crear 

  d) Escriba el mensaje que desea publicar.
  e) Pulse botón Transferir 

. El mensaje se escribe en todas las suscripciones coincidentes.
- Para suscribirse a un tema, consulte “IBM MQ Console: Trabajar con suscripciones” en la página 115:
- Para ver y editar registros de autorizaciones para un tema:
  a) Pulse el icono de llave  junto al tema para el que desea editar los registros de autorizaciones.
  b) Pulse la pestaña **Seguridad**.
  c) Trabaje con los registros de autorizaciones tal como se describe para los registros de autorizaciones del gestor de colas, consulte “IBM MQ Console: Trabajar con registros de autorización de gestor de colas” en la página 104.

V 9.4.0 *IBM MQ Console: Trabajar con suscripciones*

Puede utilizar IBM MQ Console para añadir y suprimir suscripciones y ver y establecer las propiedades de una suscripción.

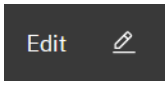
Acerca de esta tarea

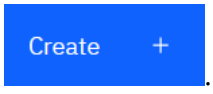
La vista de suscripciones lista las suscripciones existentes para un gestor de colas específico. Puede acceder a las suscripciones desde la pestaña **Sucesos** del gestor de colas. Puede seleccionar temas individuales de la lista con los que trabajar. Puede seleccionar suscripciones individuales en la lista para trabajar con ellas.

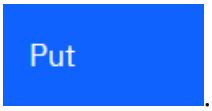
Para obtener más información sobre las suscripciones, consulte Suscriptores y suscripciones y DEFINE SUB.

z/OS No puede ver ni editar registros de autorizaciones para una suscripción en z/OS.

Procedimiento

- Para añadir una suscripción:
  a) En la vista del gestor de colas, abra el separador **Sucesos** y pulse **Suscripciones**.
  b) Elija si desea crear una suscripción gestionada o no gestionada.
  c) Proporcione la información necesaria para la suscripción que está creando. De forma predeterminada, se muestran las propiedades mínimas recomendadas para que pueda proporcionar valores. Puede ver todas las propiedades disponibles seleccionando **Crear personalizado**.
  d) Pulse **Crear**. Se crea la nueva suscripción.
- Para suprimir una suscripción:
  a) Pulse el icono de llave  junto a la suscripción que desea suprimir.
  b) En la vista Editar cola, pulse **Suprimir suscripción**.
  c) Confirme que desea suprimir la suscripción pulsando **Suprimir**. La suscripción se suprime.
- Para ver y editar las propiedades de una suscripción:
  a) Pulse el icono de llave  junto a la suscripción que desea editar.
  b) Pulse el botón Editar 

  c) Edite las propiedades que sean necesarias. Si el recuadro de texto de propiedad está inhabilitado, la propiedad es de sólo lectura o solo se puede editar desde la línea de mandatos.
  d) Pulse **Guardar** para guardar los cambios.
- Para publicar un mensaje en el tema al que está suscrita la suscripción:
  a) Pulse la suscripción en cuyo tema desea publicar en la lista de suscripciones.
  b) Pulse el botón Crear 

  .
  c) Escriba el mensaje que desea publicar.
  d) Pulse botón Transferir 

  . El mensaje se escribe en todas las suscripciones que coinciden con el tema en el que ha publicado.

V 9.4.0 *IBM MQ Console: Trabajar con canales de gestor de colas*

Puede utilizar IBM MQ Console para trabajar con canales de gestor de colas: puede añadir y suprimir canales de gestor de colas, iniciar y detener canales, restablecer y resolver canales y hacer ping a

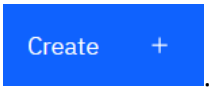
canales. También puede ver y establecer las propiedades de un canal de gestor de colas y gestionar los registros de autorizaciones del canal.

Acerca de esta tarea

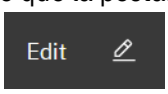
Un canal de gestor de colas es un enlace de comunicación lógica para transmitir mensajes entre gestores de colas a través de una red. La vista de canales de gestor de colas incluye un panel que muestra una vista rápida de cuántos canales se están ejecutando, cuándo lo están volviendo a intentar y cuántos se han detenido.

z/OS No puede ver ni editar registros de autorizaciones de un canal en z/OS.

Procedimiento

- Para añadir un canal de gestor de colas:
  a) En la vista de gestor de colas, abra el separador **Red deMQ** y pulse **Canales de gestor de colas** y pulse el botón Crear 

.
  b) Seleccione el tipo de canal de gestor de colas que desea crear y pulse el botón siguiente 

.
  c) Proporcione la información necesaria para el canal que va a crear. De forma predeterminada, se muestran las propiedades mínimas recomendadas para que pueda proporcionar valores. Puede ver todas las propiedades disponibles seleccionando **Crear personalizado**.
  d) Pulse **Crear**. Se creará el nuevo canal con el estado **inactivo**.
- Para iniciar un canal de gestor de colas:
  a) Localice el canal que desea iniciar en la lista.
  b) Seleccione **Iniciar** en el menú ⋮.
- Para detener un canal de gestor de colas:
  a) Localice el canal que desea detener en la lista.
  b) Seleccione **Detener** en el menú ⋮.
- Para ver las propiedades de un canal de gestor de colas:
  a) Localice el canal en la lista.
  b) Seleccione **Ver configuraciones** en el menú ⋮.
  c) Asegúrese de que la pestaña **Propiedades** está seleccionada. Para editar las propiedades, pulse el botón Editar 

  d) Edite las propiedades que sean necesarias. Si el recuadro de texto de propiedad está inhabilitado, la propiedad es de sólo lectura o solo se puede editar desde la línea de mandatos. Para obtener información sobre las propiedades, consulte Propiedades de canal en la documentación de MQ Explorer.
  e) Pulse **Guardar** para guardar los cambios.
- Para restablecer un canal de gestor de colas:
  a) Localice el canal en la lista.

b) Seleccione **Avanzada** en el menú .

c) En la sección **Restablecer**, especifique un número de secuencia de mensaje.

Debe restablecer un canal si no se inicia porque los dos extremos discrepan sobre el número de secuencia del siguiente mensaje a enviar. El número de secuencia de mensaje especifica dicho número.

d) Pulse **Restablecer canal**.

- Para resolver un canal emisor o servidor:

a) Localice el canal en la lista.

b) Seleccione **Avanzada** en el menú .

c) En la sección **Resolver**, elija si desea confirmar o restituir el lote actual de mensajes pulsando **Restaurar mensajes en la cola de transmisión** o **Descartar mensajes**.

- Para hacer ping a un canal de gestor de colas:

a) Localice el canal en la lista.

b) Seleccione **Ping** en el menú .

- Para ver y editar los registros de autorizaciones para un canal de gestor de colas:

a) Localice el canal en la lista.

b) Seleccione **Ver configuración** en el menú .

c) Pulse la pestaña **Seguridad**.

d) Trabaje con los registros de autorizaciones tal como se describe para los registros de autorizaciones del gestor de colas, consulte "IBM MQ Console: Trabajar con registros de autorización de gestor de colas" en la página 104.

- Para suprimir un canal de gestor de colas:

a) Localice el canal en la lista.

b) Seleccione **Configurar** en el menú .

c) Pulse **Suprimir canal**.

V 9.4.0 *IBM MQ Console: Trabajar con canales de aplicación*

Puede utilizar IBM MQ Console para trabajar con canales de aplicación: puede añadir y suprimir canales, iniciar y detener canales, restablecer y recibir canales y hacer ping a canales. También puede ver y establecer las propiedades de un canal de aplicación y gestionar los registros de autorizaciones del canal.

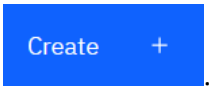
Acerca de esta tarea

Un canal de aplicación es un enlace de comunicación lógica, que utilizan las aplicaciones para conectarse a un gestor de colas a través de una red. La vista del canal de aplicación incluye un panel que muestra una vista rápida de cuántos canales se están ejecutando, cuándo lo están volviendo a intentar y cuántos se han detenido.

z/OS No puede ver ni editar registros de autorizaciones de un canal en z/OS.

Procedimiento

- Para añadir un canal de aplicación:

a) En la vista del gestor de colas, abra la pestaña **Aplicaciones** y pulse **Canales de aplicaciones** y pulse el botón Crear 

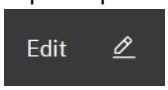
.

b) Pulse el botón siguiente 

.

c) Proporcione la información necesaria para el canal que va a crear. De forma predeterminada, se muestran las propiedades mínimas recomendadas para que pueda proporcionar valores. Puede ver todas las propiedades disponibles seleccionando **Crear personalizado**.

d) Pulse **Crear**. Se creará el nuevo canal con el estado **inactivo**.

- Para iniciar un canal de aplicación:
  a) Localice el canal que desea iniciar en la lista.
  b) Seleccione **Iniciar** en el menú ⋮.
- Para detener un canal de aplicación:
  a) Localice el canal que desea detener en la lista.
  b) Seleccione **Detener** en el menú ⋮.
- Para ver las propiedades de un canal de aplicación:
  a) Localice el canal en la lista.
  b) Seleccione **Ver configuración** en el menú ⋮.
  c) Asegúrese de que la pestaña **Propiedades** está seleccionada. Para editar las propiedades, pulse el botón Editar 

  d) Edite las propiedades que sean necesarias. Si el recuadro de texto de propiedad está inhabilitado, la propiedad es de sólo lectura o solo se puede editar desde la línea de mandatos. Para obtener información sobre las propiedades, consulte Propiedades de canal en la documentación de MQ Explorer.
  e) Pulse **Guardar** para guardar los cambios.
- Para restablecer un canal de aplicación:
  a) Localice el canal en la lista.
  b) Seleccione **Avanzada** en el menú ⋮.
  c) En la sección **Restablecer**, especifique un número de secuencia de mensaje.
  Debe restablecer un canal si no se inicia porque los dos extremos discrepan sobre el número de secuencia del siguiente mensaje a enviar. El número de secuencia de mensaje especifica dicho número.
  d) Pulse **Restablecer canal**.
- Para resolver un canal emisor o servidor:
  a) Localice el canal en la lista.
  b) Seleccione **Avanzada** en el menú ⋮.
  c) En la sección **Resolver**, elija si desea confirmar o restituir el lote actual de mensajes pulsando **Restaurar mensajes en la cola de transmisión** o **Descartar mensajes**.
- Para hacer ping de un canal:

a) Localice el canal en la lista.

b) Seleccione **Ping** en el menú .

- Para ver y editar los registros de autorizaciones para un canal de aplicación:
  a) Localice el canal en la lista.

  b) Seleccione **Configurar** en el menú .

  c) Pulse la pestaña **Seguridad**.

  d) Trabaje con los registros de autorizaciones tal como se describe para los registros de autorizaciones del gestor de colas, consulte "IBM MQ Console: Trabajar con registros de autorización de gestor de colas" en la página 104.
- Para suprimir un canal de aplicación:
  a) Localice el canal en la lista.

  b) Seleccione **Configurar** en el menú .

  c) Pulse **Suprimir canal**.

V 9.4.0 *IBM MQ Console: Trabajar con aplicaciones*

Puede utilizar IBM MQ Console para ver información sobre las aplicaciones conectadas a un gestor de colas.

Acerca de esta tarea

Una aplicación se conecta a un gestor de colas a través de una red utilizando un canal de conexión de servidor. La vista de aplicaciones incluye un panel que muestra una vista rápida de cuántas aplicaciones están conectadas a un gestor de colas.

Procedimiento

- Para ver la información de la aplicación:
  a) En la vista del gestor de colas, abra la pestaña **Aplicaciones** .

  b) Pulse **Aplicaciones conectadas** para abrir la vista de aplicaciones.

  c) Si hay varias instancias de una aplicación, pulse la flecha abajo para ver los detalles de cada instancia.

  d) Pulse los objetos de la vista para obtener más detalles.

V 9.4.0 z/OS *IBM MQ Console: Working with storage classes*

You can use the IBM MQ Console to add, view, delete and update storage classes on z/OS queue managers.

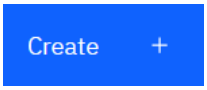
About this task

The storage classes view lists the storage classes that exist for a specific queue manager. You access **Storage classes** from the sidebar on the queue manager **Queues** tab.

See Storage classes for IBM MQ for z/OS and DEFINE STGLASS for more information about storage classes.

Procedure

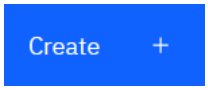
- To add a storage class:
  a) From the queue manager view, open the **Queues** tab, and click **Storage classes**.

b) On the **Storage classes** screen, click the **Create** 

button.

c) Provide the required information for the storage class you are creating.

By default, the minimum recommended properties you need to provide values for are displayed. You can view all of the available properties by selecting **Custom create**.

d) Click the **Create** 

button.

The new storage class is created.

- To delete a storage class:

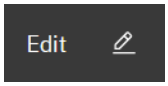
a) Click the spanner button  next to the storage class that you want to delete.

b) In the Edit storage class view, click **Delete storage class**.

c) Confirm that you want to delete the queue by clicking **Delete**. The storage class is deleted.

- To view and edit the properties of a storage class:

a) Click the spanner button  next to the storage class that you want to edit.

b) Click the Edit button 

c) Edit the properties as required. If the property text box is disabled, the property is read-only, or can be set only at the time of creation.

d) Click **Save** to save your changes.

V 9.4.0 z/OS V 9.4.0 *IBM MQ Console: Working with page sets and buffer pools*

You can use the IBM MQ Console to view page sets and buffer pools on z/OS queue managers.

About this task

The page sets and buffer pools views list the page sets and buffer pools that exist for a specific queue manager. You access the **Page sets** and **Buffer pools** views from the sidebar of the queue manager **Queues** tab

See Page sets for IBM MQ for z/OS for more information about page sets, and Buffers and buffer pools for IBM MQ for z/OS for more information about buffer pools.

Procedure

- To view the properties of a page set

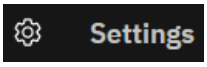
Click the spanner button  next to the page set that you want to view.

- To view the properties of a buffer pool

Click the spanner button  next to the buffer pool that you want to view.

IBM MQ Console valores

Puede especificar algunos valores generales para IBM MQ Console.

Pulse el icono de valores 

para conmutar a la vista de valores de IBM MQ Console .

Utilice los valores para controlar las características siguientes:

- Renovación automática de los gestores de colas cada diez segundos. Esta característica se puede activar o desactivar.
- Si se muestran o no los objetos del sistema. Puede especificar esto para todos los tipos de objeto, o bien seleccionar tipos de objeto individuales.
- Indica si se recopila o no información de rastreo.

Windows Linux Administración utilizando IBM MQ Explorer

IBM MQ Explorer le permite realizar la administración local o remota de la red desde un sistema que ejecute Windows o Linux (solo x86-64).

IBM MQ for Windows e IBM MQ para Linux x86-64 proporcionan una interfaz de administración llamada IBM MQ Explorer para realizar tareas de administración, como alternativa al uso de los mandatos de control o MQSC. Comparación de juegos de mandatos muestra lo que puede hacer utilizando IBM MQ Explorer.

IBM MQ Explorer le permite realizar la administración local o remota de la red desde un sistema que ejecute Windows, o Linux x86-64, apuntando IBM MQ Explorer a los gestores de colas y los clústeres en los que esté interesado. Se puede conectar de forma remota a los gestores de colas que se ejecutan en cualquier plataforma soportada incluyendo z/OS, lo que permite ver, explorar y modificar toda la red troncal de mensajería desde la consola.

Para configurar gestores de colas remotas de IBM MQ para que IBM MQ Explorer los pueda administrar, consulte “Software de requisito previo y definiciones para IBM MQ Explorer” en la página 124.

Le permite realizar tareas, normalmente asociadas a la configuración y el ajuste preciso del entorno de trabajo para IBM MQ, ya sea de forma local o remota dentro de un dominio de sistema Windows o Linux x86-64.

En Linux, puede que IBM MQ Explorer no se inicie correctamente si dispone de una instalación Eclipse. Si esto sucediera, inicie IBM MQ Explorer utilizando un ID de usuario diferente del que utiliza para la otra instalación de Eclipse.

En Linux, para iniciar correctamente IBM MQ Explorer, debe poder grabar un archivo en el directorio de inicio y debe existir un directorio de inicio.

IBM MQ Explorer se puede instalar desde la descarga de IBM MQ Explorer autónoma disponible en Fix Central. Para obtener más información, consulte Instalación y desinstalación de IBM MQ Explorer como una aplicación autónoma en Linux y Windows.

Windows Linux Lo que se puede hacer con IBM MQ Explorer

Puede utilizar IBM MQ Explorer para realizar tareas de administración utilizando una serie de vistas de contenido y diálogos de propiedades. También puede ampliar IBM MQ Explorer escribiendo uno o varios plugins de Eclipse.

Tareas IBM MQ Explorer

Con IBM MQ Explorer, puede realizar las tareas siguientes:

- Crear y suprimir un gestor de colas (solo en la máquina local).
- Iniciar y detener un gestor de colas (solo en la máquina local).
- Definir, visualizar y modificar las definiciones de objetos de IBM MQ, como colas, temas y canales.
- Examinar los mensajes de una cola.
- Iniciar y detener un canal.
- Ver información del estado de un canal, una escucha, una cola u objetos servicio.
- Ver los gestores de colas de un clúster.
- Comprobar qué aplicaciones, usuarios o canales tienen una determinada cola abierta.
- Crear un nuevo clúster de gestor de colas utilizando el asistente para Crear un nuevo clúster.
- Añadir un gestor de colas a un clúster utilizando el asistente para Añadir un gestor de colas a un clúster.
- Gestionar el objeto de información de autenticación, que se utiliza con la seguridad de canal de TLS (seguridad de la capa de transporte).
- Crear y suprimir iniciadores de canal, supervisores desencadenantes y escuchas.
- Iniciar o detener los servidores de mandatos, los iniciadores de canal, los supervisores de activación y las escuchas.
- Configurar servicios específicos para que se inicien automáticamente cuando se inicie un gestor de colas..
- Modificar las propiedades de los gestores de colas.
- Cambiar el gestor de colas local predeterminado.
- Crear objetos de JMS a partir de objetos de IBM MQ y objetos de IBM MQ a partir de objetos de JMS.
- Crear una Fábrica de conexiones JMS para cualquiera de los tipos soportados actualmente.
- Modificar los parámetros de cualquier servicio, como el número de puerto TCP de un escucha o el nombre de una cola de iniciador de canal.
- Iniciar o detener el rastreo de servicio.

Vistas de contenido y diálogos de propiedad

Realice tareas de administración utilizando una serie de Vistas de contenido y Diálogos de propiedades.

Vista de Contenido
Una vista de Contenido es un panel que puede mostrar lo siguiente:

- Atributos y opciones administrativas relativos al propio producto IBM MQ.
- Atributos y opciones administrativas relativos a uno o más objetos relacionados.
- Atributos y opciones administrativas de un clúster.

Diálogos de propiedades
Un diálogo de propiedades es un panel que muestra atributos relativos a un objeto en una serie de campos, algunos de los cuales se pueden editar.

Navegue por IBM MQ Explorer utilizando la vista de navegador. El navegador le permite seleccionar la vista de Contenido que necesite.

Ampliación de IBM MQ Explorer

IBM MQ Explorer presenta la información en un estilo coherente con el de la infraestructura de Eclipse y las aplicaciones de plug-in que soporta Eclipse.

Mediante la ampliación de IBM MQ Explorer, los administradores del sistema tienen la posibilidad de personalizar IBM MQ Explorer para mejorar su forma de administrar IBM MQ.

Para obtener más información, consulte Ampliación de MQ Explorer.

Windows Linux Decidir si utilizar IBM MQ Explorer

A la hora de decidir si desea utilizar IBM MQ Explorer en su instalación, tenga en cuenta la información que figura en este tema.

Debe tener en cuenta los puntos siguientes:

Nombres de objetos
Si utiliza nombres en minúsculas para gestores de colas y otros objetos con IBM MQ Explorer, cuando trabaje con los objetos utilizando mandatos MQSC, debe encerrar los nombres de objetos entre comillas simples o IBM MQ no los reconoce.

Gestores de colas grandes
IBM MQ Explorer funciona mejor con gestores de colas pequeños. Si tiene un gran número de objetos en un solo gestor de colas, puede sufrir retrasos mientras IBM MQ Explorer extrae la información necesaria para presentarla en una vista.

Clústeres
Los clústeres de IBM MQ pueden potencialmente contener cientos o miles de gestores de colas. IBM MQ Explorer presenta los gestores de colas de un clúster mediante una estructura de árbol. El tamaño físico de un clúster no afecta drásticamente la velocidad de IBM MQ Explorer, ya que IBM MQ Explorer no se conecta a los gestores de colas del clúster hasta que el usuario los selecciona.

Configuración de IBM MQ Explorer

En esta sección se describen los pasos que debe realizar para configurar IBM MQ Explorer.

- “Software de requisito previo y definiciones para IBM MQ Explorer” en la página 124
- “Seguridad para IBM MQ Explorer” en la página 125
- “Mostrar y ocultar gestores de colas y clústeres en IBM MQ Explorer” en la página 128
- “Pertenencia a un clúster y IBM MQ Explorer” en la página 129
- “Conversión de datos para IBM MQ Explorer” en la página 130

Software de requisito previo y definiciones para IBM MQ Explorer

Asegúrese de cumplir los siguientes requisitos antes de intentar utilizar el IBM MQ Explorer.

El IBM MQ Explorer sólo puede conectarse con gestores de colas remotos utilizando el protocolo de comunicaciones TCP/IP.

Compruebe que:

1. Se está ejecutando un servidor de mandatos en cada gestor de colas administrado de forma remota.
2. Se está ejecutando un objeto de escucha TCP/IP adecuado en cada gestor de colas remoto. Este objeto puede ser el escucha de IBM MQ o, en sistemas AIX and Linux, el daemon inetd.
3. Existe un canal de conexión de servidor, cuyo nombre predeterminado es SYSTEM.ADMIN.SVRCONN, en todos los gestores de colas remotos.

   Puede crear el canal utilizando el siguiente mandato MQSC:

   ```
   DEFINE CHANNEL(SYSTEM.ADMIN.SVRCONN) CHLTYPE(SVRCONN)
   ```

   Este mandato crea una definición de canal básica. Si desea una definición más compleja (por ejemplo, para configurar la seguridad), necesita parámetros adicionales. Para obtener más información, consulte DEFINE CHANNEL.
4. Debe existir la cola del sistema, SYSTEM.MQEXPLORER.REPLY.MODEL.

Seguridad para IBM MQ Explorer

Si está utilizando IBM MQ en un entorno en el que es importante para usted controlar el acceso de usuarios a determinados objetos, tal vez necesite considerar los aspectos de seguridad que implica la utilización de IBM MQ Explorer.

Autorización para utilizar IBM MQ Explorer

Cualquier usuario puede utilizar IBM MQ Explorer, pero se requieren ciertas autorizaciones para conectarse y acceder a los gestores de colas así como para gestionarlos.

Para realizar tareas tarea administrativas locales utilizando IBM MQ Explorer, un usuario debe tener la autorización necesaria para realizar las tareas de administración. Si el usuario es miembro del grupo mqm, el usuario tiene autorización para realizar todas las tareas administrativas locales.

Para conectarse a un gestor de colas remoto y realizar tareas administrativas remotas utilizando IBM MQ Explorer, el usuario que ejecuta IBM MQ Explorer debe tener las siguientes autorizaciones:

- Autorización CONNECT en el objeto de gestor de colas de destino
- Autorización INQUIRE en el objeto de gestor de colas de destino
- Autorización DISPLAY para el objeto de gestor de colas de destino
- Autorización INQUIRE para la cola SYSTEM.MQEXPLORER.REPLY.MODEL
- Autorización DISPLAY para la cola SYSTEM.MQEXPLORER.REPLY.MODEL
- Autorización INPUT (get) para la cola SYSTEM.MQEXPLORER.REPLY.MODEL
- Autorización OUTPUT (put) para la cola, SYSTEM.MQEXPLORER.REPLY.MODEL
- Autorización OUTPUT (put) para la cola SYSTEM.ADMIN.COMMAND.QUEUE
- Autorización INQUIRE en la cola SYSTEM.ADMIN.COMMAND.QUEUE
- Autorización para realizar la acción seleccionada

Nota: La autorización INPUT está relacionada con la entrada al usuario desde una cola (una operación de obtener). Una autorización OUTPUT está relacionada con la salida desde el usuario a una cola (una operación de transferir).

Para conectarse a un gestor de colas remoto en IBM MQ for z/OS y realizar tareas administrativas remotas utilizando IBM MQ Explorer, debe proporcionarse lo siguiente:

- Un perfil RACF para la cola del sistema, SYSTEM.MQEXPLORER.REPLY.MODEL
- Un perfil RACF para las colas, AMQ.MQEXPLORER.*

Además, el usuario que ejecute IBM MQ Explorer debe tener las siguientes autorizaciones:

- Autorización RACF UPDATE para la cola del sistema, SYSTEM.MQEXPLORER.REPLY.MODEL
- Autorización RACF UPDATE para las colas, AMQ.MQEXPLORER.*
- Autorización CONNECT en el objeto de gestor de colas de destino
- Autorización para realizar la acción seleccionada
- Autorización READ para todos los perfiles hlq.DISPLAY.object de la clase MQCMDS

Para obtener información sobre cómo otorgar autorización a objetos de IBM MQ, consulte Otorgar acceso a un objeto de IBM MQ en sistemas AIX, Linux, and Windows.

Si un usuario intenta realizar una operación para la que no está autorizado, el gestor de colas de destino invoca procedimientos de error de autorización y la operación no se realiza correctamente.

El filtro predeterminado en IBM MQ Explorer es mostrar todos los objetos de IBM MQ. Si hay algún objeto de IBM MQ para el que un usuario no tiene autorización DISPLAY, se generan errores de autorización. Si se están registrando sucesos de autorización, restrinja el rango de objetos que se muestran a aquellos para los que el usuario tiene autorización DISPLAY.

Seguridad de la conexión a gestores de colas remotos desde IBM MQ Explorer

Debe proteger el canal entre IBM MQ Explorer y cada gestor de colas remoto.

IBM MQ Explorer se conecta a los gestores de colas remotos como una aplicación cliente MQI. Esto significa que cada gestor de colas remoto debe tener una definición de un canal de conexión de servidor y un escucha TCP/IP adecuado. Si no protege el canal de conexión del servidor, es posible que una aplicación maliciosa preparada para ello se conecte al mismo canal de conexión de servidor y obtenga acceso a los objetos de gestor de colas con autorización ilimitada. Para proteger el canal de conexión del servidor, especifique un valor, que no sea dejarlo en blanco, para el atributo MCAUSER del canal, utilice los registros de autenticación de canal o bien utilice una salida de seguridad.

El valor predeterminado del atributo MCAUSER es el ID de usuario local. Si especifica un nombre de usuario que no sea un blanco como el atributo MCAUSER del canal de conexión de servidor, todos los programas que se conecten al gestor de colas utilizando este canal se ejecutan con la identidad del usuario nombrado y tiene el mismo nivel de autorización. Esto no ocurre si utiliza registros de autenticación de canal.

Utilización de una salida de seguridad con IBM MQ Explorer

Puede especificar una salida de seguridad predeterminada y salidas de seguridad específicas del gestor de colas utilizando IBM MQ Explorer.

Puede definir una salida de seguridad predeterminada, que se puede utilizar para todas las conexiones de cliente nuevas desde IBM MQ Explorer. Esta salida predeterminada se puede alterar temporalmente en el momento en que se realiza una conexión. También puede definir una salida de seguridad para un gestor de colas individual o para un conjunto de gestores de colas, lo que se lleva a cabo cuando se realiza la conexión. Las salidas se especifican utilizando IBM MQ Explorer. Para obtener más información, consulte la ayuda de IBM MQ Explorer.

Utilización de IBM MQ Explorer para conectarse a un gestor de colas remoto utilizando canales MQI habilitados para TLS

IBM MQ Explorer se conecta a los gestores de colas remotos utilizando un canal MQI. Si desea proteger el canal MQI con el protocolo de seguridad TLS, debe establecer el canal utilizando una tabla de definiciones de canal de cliente.

Para obtener información sobre cómo establecer un canal MQI utilizando una tabla de definición de canal de cliente, consulte IBM MQ MQI clients.

Cuando haya establecido el canal utilizando una tabla de definiciones de canal de cliente, puede utilizar el IBM MQ Explorer para conectarse a un gestor de colas remoto utilizando canales MQI habilitados para TLS, tal como se describe en "Tareas en el sistema en el que se aloja el gestor de colas remoto" en la página 126 y en "Tareas en el sistema que aloja IBM MQ Explorer" en la página 127.

Tareas en el sistema en el que se aloja el gestor de colas remoto

En el sistema en el que se aloje el gestor de colas remoto, efectúe las tareas siguientes:

1. Defina un par de canales de conexión de servidor y conexión de cliente y especifique el valor adecuado para el atributo *SSLCIPH* en la conexión de servidor en ambos canales. Para obtener más información sobre el atributo *SSLCIPH*, consulte Protección de canales con TLS.
2. Envíe la tabla de definición de canal `AMQCLCHL.TAB`, que se encuentra en el directorio `@ipcc` del gestor de colas, al sistema que aloja IBM MQ Explorer.
3. Inicie un escucha TCP/IP en un puerto designado.
4. Coloque los certificados de CA y TLS personales en el directorio SSL del gestor de colas:
   - Linux AIX `/var/mqm/qmgrs/+QMNAME+/SSL` para sistemas AIX and Linux.
   - Windows `C:\Archivos de programa\IBM\MQ\qmgrs\+QMNAME+\SSL` para sistemas Windows.

     Donde `+QMNAME+` es una señal que representa el nombre del gestor de colas.

5. Cree un archivo de base de datos de claves de tipo CMS denominado `key.kdb`. Oculta la contraseña de la base de datos de claves en un archivo especificando el parámetro `-stash` en el mandato **runmqakm** que se utiliza para crear la base de datos de claves.
6. Añada los certificados CA a la base de datos de claves creada en el paso anterior.
7. Importe el certificado personal del gestor de colas a la base de datos de claves.

Para obtener información más detallada sobre cómo trabajar con TLS en sistemas Windows, consulte Trabajar con TLS en AIX, Linux, and Windows.

Tareas en el sistema que aloja IBM MQ Explorer

En el sistema que aloja IBM MQ Explorer, realice las tareas siguientes:

1. Cree un archivo de base de datos de claves de tipo JKS denominado `key.jks`. Establezca una contraseña para este archivo.

   El almacén de claves que IBM MQ Explorer utiliza para la seguridad TLS debe ser un archivo de almacén de claves Java (JKS).
2. Añada los certificados CA a la base de datos de claves creada en el paso anterior.
3. Importe el certificado personal del gestor de colas a la base de datos de claves.
4. En sistemas Windows y Linux, inicie IBM MQ Explorer utilizando el menú del sistema, el archivo ejecutable `MQExplorer` o el mandato **strmqcfg**.
5. En la barra de herramientas de IBM MQ Explorer, pulse **Ventana -> Preferencias**, luego expanda **IBM MQ Explorer** y pulse **Almacenes de certificados SSL de cliente**. Entre el nombre y la contraseña del archivo JKS creado en el paso 1 de “Tareas en el sistema que aloja IBM MQ Explorer” en la página 127, tanto en el Almacén de certificados fiable como en el Almacén de certificados personal, y pulse **Aceptar**.
6. Cierre la ventana **Preferencias** y pulse el botón derecho del ratón en **Gestores de colas**. Pulse **Mostrar/ocultar gestores de colas** y luego pulse **Añadir** en la pantalla **Mostrar/ocultar gestores de colas**.
7. Escriba el nombre del gestor de colas y seleccione la opción **Conectar directamente**. Pulse Siguiente.
8. Seleccione **Utilizar tabla de definiciones de canal de cliente (CCDT)** y especifique la ubicación del archivo de tabla de canales que ha transferido del gestor de colas remoto en el paso 2 en “Tareas en el sistema en el que se aloja el gestor de colas remoto” en la página 126 en el sistema en el que se aloja el gestor de colas remoto.
9. Pulse **Finalizar**. Ahora puede acceder al gestor de colas remoto desde el IBM MQ Explorer.

Conexión a través de otro gestor de colas con IBM MQ Explorer

IBM MQ Explorer le permite conectarse a un gestor de colas a través de un gestor de colas intermedio, al que IBM MQ Explorer ya está conectado.

En este caso, IBM MQ Explorer transfiere mensajes de mandato PCF al gestor de colas intermedio, especificando lo siguiente:

- El parámetro *NombGstColasObj* del descriptor de objeto (MQOD) como nombre del gestor de colas de destino. Para obtener más información sobre la resolución de nombres de cola, consulte Resolución de nombres.
- El parámetro *IdentificadorUsuario* del descriptor de mensaje (MQMD) como el ID de usuario local.

Si la conexión se utiliza después para conectar con el gestor de colas de destino mediante un gestor de colas intermediario, el ID de usuario se incorpora al parámetro *IdentificadorUsuario* del descriptor de mensaje (MQMD). Para que el escucha MCA del gestor de colas de destino acepte este mensaje, debe establecerse el atributo MCAUSER o el ID de usuario ya debe existir con autorización para transferencias.

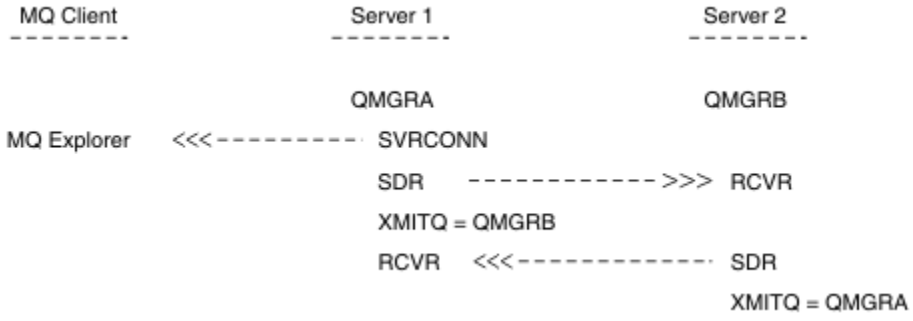
El servidor de mandatos del gestor de colas de destino transfiere mensajes a la cola de transmisión especificando el ID de usuario del parámetro *IdentificadorUsuario* del descriptor de mensaje (MQMD).

Para que esta transferencia se realice correctamente, el ID de usuario ya debe existir en el gestor de colas de destino con autorización para transferencias.

El siguiente ejemplo le muestra cómo conectar un gestor de colas, a través de un gestor de colas intermedio, a IBM MQ Explorer.

Establezca una conexión de administración remota con un gestor de colas. Verifique que:

- El gestor de colas en el servidor está activo y tiene un canal de conexión de servidor (SVRCONN) definido.
- El escucha está activo.
- El servidor de mandatos está activo.
- La cola SYSTEM.MQ EXPLORER.REPLY.MODEL se ha creado y que tiene suficiente autorización.
- Los escuchas, los servidores de mandatos y los canales emisores de los gestores de colas están iniciados.



En este ejemplo:

- IBM MQ Explorer se conecta al gestor de colas QMGRA (que se ejecuta en el Servidor1) utilizando una conexión de cliente.
- El gestor de colas QMGRB en el Servidor2 ahora puede conectarse a IBM MQ Explorer a través de un gestor de colas intermedio (QMGRA)
- Al conectarse a QMGRB con IBM MQ Explorer, seleccione QMGRA como el gestor de colas intermedio

En esta situación, no hay ninguna conexión directa a QMGRB desde IBM MQ Explorer; la conexión a QMGRB se realiza a través de QMGRA.

El gestor de colas QMGRB en el Servidor2 se conecta a QMGRA en el Servidor1 utilizando canales emisor-receptor. El canal entre QMGRA y QMGRB debe configurarse de tal manera que sea posible la administración remota; consulte “Configuración de gestores de colas para la administración remota” en la página 201.

Mostrar y ocultar gestores de colas y clústeres en IBM MQ Explorer

IBM MQ Explorer puede mostrar más de un gestor de colas a la vez. En el panel Mostrar/ocultar gestor de colas (que se puede seleccionar desde el menú del nodo de árbol Gestores de colas), puede decidir si se visualiza información sobre otra máquina (remota). Los gestores de colas locales se detectan automáticamente.

Para mostrar un gestor de colas remoto:

1. Pulse el botón derecho (del ratón) en el nodo de árbol **Gestores de colas** y, a continuación, seleccione **Mostrar/ocultar gestores de colas**.
2. Pulse **Añadir**. Aparece el panel Mostrar/ocultar gestores de colas.
3. Escriba el nombre del gestor de colas remoto y el nombre de host o dirección IP en los campos correspondientes.

El nombre de host o dirección IP se utiliza para establecer una conexión de cliente con el gestor de colas remoto utilizando el canal de conexión de servidor predeterminado, SYSTEM.ADMIN.SVRCONN, o un canal de conexión de servidor definido por el usuario.

4. Pulse **Finalizar**.

El panel Mostrar/ocultar gestores de colas también muestra una lista de todos los gestores de colas visibles. Puede utilizar este panel para ocultar gestores de colas de la vista de navegación.

Si IBM MQ Explorer muestra un gestor de colas que es miembro de un clúster, el clúster se detecta y se muestra automáticamente.

Para exportar la lista de gestores de colas remotos desde este panel:

1. Cierre el panel Mostrar/ocultar gestores de colas.
2. Pulse el botón derecho del ratón en el nodo de árbol de **IBM MQ** superior en el panel Navegación de IBM MQ Explorer, y luego seleccione **Exportar valores de IBM MQ Explorer**
3. Pulse **IBM MQ Explorer > Valores de IBM MQ Explorer**
4. Seleccione **Información de conexión > Gestores de colas remotos**.
5. Seleccione un archivo para almacenar los valores de configuración exportados.
6. Por último, pulse **Finalizar** para exportar la información de conexión de gestor de colas remoto al archivo especificado.

Para importar una lista de gestores de colas remotos:

1. Pulse el botón derecho del ratón en el nodo de árbol de **IBM MQ** superior en el panel Navegación de IBM MQ Explorer y luego seleccione **Importar valores de IBM MQ Explorer**
2. Pulse **IBM MQ Explorer > Valores de IBM MQ Explorer**
3. Pulse **Examinar** y desplácese a la vía de acceso del archivo que contiene la información de conexión de gestor de colas remoto.
4. Pulse **Abrir**. Si el archivo contiene una lista de gestores de colas remotos, se selecciona el recuadro **Información de conexión > Gestores de colas remotos**.
5. Por último, pulse **Finalizar** para importar la información de conexión de gestor de colas remoto a IBM MQ Explorer.

Pertenencia a un clúster y IBM MQ Explorer

IBM MQ Explorer necesita información sobre los gestores de colas que son miembros de un clúster.

Si un gestor de colas es miembro de un clúster, el nodo de árbol del clúster se llenará automáticamente.

Si los gestores de colas se convierten en miembros de clústeres mientras IBM MQ Explorer está ejecutándose, debe mantener IBM MQ Explorer con datos de administración actualizados sobre los clústeres para que éste pueda comunicarse eficazmente con ellos y mostrar información correcta sobre los clústeres cuando se le solicite. Para ello, IBM MQ Explorer necesita la siguiente información:

- El nombre de un gestor de colas de depósito
- El nombre de conexión del gestor de colas de repositorio si éste se encuentra en un gestor de colas remoto.

Con esta información, IBM MQ Explorer puede:

- Utilizar el gestor de colas de repositorio para obtener una lista de los gestores de colas del clúster.
- Administrar los gestores de colas que sean miembros del clúster y estén en plataformas y niveles de mandatos soportados.

La administración no es posible si:

- El depósito elegido deja de estar disponible. IBM MQ Explorer no cambia automáticamente a un repositorio alternativo.
- No se puede contactar con el depósito elegido a través de TCP/IP.

- El repositorio elegido está ejecutándose en un gestor de colas que se está ejecutando en una plataforma y un nivel de mandatos no soportados por IBM MQ Explorer.

Los miembros del clúster que pueden administrarse pueden ser locales o remotos (si es posible comunicar con ellos a través de TCP/IP). IBM MQ Explorer se conecta directamente con los gestores de colas locales que son miembros de un clúster, sin utilizar una conexión de cliente.

Conversión de datos para IBM MQ Explorer

IBM MQ Explorer funciona con el CCSID 1208 (UTF-8). Esto permite a IBM MQ Explorer mostrar los datos de gestores de colas remotos correctamente. Tanto si se conecta a un gestor de colas directamente como si lo hace a través de un gestor de colas intermedio, IBM MQ Explorer requiere que todos los mensajes entrantes se conviertan a CCSID 1208 (UTF-8).

Se emite un mensaje de error si se intenta establecer una conexión entre IBM MQ Explorer y un gestor de colas con un CCSID que IBM MQ Explorer no reconoce.

Las conversaciones soportadas se describen en Conversión de página de códigos.

Windows Utilización de la aplicación Barra de tareas de IBM MQ (sólo Windows)

La aplicación Barra de tareas de IBM MQ muestra un icono en la bandeja del sistema de Windows del servidor. El icono le proporciona el estado actual de IBM MQ y un menú desde el que puede ejecutar algunas acciones simples.

En Windows, el icono de IBM MQ está en la bandeja del sistema del servidor y tiene superpuesto un símbolo de estado diferenciado con colores que puede tener uno de los siguientes significados:

Verde
Funciona correctamente, no hay alertas en este momento

Azul
Indeterminado; IBM MQ se está iniciando o cerrando

Amarillo
Alerta; uno o más servicios están fallando o ya han fallado.

Para visualizar el menú, pulse el botón derecho del ratón en el icono de IBM MQ. Desde el menú puede realizar las acciones siguientes:

- Pulse **Abrir** para abrir el supervisor de alertas de IBM MQ.
- Pulse **Salir** para salir de la aplicación de barra de tareas de IBM MQ.
- Pulse **IBM MQ Explorer** para iniciar IBM MQ Explorer.
- Pulse **Detener IBM MQ** para detener IBM MQ.
- Pulse **Acerca de IBM MQ** para visualizar información sobre el supervisor de alertas de IBM MQ.

Windows La aplicación de supervisor de alertas de IBM MQ (sólo Windows)

El supervisor de alertas de IBM MQ es una herramienta de detección de errores que identifica y registra problemas con IBM MQ en una máquina local.

El supervisor de alertas muestra información sobre el estado actual de la instalación local de un servidor de IBM MQ. También supervisa la Interfaz avanzada de configuración y energía (ACPI) de Windows y asegura que se utilicen los valores de la ACPI.

Desde el supervisor de alertas de IBM MQ, puede:

- Acceder a IBM MQ Explorer directamente
- Ver información relacionada con todas las alertas pendientes
- Concluir el servicio IBM MQ en la máquina local

- Direccionar mensajes de alerta a través de la red a una cuenta de usuario configurable o a una estación de trabajo o un servidor de Windows

Trabajar con objetos de IBM MQ locales

Se pueden administrar objetos locales de IBM MQ para soportar programas de aplicación que usen la interfaz de cola de mensajes (Message Queue Interface, MQI).

Acerca de esta tarea

En este contexto, administración local significa crear, visualizar, cambiar, copiar y suprimir objetos de IBM MQ.

Además de los enfoques descritos en esta sección, puede utilizar la IBM MQ Explorer local para administrar objetos IBM MQ. Para obtener más información, consulte “Administración utilizando IBM MQ Explorer” en la página 122.

Procedimiento

- Utilice la información de los temas siguientes como ayuda en la administración local objetos de IBM MQ.
  - Programas de aplicación que utilizan la MQI
  - “Administración de IBM MQ utilizando mandatos MQSC” en la página 12
  - “Visualización y modificación de atributos del gestor de colas” en la página 139
  - “Trabajar con colas locales” en la página 142
  - “Trabajar con colas alias” en la página 155
  - “Trabajar con colas modelo” en la página 157
  - “Trabajar con servicios” en la página 186
  - “Gestión de objetos para desencadenamiento” en la página 194

Trabajar con gestores de colas

Puede utilizar los mandatos de control para iniciar y detener un gestor de colas. Puede utilizar los mandatos MQSC para visualizar o modificar los atributos del gestor de colas.

Tareas relacionadas

Creación de gestores de colas en Multiplatforms

Multi Inicio de un gestor de colas

Cuando se crea un gestor de colas, debe iniciarse para habilitarlo para que pueda procesar mandatos o llamadas MQI.

Acerca de esta tarea

Puede iniciar un gestor de colas mediante el mandato **strmqm**. Para obtener una descripción del mandato **strmqm** y sus opciones, consulte strmqm.

Windows Linux O bien, en los sistemas Windows y Linux (plataformas x86 y x86-64), puede iniciar un gestor de colas utilizando IBM MQ Explorer.

Windows En Windows puede iniciar un gestor de colas automáticamente cuando se inicia el sistema utilizando IBM MQ Explorer. Para obtener más información, consulte “Administración utilizando IBM MQ Explorer” en la página 122.

Procedimiento

- Para iniciar un gestor de colas mediante el mandato **strmqm**, especifique el mandato seguido del nombre del gestor de colas que desee iniciar.

  Por ejemplo, para iniciar un gestor de colas llamado QMB, especifique el siguiente mandato:

  ```
  strmqm QMB
  ```

  Nota: Debe utilizar el mandato **strmqm** desde la instalación asociada al gestor de colas con el que está trabajando. Puede averiguar con qué instalación está asociado un gestor de colas utilizando el mandato `dspmq -o installation` .

  El mandato `strmqm` no devolverá el control hasta que el gestor de colas que se haya iniciado y esté preparado para aceptar solicitudes de conexión.

- Windows Linux

  Para iniciar un gestor de colas utilizando IBM MQ Explorer, realice los pasos siguientes:

  a) Abra IBM MQ Explorer.

  b) En la vista de Navegador, seleccione el gestor de colas.

  c) Pulse **Iniciar**.

Resultados

El gestor de colas se inicia.

Si el inicio del gestor de colas tarda más de unos segundos, IBM MQ emite mensajes informativos de forma intermitente que detallan el progreso del inicio.

Multi Detención de un gestor de colas

Puede utilizar el mandato **endmqm** para detener un gestor de colas. Este mandato proporciona cuatro formas para detener un gestor de colas: un cierre controlado, o desactivado temporalmente, una conclusión inmediata, una conclusión preferente o una conclusión en espera. O bien, en los sistemas Windows y Linux, se puede parar un gestor de colas utilizando IBM MQ Explorer.

Acerca de esta tarea

Existen cuatro formas para detener un gestor de colas de una sola instancia con el mandato **endmqm**:

Cierre controlado (por desactivación temporal)
De forma predeterminada, el mandato **endmqm** realiza una conclusión por desactivación temporal del gestor de colas especificado. Un cierre por desactivación temporal espera hasta que todas las aplicaciones conectadas se hayan desconectado, por tanto podría tardar un poco en completarse.

Conclusión inmediata
Para una conclusión inmediata, se permite que terminen las llamadas MQI que haya en ese momento, pero las llamadas nuevas fallarán. Este tipo de conclusión no espera a que las aplicaciones se desconecten del gestor de colas.

Conclusión preferente
El gestor de colas se para de forma inmediata. Utilice este tipo de cierre solo en circunstancias excepcionales, por ejemplo, cuando un gestor de colas no se pare como resultado de un mandato **endmqm** normal.

Conclusión de espera
Este tipo de conclusión es parecido a la conclusión controlada excepto en que el usuario sólo recupera el control después de que el gestor de colas se haya detenido.

El mandato **endmqm** detiene todas las instancias de un gestor de colas multiinstancia de la misma manera que detiene un gestor de colas de una sola instancia. Puede emitir el mandato **endmqm** en una instancia

activa o en una de las instancias en espera de un gestor de colas multiinstancia. Sin embargo, debe emitir **endmqm** en la instancia activa para finalizar el gestor de colas.

Tiene la opción de finalizar el gestor de colas dentro de un tiempo de destino de un número de segundos que especifique, con o sin interrumpir las tareas de mantenimiento del gestor de colas no esenciales, consulte “Finalización de un gestor de colas dentro de un tiempo de destino” en la página 135.

Atención:

- Los mensajes persistentes persistirán independientemente del tipo de conclusión utilizado (incluidos los procesos IBM MQ que finalizan manualmente), mientras que no se puede garantizar que los mensajes no persistentes sobrevivan a cualquier tipo de conclusión.

  La especificación de la propiedad de cola NPMCLASS (HIGH) guarda los mensajes no persistentes de la mejor manera posible. El uso de **endmqm -t**, **endmqm -tp**, **endmqm -p**o la finalización manual de procesos IBM MQ reduce las posibilidades de que los mensajes NPMCLASS (HIGH) sobrevivan a un ciclo de conclusión o reinicio de IBM MQ en comparación con **endmqm -w** o **endmqm -i**
- El tiempo combinado para finalizar y reiniciar el gestor de colas puede ser más largo como resultado de utilizar un método de conclusión más abrupto, especialmente cuando se utilizan las opciones **-p** y **-tp** .

  Si el gestor de colas tiene que volver a recurrir a la finalización de procesos IBM MQ para finalizar el gestor de colas, es probable que sea necesario más reconciliación del estado del gestor de colas, cuando se reinicia el gestor de colas.

Para ver una explicación detallada del mandato **endmqm** y sus opciones, consulte el apartado endmqm.

Consejo: A menudo, la causa de los problemas que pueden surgir al cerrar un gestor de colas está en las aplicaciones. Por ejemplo, cuando las aplicaciones:

- No comprueban adecuadamente los códigos de retorno MQI
- No solicitan notificación de una conclusión progresiva
- Terminan sin desconectarse del gestor de colas (emitiendo una llamada MQDISC)

Si se produce un problema al intentar parar el gestor de colas, se puede salir del mandato **endmqm** con Control+C. Después, se puede ejecutar otro mandato **endmqm**, pero esta vez con un distintivo que especifique el tipo de cierre que se desea realizar.

Windows Linux Como alternativa al mandato **endmqm**, en Windows y Linux, se puede parar un gestor de colas utilizando IBM MQ Explorer para llevar a cabo un cierre controlado o inmediato.

Procedimiento

- Para detener el gestor de colas mediante el mandato **endmqm**, especifíquelo seguido del parámetro adecuado, si es necesario, y del nombre del gestor de colas que desee detener.

  Nota: Debe utilizar el mandato **endmqm** desde la instalación asociada al gestor de colas con el que está trabajando. Para averiguar qué instalación está asociada a un gestor de colas, utilice el mandato **dspmq**:

  ```
  dspmq -o installation
  ```

  - Para realizar un cierre controlado (por desactivación temporal), ejecute el mandato **endmqm** tal y como se muestra en el ejemplo siguiente, que para un gestor de colas llamado QMB:

    ```
    endmqm QMB
    ```

    De forma alternativa, la ejecución del mandato **endmqm** con el parámetro **-c**, tal y como se muestra en el ejemplo siguiente, es equivalente al mandato endmqm QMB.

    ```
    endmqm -c QMB
    ```

En ambos casos, el control se devuelve inmediatamente y no se notifica cuándo se para el gestor de colas. Si desea que el mandato espere a que todas las aplicaciones se hayan parado y el gestor de colas haya finalizado antes de devolverle el control, utilice el parámetro **-w** en su lugar, como se muestra en el ejemplo siguiente.

```
endmqm -w QMB
```

- Para llevar a cabo un cierre inmediato, escriba el mandato **endmqm** con el parámetro **-i**, como se muestra en el ejemplo siguiente:

  ```
  endmqm -i QMB
  ```

- Para llevar a cabo un cierre preferente, escriba el mandato **endmqm** con el parámetro **-p**, como se muestra en el ejemplo siguiente:

  ```
  endmqm -p QMB
  ```

  Atención: Un cierre preferente puede tener consecuencias imprevisibles para las aplicaciones conectadas. No utilice esta opción a menos que todos los demás intentos de parar el gestor de colas con un mandato **endmqm** normal hayan fracasado. ALW Si el cierre preferente no funciona, intente "Detención manual de un gestor de colas" en la página 135 en su lugar.

- Para solicitar la reconexión de cliente automática, especifique el mandato **endmqm** con el parámetro `-r`. Este parámetro tiene el efecto de restablecer la conectividad de los clientes con otros gestores de colas en el grupo de gestores de colas.

  Nota: La finalización de un gestor de colas mediante el mandato **endmqm** predeterminado, no desencadena la reconexión automática del cliente.

- Para transferir a una instancia en espera de un gestor de colas multiinstancia después de concluir la instancia activa, especifique el mandato **endmqm** con el parámetro `-s` en la instancia activa del gestor de colas multiinstancia.
- Para finalizar la instancia en espera de un gestor de colas multiinstancia y dejar la instancia activa en ejecución, especifique el mandato **endmqm** con el parámetro `-x` en la instancia en espera del gestor de colas multiinstancia.

- Windows Linux

  En Windows y Linux, para detener el gestor de colas con IBM MQ Explorer, siga los pasos siguientes:

  a) Abra IBM MQ Explorer.

  b) Seleccione el gestor de colas en la vista de navegador.

  c) Pulse **Detener**.

  Aparece el panel **Terminar gestor de colas**.

  d) Seleccione **Controlado** o **Inmediato**.

  e) Pulse **Aceptar**.

  El gestor de colas se detiene.

Tareas relacionadas

Aplicación de actualizaciones de nivel de mantenimiento a gestores de colas multiinstancia en AIX

Aplicación de actualizaciones de nivel de mantenimiento a gestores de colas multiinstancia en Linux

Aplicación de actualizaciones de nivel de mantenimiento a gestores de colas multiinstancia en Windows

Referencia relacionada

endmqm (finalizar gestor de colas)

Finalización de un gestor de colas dentro de un tiempo de destino

Puede finalizar el gestor de colas dentro de un tiempo de destino de un número de segundos que especifique, con o sin interrumpir las tareas de mantenimiento del gestor de colas no esenciales.

Existen dos formas de especificar una hora de destino cuando se utiliza el mandato **endmqm** . La opción **-t** permite que se completen todas las tareas de mantenimiento del gestor de colas, lo que puede prolongar la fase de finalización del gestor de colas. La opción **-tp** interrumpe las tareas de mantenimiento del gestor de colas no esenciales si es necesario para cumplir con el tiempo de destino especificado.

Las tareas de mantenimiento no esenciales incluyen: compactación de archivos en cola y mensajes NPMCLASS (HIGH) persistentes. En el resto de esta página, se utiliza la palabra 'limpieza'.

En función de los patrones de uso de la aplicación, la compactación del archivo de cola puede tardar mucho tiempo, por lo que si el objetivo principal es finalizar el gestor de colas rápidamente, utilice la opción **-tp** .

Cuando se especifica un objetivo de tiempo, el tipo de conclusión de **-w**, **-i** o **-p** indica el tipo de conclusión de inicio.

Nota: Una conclusión `immediate` sigue siendo ordenada, que difiere de una conclusión `controlled` básicamente en la forma en la que se desactivan temporalmente las aplicaciones en ejecución. Una conclusión de `immediate` sigue realizando tareas de mantenimiento. Una conclusión de tiempo limitado abandona estas acciones cuando interfieren con el cumplimiento de la hora de destino.

El gestor de colas escala el tipo de conclusión según sea necesario, en un intento por cumplir el objetivo de tiempo. Por ejemplo:

- Un objetivo de tiempo de 10 segundos **-t** que empieza en **-w** podría estar siete segundos con la inmovilización, dos segundos con la conclusión inmediata del gestor de colas, incluyendo el mantenimiento, y después realizar una conclusión inmediata sin realizar ningún otro mantenimiento:

  ```
  endmqm  -w -t 10 queue_manager
  ```

- Un objetivo de tiempo **-tp** de 10 segundos podría estar siete segundos con la inmovilización, dos segundos con la conclusión inmediata del gestor de colas, incluido el mantenimiento, un segundo con la conclusión inmediata sin realizar ningún otro mantenimiento y, a continuación, empezar a finalizar los procesos de IBM MQ:

  ```
  endmqm -c -tp 10 queue_manager
  ```

- Un objetivo de tiempo de dos segundos **-tp** en **-i** podría estar un segundo con la conclusión inmediata del gestor de colas, incluido el mantenimiento, un segundo con la conclusión inmediata sin realizar ningún otro mantenimiento y, a continuación, empezar a finalizar los procesos de IBM MQ:

  ```
  endmqm -i -tp 2 queue_manager
  ```

- Un objetivo de tiempo de un segundo en **-w** podría estar 0,1 segundos en `wait` (espera), por ejemplo, solo el tiempo suficiente para enviar los códigos de retorno de IBM MQ a las aplicaciones conectadas, 0,9 segundos con la conclusión inmediata del gestor de colas, incluido el mantenimiento, y después con la conclusión inmediata sin realizar ningún otro mantenimiento; a continuación, empezar a finalizar los procesos de IBM MQ:

Referencia relacionada

endmqm (finalizar gestor de colas)

ALW Detención manual de un gestor de colas

Si fallan los métodos empleados habitualmente para detener y eliminar gestores de colas, puede recurrir a los métodos aquí descritos.

Acerca de esta tarea

La forma estándar de detener los gestores de colas es utilizar el mandato **endmqm**, tal como se describe en “Detención de un gestor de colas” en la página 132. Si no puede parar un gestor de colas de la forma habitual, puede intentar pararlo manualmente. El modo en que lo haga dependerá de la plataforma que esté utilizando.

Procedimiento

- Windows
  Para detener un gestor de colas en Windows, consulte “Parada manual de un gestor de colas en Windows” en la página 136.
- Linux AIX
  Para detener un gestor de colas en AIX o Linux, consulte “Parada manual de un gestor de colas en AIX and Linux” en la página 137.

Tareas relacionadas

Creación y gestión de gestores de colas en Multiplatforms

Referencia relacionada

endmqm

Windows *Parada manual de un gestor de colas en Windows*

Si no se puede parar un gestor de colas en Windows con el mandato **endmqm**, se puede intentar parar el gestor de colas manualmente terminando los procesos que están ejecutando y parando el servicio de IBM MQ.

Acerca de esta tarea

Consejo: El Administrador de tareas de Windows y el mandato **tasklist** proporcionan información limitada sobre las tareas. Para obtener más información para determinar qué procesos se relacionan con un gestor de colas determinado, considere la posibilidad de utilizar una herramienta como *Explorador de procesos* (`procexp.exe`), que está disponible para su descarga desde el sitio web de Microsoft en http://www.microsoft.com.

Para parar un gestor de colas en Windows, siga estos pasos:

Procedimiento

1. Enumere los nombres (ID) de los procesos que están ejecutándose actualmente utilizando el Administrador de tareas de Windows.
2. Finalice los procesos utilizando el Administrador de tareas de Windows o el mandato **taskkill**, en el orden siguiente (si están en ejecución):

Tabla 7. Procesos Windows que hay que parar si están ejecutando

Nombre de proceso	Descripción
AMQZMUC0	Gestor de procesos críticos
AMQZXMA0	Controlador de ejecución
AMQZFUMA	Proceso del OAM
AMQZLAA0	Agentes LQM
AMQZLSA0	Agentes LQM
AMQZMUF0	Gestor de programas de utilidad
AMQZMGR0	Controlador de procesos

Tabla 7. Procesos Windows que hay que parar si están ejecutando (continuación)	
Nombre de proceso	**Descripción**
AMQZMUR0	Gestor de procesos reiniciable
AMQFQPUB	Proceso de Publicación/suscripción
AMQFCXBA	Proceso de operador de intermediario
AMQRMPPA	Proceso de agrupación de procesos
AMQCRSTA	Proceso de trabajo de respuesta sin hebra
AMQCRS6B	Conexión de cliente y canal receptor LU62
AMQRRMFA	Proceso de depósito (para clústeres)
AMQPCSEA	El servidor de mandatos
RUNMQTRM	Invocar un supervisor de activación para un servidor
RUNMQDLQ	Invocar manejador de la cola de mensajes no entregados
RUNMQCHI	El proceso de iniciado de canal
RUNMQLSR	El proceso de escucha de canal
AMQXSSVN	Servidores de memoria compartida

3. Detenga el servicio IBM MQ desde **Herramientas de administración** > **Servicios** en el Panel de control de Windows.
4. Si ha probado todos los métodos y el gestor de colas no se ha detenido, reinicie el sistema.

Linux AIX *Parada manual de un gestor de colas en AIX and Linux*

Si no puede detener un gestor de colas en AIX o Linux utilizando el mandato **endmqm**, puede intentar detener el gestor de colas manualmente finalizando los procesos que están ejecutando y deteniendo el servicio de IBM MQ.

Acerca de esta tarea

Para detener un gestor de colas en AIX o Linux, realice los pasos siguientes.

Si detiene el gestor de colas manualmente, es posible que se tome FFST y que los archivos FDC se coloquen en `/var/mqm/errors`. Esto no debe considerarse un defecto del gestor de colas.

El gestor de colas se reiniciará normalmente, incluso después de haberlo parado con el método manual.

Procedimiento

1. Busque los identificadores de proceso (PID) de los programas del gestor de colas que todavía se están ejecutando utilizando el mandato **ps** .

   Por ejemplo, si el nombre del gestor de colas es QMNAME, puede utilizar el siguiente mandato:

   ```
   ps -ef | grep QMNAME
   ```

2. Finalice los procesos de gestor de colas que todavía estén en ejecución utilizando el mandato **kill** , especificando los PID descubiertos utilizando el mandato **ps** .

   Para finalizar un proceso, utilice **kill -KILL <pid>** o el mandato **kill -9 <pid>** equivalente.

   Tiene que trabajar a través de los PID que desea matar, uno por uno, emitiendo ese mandato cada vez.

Importante: Si utiliza cualquier señal que no sea **9(SIGKILL)** , el proceso probablemente no se detendrá y obtendrá resultados imprevisibles.

Finalice los procesos en el orden siguiente:

Tabla 8. Procesos AIX and Linux que hay que parar si están ejecutando

Nombre de proceso	**Descripción**
amqzmuc0	Gestor de procesos críticos
amqzxma0	Controlador de ejecución
amqzfuma	Proceso del OAM
amqzlaa0	Agentes LQM
amqzlsa0	Agentes LQM
amqzmuf0	Gestor de programas de utilidad
amqzmur0	Gestor de procesos reiniciable
amqzmgr0	Controlador de procesos
amqfqpub	Proceso de Publicación/suscripción
amqfcxba	Proceso de operador de intermediario
amqrmppa	Proceso de agrupación de procesos
amqcrsta	Proceso de trabajo de respuesta sin hebra
amqcrs6b	Conexión de cliente y canal receptor LU62
amqrrmfa	Proceso de depósito (para clústeres)
amqpcsea	El servidor de mandatos
runmqtrm	Invocar un supervisor de activación para un servidor
runmqdlq	Invocar manejador de la cola de mensajes no entregados
runmqchi	El proceso de iniciado de canal
runmqlsr	El proceso de escucha de canal

Tareas relacionadas

"Detención de un gestor de colas" en la página 132
Puede utilizar el mandato **endmqm** para detener un gestor de colas. Este mandato proporciona cuatro formas para detener un gestor de colas: un cierre controlado, o desactivado temporalmente, una conclusión inmediata, una conclusión preferente o una conclusión en espera. O bien, en los sistemas Windows y Linux, se puede parar un gestor de colas utilizando IBM MQ Explorer.

Multi Reinicio de un gestor de colas

Puede utilizar el mandato **strmqm** para reiniciar un gestor de colas o, en los sistemas Windows y Linux x86-64, puede reiniciar un gestor de colas desde IBM MQ Explorer.

Acerca de esta tarea

Puede reiniciar un gestor de colas mediante el mandato **strmqm**. Para obtener una descripción del mandato **strmqm** y sus opciones, consulte strmqm.

Windows Linux En los sistemas Windows y Linux x86-64, puede reiniciar un gestor de colas utilizando IBM MQ Explorer del mismo modo que para iniciar un gestor de colas.

Procedimiento

- Para reiniciar un gestor de colas mediante el mandato **strmqm**, especifique el mandato seguido del nombre del gestor de colas que desee reiniciar.

  Por ejemplo, para iniciar un gestor de colas denominado `strmqm saturn.queue.manager`, especifique el siguiente mandato:

  ```
  strmqm saturn.queue.manager
  ```

- Windows Linux

  Para iniciar un gestor de colas utilizando IBM MQ Explorer, realice los pasos siguientes:

  a) Abra IBM MQ Explorer.

  b) En la vista de Navegador, seleccione el gestor de colas.

  c) Pulse **Iniciar**.

Resultados

El gestor de colas se reinicia.

Si el reinicio del gestor de colas tarda más de unos pocos segundos, IBM MQ emite mensajes de información de forma intermitente que detallan el progreso del inicio.

Visualización y modificación de atributos del gestor de colas

Utilice el mandato MQSC de **DISPLAY QMGR** para visualizar los parámetros del gestor de colas para un gestor de colas. Utilice el mandato MQSC de **ALTER QMGR** para modificar los parámetros del gestor de colas para un gestor de colas local.

Antes de empezar

Nota: Los pasos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Procedimiento

- Para visualizar los atributos del gestor de colas especificado en el mandato **runmqsc**, utilice el mandato MQSC **DISPLAY QMGR**:

  ```
  DISPLAY QMGR
  ```

  En el ejemplo siguiente se muestra la salida típica de este mandato:

  ```
  DISPLAY QMGR
       1 : DISPLAY QMGR
  AMQ8408I: Display Queue Manager details.
     QMNAME(QM1)                             ACCTCONO(DISABLED)
     ACCTINT(1800)                           ACCTMQI(OFF)
     ACCTQ(OFF)                              ACTIVREC(MSG)
     ACTVCONO(DISABLED)                      ACTVTRC(OFF)
     ADVCAP(DISABLED)                        ALTDATE(2022-05-05)
     ALTTIME(14.24.34)                       AMQPCAP(NO)
     AUTHOREV(DISABLED)                      CCSID(437)
     CERTLABL(ibmwebspheremqqm1)             CERTVPOL(ANY)
     CHAD(DISABLED)                          CHADEV(DISABLED)
     CHADEXIT( )                             CHLEV(DISABLED)
     CHLAUTH(ENABLED)                        CLWLDATA( )
     CLWLEXIT( )                             CLWLLEN(100)
     CLWLMRUC(999999999)                     CLWLUSEQ(LOCAL)
  ```

```
   CMDEV(DISABLED)                         CMDLEVEL(930)
   COMMANDQ(SYSTEM.ADMIN.COMMAND.QUEUE)    CONFIGEV(DISABLED)
   CONNAUTH(SYSTEM.DEFAULT.AUTHINFO.IDPWOS)
   CRDATE(2020-12-22)                      CRTIME(15.42.49)
   CUSTOM( )                               DEADQ( )
   DEFCLXQ(SCTQ)                           DEFXMITQ( )
   DESCR( )                                DISTL(YES)
   IMGINTVL(60)                            IMGLOGLN(OFF)
   IMGRCOVO(YES)                           IMGRCOVQ(YES)
   IMGSCHED(MANUAL)                        INHIBTEV(DISABLED)
   INITKEY( )                              IPADDRV(IPV4)
   LOCALEV(DISABLED)                       LOGGEREV(DISABLED)
   MARKINT(5000)                           MAXHANDS(256)
   MAXMSGL(4194304)                        MAXPROPL(NOLIMIT)
   MAXPRTY(9)                              MAXUMSGS(10000)
   MONACLS(QMGR)                           MONCHL(OFF)
   MONQ(OFF)                               PARENT( )
   PERFMEV(DISABLED)                       PLATFORM(WINDOWS10)
   PSMODE(ENABLED)                         PSCLUS(ENABLED)
   PSNPMSG(DISCARD)                        PSNPRES(NORMAL)
   PSRTYCNT(5)                             PSSYNCPT(IFPER)
   QMID(QM1_2020-12-22_15.42.49)           REMOTEEV(DISABLED)
   REPOS( )                                REPOSNL( )
   REVDNS(ENABLED)                         ROUTEREC(MSG)
   SCHINIT(QMGR)                           SCMDSERV(QMGR)
   SPLCAP(DISABLED)                        SSLCRLNL( )
   SSLCRYP( )                              SSLEV(DISABLED)
   SSLFIPS(NO)                             KEYRPWD( )
   SSLKEYR(C:\ProgramData\IBM\MQ\qmgrs\QM1\ssl\key)
   SSLRKEYC(32767)                         STATACLS(QMGR)
   STATCHL(OFF)                            STATINT(1800)
   STATMQI(OFF)                            STATQ(OFF)
   STRSTPEV(ENABLED)                       SUITEB(NONE)
   SYNCPT                                  TREELIFE(1800)
   TRIGINT(999999999)                      VERSION(09030000)
   XRCAP(NO)
```

Nota: SYNCPT es un atributo de gestor de colas de sólo lectura.

El parámetro **ALL** es el valor predeterminado en el mandato **DISPLAY QMGR**. Muestra todos los atributos del gestor de colas. En particular, la salida indica el nombre del gestor de colas predeterminado, el nombre de la cola de mensajes no entregados y el nombre de la cola de mandatos.

La existencia de estas colas puede confirmarse entrando el mandato:

```
DISPLAY QUEUE (SYSTEM.*)
```

Esto hace que se visualice una lista de colas que coinciden con la raíz SYSTEM.*. Los paréntesis son obligatorios.

- Para modificar los atributos del gestor de colas especificado en el mandato **runmqsc**, utilice el mandato MQSC **ALTER QMGR**, especificando los atributos y valores que desee cambiar.

  Por ejemplo, utilice los siguientes mandatos para modificar los atributos de `jupiter.queue.manager`:

```
runmqsc jupiter.queue.manager
ALTER QMGR DEADQ (ANOTHERDLQ) INHIBTEV (ENABLED)
```

  El mandato **ALTER QMGR** cambia la cola de mensajes no entregados utilizada y habilita los sucesos de inhibición.

  Los parámetros no especificados en el mandato **ALTER QMGR** provocan que los valores existentes para esos parámetros se queden sin modificar.

Tareas relacionadas

Creación de gestores de colas en Multiplatforms

Referencia relacionada

Atributos para el gestor de colas

runmqsc (ejecutar mandatos MQSC)

DISPLAY QMGR
ALTER QMGR

Multi Supresión de un gestor de colas

Puede suprimir un gestor de colas utilizando el mandato de control **dltmqm** . O bien, en los sistemas Windows y Linux, puede utilizar IBM MQ Explorer para suprimir un gestor de colas.

Antes de empezar

Atención:

- Suprimir un gestor de colas es una medida muy drástica, ya que también se suprimen todos los recursos asociados al mismo, incluidas todas las colas, sus mensajes y todas las definiciones de objetos. Si utiliza el mandato de control **dltmqm** , no aparece ninguna solicitud que le permita cambiar de opinión; al pulsar la tecla Intro, se perderán todos los recursos asociados.
- Windows En Windows, al suprimir un gestor de colas también se elimina el gestor de colas de la lista de inicio automático (que se describe en “Inicio de un gestor de colas” en la página 131). Cuando se haya completado el mandato, se mostrará un mensaje `IBM MQ queue manager ending` ; no se le indicará que el gestor de colas se ha suprimido.
- Al suprimir un gestor de colas de clúster, no se suprime del clúster. Para obtener más información, consulte las notas de uso en dltmqm.

Acerca de esta tarea

Puede suprimir un gestor de colas utilizando el mandato de control **dltmqm** . Para obtener una descripción del mandato **dltmqm** y sus opciones, consulte dltmqm. Debería asegurarse de que sólo los administradores de confianza tengan autorización para utilizar este mandato. (Para obtener información sobre la seguridad, consulte Configuración de la seguridad en sistemas AIX, Linux, and Windows.)

Windows Linux O bien, en los sistemas Windows y Linux (plataformas x86 y x86-64), puede suprimir un gestor de colas utilizando IBM MQ Explorer.

Procedimiento

- Para suprimir un gestor de colas mediante el mandato **dltmqm**, realice los pasos siguientes:
  a) Detenga el gestor de colas.
  b) Emita el mandato siguiente:

  ```
  dltmqm QMB
  ```

  Nota: Debe utilizar el mandato **dltmqm** desde la instalación asociada al gestor de colas con el que está trabajando. Puede averiguar con qué instalación está asociado un gestor de colas mediante el mandato `dspmq -o installation`.
- Windows Linux
  Para suprimir un gestor de colas utilizando IBM MQ Explorer, realice los pasos siguientes:
  a) Abra IBM MQ Explorer.
  b) En la vista de Navegador, seleccione el gestor de colas.
  c) Si el gestor de colas no se ha detenido, deténgalo.
  Para detener el gestor de colas, pulse con el botón derecho y seleccione **Detener**.
  d) Suprima el gestor de colas.
  Para suprimir el gestor de colas, pulse con el botón derecho y seleccione **Suprimir**.

Resultados

Se suprime el gestor de colas.

Detención de canales MQI

Cuando se emite un mandato STOP CHANNEL contra un canal de conexión del servidor, puede elegir qué método que se debe utilizar para detener el canal de conexión de cliente. Esto significa que un canal de cliente que emite una llamada de espera MQGET se puede controlar y puede decidir cómo y cuándo se detendrá el canal.

El mandato STOP CHANNEL se puede emitir con tres modalidades, que indican cómo debe detenerse el canal:

Inmovilizar
Detiene el canal después de que se hayan procesado los mensajes actuales.

Si la compartición de conversaciones está habilitada, el IBM MQ MQI client se percata de la solicitud de detención puntualmente; este tiempo depende de la velocidad de la red. La aplicación cliente se percata de la solicitud de detención como resultado de emitir una llamada subsiguiente a IBM MQ.

Forzar
Detiene el canal de forma inmediata.

Terminar
Detiene el canal de forma inmediata. Si el canal se está ejecutando como un proceso, se puede finalizar el proceso de canal o si el canal se está ejecutando como una hebra, finalizará su hebra.

Se trata de un proceso de varias fases. Si se emplea la modalidad de terminación, se intenta detener el canal de conexión con el servidor, primero con la modalidad de desactivación temporal, después con la modalidad de forzar y, si es necesario, con la modalidad de terminación. El cliente puede recibir distintos códigos de retorno durante las distintas fases de terminación. Si se termina el proceso o la hebra, el cliente recibe un error de comunicación.

Los códigos de retorno que se devuelven a la aplicación varían según la llamada MQI emitida y el canal STOP CHANNEL emitido. El cliente recibirá un código de retorno MQRC_CONNECTION_QUIESCING o MQRC_CONNECTION_BROKEN. Si un cliente detecta MQRC_CONNECTION_QUIESCING, debe intentar completar la transacción actual y terminar. Esto no es posible con MQRC_CONNECTION_BROKEN. Si el cliente no completa la transacción y termina con suficiente rapidez, obtendrá CONNECTION_BROKEN al cabo de unos segundos. Es más probable que un mandato STOP CHANNEL con MODE(FORCE) o MODE(TERMINATE) dé como resultado CONNECTION_BROKEN que con MODE(QUIESCE).

Conceptos relacionados

Canales

Trabajar con colas locales

Esta sección contiene ejemplos de algunos mandatos MQSC que puede utilizar para gestionar colas locales, modelo y alias.

Consulte la sección Mandatos MQSC para obtener información detallada acerca de estos mandatos.

Referencia relacionada

Restricciones de nomenclatura para las colas

Restricciones de nomenclatura para los otros objetos

Definición de una cola local con DEFINE QLOCAL

Para una aplicación, el gestor de colas local es el gestor de colas al que está conectada la aplicación. Las colas que están gestionadas por el gestor de colas local se dice que son locales respecto a ese gestor de colas. El mandato MQSC **DEFINE QLOCAL** se utiliza para crear una cola local.

Antes de empezar

Nota: Los pasos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Acerca de esta tarea

El mandato MQSC **DEFINE QLOCAL** se utiliza para crear una cola local. También puede utilizar el valor predeterminado incluido en la definición de la cola local predeterminada, o puede modificar las características de la cola de las existentes para la cola local predeterminada.

Nota: El nombre de la cola local predeterminada es SYSTEM.DEFAULT.LOCAL.QUEUE y se crea durante la instalación del sistema.

Procedimiento

- Para crear una cola local, especifique el mandato **DEFINE QLOCAL**, tal como se muestra en el ejemplo siguiente.

  En este ejemplo, el mandato **DEFINE QLOCAL** define una cola llamada ORANGE.LOCAL.QUEUE con estas características:

  – Está habilitada para operaciones get, para operaciones put y funciona de acuerdo con un orden de prioridades.
  – Es una cola *normal*, es decir, no es una cola de inicio ni una cola de transmisión y no genera mensajes mensaje desencadenantes.
  – La profundidad de cola máxima es 5.000 mensajes; la longitud máxima de un mensaje es de 4.194.304 bytes.

```
DEFINE QLOCAL(ORANGE.LOCAL.QUEUE) +
       DESCR('Queue for messages from other systems') +
       PUT(ENABLED) +
       GET(ENABLED) +
       NOTRIGGER +
       MSGDLVSQ(PRIORITY) +
       MAXDEPTH(5000) +
       MAXMSGL(4194304) +
       USAGE(NORMAL)
```

  Notas:

  1. Salvo el valor para la descripción, todos los valores de atributo que se muestran en el ejemplo son valores predeterminados. Estos ejemplos están incluidos con fines de ilustración. Puede omitirlos si está seguro de que los valores predeterminados son los que desea o no han sido modificados. Consulte también “Visualización de atributos de objeto predeterminados con DISPLAY QUEUE” en la página 143.
  2. **USAGE(NORMAL)** indica que esta cola no es una cola de transmisión.
  3. Si ya hay una cola local llamada ORANGE.LOCAL.QUEUE en el mismo gestor de colas, este mandato no podrá ejecutarse. Utilice el atributo **REPLACE** si desea sobrescribir la definición existente de una cola, pero consulte también “Cambio de los atributos de cola local con ALTER QLOCAL o DEFINE QLOCAL” en la página 145.

Referencia relacionada

DEFINE QLOCAL

Visualización de atributos de objeto predeterminados con DISPLAY QUEUE

Puede utilizar el mandato MQSC de **DISPLAY QUEUE** para visualizar los atributos que se tomaron del objeto predeterminado cuando se definió un objeto IBM MQ .

Antes de empezar

Nota: Los pasos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Acerca de esta tarea

Cuando define un objeto de IBM MQ, éste toma del objeto predeterminado todos los atributos que no especifique. Por ejemplo, cuando define una cola local, la cola hereda todos los atributos que omita en la definición de la cola local predeterminada, que tiene el nombre SYSTEM.DEFAULT.LOCAL.QUEUE. Puede utilizar el mandato **DISPLAY QUEUE** para ver exactamente cuáles son estos atributos.

Procedimiento

- Para visualizar los atributos de objeto predeterminados para una cola local, utilice el mandato siguiente:

```
DISPLAY QUEUE (SYSTEM.DEFAULT.LOCAL.QUEUE)
```

  La sintaxis del mandato **DISPLAY** es distinta de la del correspondiente mandato **DEFINE**. En el mandato **DISPLAY** basta con dar el nombre de cola, mientras que en el mandato **DEFINE** hay que especificar el tipo de cola, es decir, QLOCAL, QALIAS, QMODEL o QREMOTE.

  Puede visualizar de forma selectiva los atributos especificándolos individualmente. Por ejemplo:

```
DISPLAY QUEUE (ORANGE.LOCAL.QUEUE) +
MAXDEPTH +
MAXMSGL +
CURDEPTH;
```

  Este mandato muestra los tres atributos especificados de la siguiente manera:

```
AMQ8409: Display Queue details.
QUEUE(ORANGE.LOCAL.QUEUE)        TYPE(QLOCAL)
CURDEPTH(0)                MAXDEPTH(5000)
MAXMSGL(4194304)
```

  CURDEPTH es la profundidad de cola actual, es decir, el número de mensajes que hay en la cola. Visualizar este atributo resulta de gran utilidad, ya que puede utilizarlo para supervisar la profundidad de cola y asegurarse de que no se llena.

Referencia relacionada

DISPLAY QUEUE

DEFINE colas

Copia de una definición de cola local con DEFINE QLOCAL

Puede copiar una definición de cola utilizando el atributo **LIKE** en el mandato MQSC **DEFINE QLOCAL** .

Antes de empezar

Nota: Los pasos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Acerca de esta tarea

Puede utilizar el mandato **DEFINE** con el atributo **LIKE** para crear una cola que tiene los mismos atributos que la cola especificada, en vez de tener los atributos de la cola local predeterminada del sistema. También puede utilizar esta forma del mandato **DEFINE** para copiar una definición de cola y sustituir una o más modificaciones en los atributos de la original.

Notas:

1. Cuando se utiliza el atributo **LIKE** en un mandato **DEFINE**, solo se copian los atributos de la cola. No se copian los mensajes existente en la cola.
2. Si define una cola local, sin especificar **LIKE**, es lo mismo que:

```
DEFINE LIKE(SYSTEM.DEFAULT.LOCAL.QUEUE)
```

Procedimiento

- Para crear una cola con los mismos atributos que la cola especificada, en lugar de los de la cola local predeterminada del sistema, especifique el mandato **DEFINE**, tal como se muestra en el ejemplo siguiente.

  Escriba el nombre de la cola que se va a copiar exactamente como fue especificada al crearla. Si el nombre contiene caracteres en minúscula, especifique el nombre entre comillas simples.

  Este ejemplo crea una cola que tiene los mismos atributos que la cola ORANGE.LOCAL.QUEUE, en vez de tener los atributos de la cola local predeterminada del sistema:

```
DEFINE QLOCAL (MAGENTA.QUEUE) +
LIKE (ORANGE.LOCAL.QUEUE)
```

- Para copiar una definición de cola, pero sustituir uno o más cambios en los atributos del original, especifique el mandato **DEFINE**, tal como se muestra en el ejemplo siguiente.

  Este mandato copia los atributos de la cola ORANGE.LOCAL.QUEUE en la cola THIRD.QUEUE, pero especifica que la longitud máxima de mensaje en la nueva cola será 1024 bytes en lugar de 4194304:

```
DEFINE QLOCAL (THIRD.QUEUE) +
LIKE (ORANGE.LOCAL.QUEUE) +
MAXMSGL(1024);
```

Referencia relacionada

DEFINE colas

Cambio de los atributos de cola local con ALTER QLOCAL o DEFINE QLOCAL

Puede cambiar los atributos de cola de dos maneras, utilizando **ALTER QLOCAL** o el mandato MQSC **DEFINE QLOCAL** con el atributo **REPLACE** .

Antes de empezar

Nota: Los pasos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Acerca de esta tarea

Puede utilizar el atributo **REPLACE** de los mandatos **ALTER** y **DEFINE** para sustituir una definición existente por la definición nueva especificada. Esta diferencia entre utilizar **ALTER** y **DEFINE** radica en que **ALTER** con **REPLACE** no cambia los parámetros no especificados, pero **DEFINE** con **REPLACE** establece todos los parámetros.

Procedimiento

- Para cambiar los atributos de cola, utilice el mandato **ALTER** o el mandato **DEFINE**, tal como se muestra en los ejemplos siguientes.

  En estos ejemplos, la longitud máxima de mensaje en la cola ORANGE.LOCAL.QUEUE se reduce a 10.000 bytes.

– Utilizando el mandato **ALTER**:

```
ALTER QLOCAL (ORANGE.LOCAL.QUEUE) MAXMSGL(10000)
```

Este mandato modifica un solo atributo, el de la longitud máxima del mensaje; todos los demás atributos no cambian.

– Usando el mandato **DEFINE** con la opción **REPLACE**, por ejemplo: example:

```
DEFINE QLOCAL (ORANGE.LOCAL.QUEUE) MAXMSGL(10000) REPLACE
```

Este mandato no sólo modifica la longitud máxima del mensaje, sino también todos los demás atributos, a los que se les asignan sus valores predeterminados. De modo que, por ejemplo, si anteriormente se inhibió la cola, esto se cambia para habilitarla porque este es el valor predeterminado, como se ha especificado en la cola SYSTEM.DEFAULT.LOCAL.QUEUE.

Si se reduce la longitud máxima del mensaje en una cola existente, los mensajes existentes no se ven afectados. Sin embargo, todos los mensajes nuevos deben cumplir los nuevos criterios.

Referencia relacionada
Colas ALTER
ALTER QLOCAL
DEFINE colas
DEFINE QLOCAL

Borrado de una cola local con CLEAR QLOCAL

Puede utilizar el mandato MQSC de **CLEAR QLOCAL** para borrar una cola local.

Antes de empezar

Nota: Los pasos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Una cola no se puede vaciar si:

- Hay mensajes no confirmados que se han transferido a la cola bajo punto de sincronización.
- Hay una aplicación que tiene abierta actualmente la cola.

Acerca de esta tarea

Si desea borrar una cola local utilizando el mandato **CLEAR QLOCAL**, el nombre de la cola debe estar definido en el gestor de colas local.

Nota: No hay ningún mensaje de solicitud que le permita cambiar de opinión, por lo tanto, cuando pulse la tecla Intro se perderán los mensajes.

Procedimiento

Para borrar los mensajes de una cola local, utilice el mandato **CLEAR QLOCAL**, tal como se muestra en el ejemplo siguiente.

En este ejemplo, todos los mensajes se suprimen de una cola local llamada MAGENTA.QUEUE:

```
CLEAR QLOCAL (MAGENTA.QUEUE)
```

Referencia relacionada
CLEAR QLOCAL

Supresión de una cola local con DELETE QLOCAL

Puede utilizar el mandato MQSC de **DELETE QLOCAL** para suprimir una cola local.

Antes de empezar

Nota: Los pasos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Una cola no puede suprimirse si contiene mensajes no confirmados.

Si una cola tiene uno o varios mensajes confirmados y ningún mensaje sin confirmar, solo se puede borrar si se especifica la opción **PURGE**. La supresión prosigue incluso si hay mensajes confirmados en la cola indicada y estos mensajes también se depuran.

La especificación de **NOPURGE** en lugar de **PURGE** asegura que la cola no se borre si contiene algún mensaje confirmado.

Procedimiento

- Para suprimir una cola local, utilice el mandato **DELETE QLOCAL**, tal como se muestra en el ejemplo siguiente.

  En este ejemplo se suprime la cola PINK.QUEUE si no hay mensajes confirmados en la cola:

  ```
  DELETE QLOCAL (PINK.QUEUE) NOPURGE
  ```

  En este ejemplo se suprime la cola PINK.QUEUE incluso si hay mensajes confirmados en la cola:

  ```
  DELETE QLOCAL (PINK.QUEUE) PURGE
  ```

Referencia relacionada

DELETE QLOCAL

Examinar colas con el programa de ejemplo

IBM MQ proporciona un examinador de colas de ejemplo que puede utilizar para examinar el contenido de los mensajes de una cola.

Acerca de esta tarea

El navegador se proporciona en los formatos fuente y ejecutable en las ubicaciones siguientes, donde *MQ_INSTALLATION_PATH* representa el directorio de alto nivel en el que se ha instalado IBM MQ.

Windows En Windows, los nombres de archivo y las rutas del navegador de colas de ejemplo son los siguientes:

Origen
MQ_INSTALLATION_PATH\tools\c\samples\

Ejecutable
MQ_INSTALLATION_PATH\tools\c\samples\bin\amqsbcg.exe

Linux AIX En AIX and Linux, los nombres de archivo y las vías de acceso son los siguientes:

Origen
MQ_INSTALLATION_PATH/samp/amqsbcg0.c

Ejecutable
MQ_INSTALLATION_PATH/samp/bin/amqsbcg

Procedimiento

- Para ejecutar el programa de ejemplo, especifique un mandato como se muestra en el ejemplo siguiente.

  El programa de ejemplo requiere dos parámetros de entrada, el nombre de la cola en la que se van a examinar los mensajes y el gestor de colas que es el propietario de dicha cola. Por ejemplo:

```
amqsbcg SYSTEM.ADMIN.QMGREVENT.tpp01 saturn.queue.manager
```

Resultados

Los resultados típicos de este mandato se muestran en el ejemplo siguiente:

```
AMQSBCG0 - starts here
**********************

 MQOPEN - 'SYSTEM.ADMIN.QMGR.EVENT'


 MQGET of message number 1
****Message descriptor****

  StrucId  : 'MD  '  Version : 2
  Report   : 0  MsgType : 8
  Expiry   : -1  Feedback : 0
  Encoding : 546  CodedCharSetId : 850
  Format : 'MQEVENT '
  Priority : 0  Persistence : 0
  MsgId : X'414D512073617475726E2E71756575650005D30033563DB8'
  CorrelId : X'000000000000000000000000000000000000000000000000'
  BackoutCount : 0
  ReplyToQ       : '                                                '
  ReplyToQMgr    : 'saturn.queue.manager                           '
  ** Identity Context
  UserIdentifier : '            '
  AccountingToken :
   X'0000000000000000000000000000000000000000000000000000000000000000'
  ApplIdentityData : '                                        '
  ** Origin Context
  PutApplType    : '7'
  PutApplName    : 'saturn.queue.manager        '
  PutDate  : '19970417'    PutTime  : '15115208'
  ApplOriginData : '    '

  GroupId : X'000000000000000000000000000000000000000000000000'
  MsgSeqNumber   : '1'
  Offset         : '0'
  MsgFlags       : '0'
  OriginalLength : '104'

****  Message   ****

length - 104 bytes

00000000: 0700 0000 2400 0000 0100 0000 2C00 0000 '....→.......,...'
00000010: 0100 0000 0100 0000 0100 0000 AE08 0000 '................'
00000020: 0100 0000 0400 0000 4400 0000 DF07 0000 '........D.......'
00000030: 0000 0000 3000 0000 7361 7475 726E 2E71 '....0...saturn.q'
00000040: 7565 7565 2E6D 616E 6167 6572 2020 2020 'ueue.manager    '
00000050: 2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000060: 2020 2020 2020 2020                     '                '

No more messages
MQCLOSE
MQDISC
```

Referencia relacionada

El programa de ejemplo del navegador

Habilitar colas grandes

IBM MQ da soporte a colas de más de 2 TB.

Windows En sistemas Windows, el soporte para archivos grandes está disponible sin ninguna habilitación adicional.

Linux AIX En sistemas AIX and Linux , debe habilitar explícitamente el soporte de archivos grandes para poder crear archivos de cola de varios gigabytes o terabytes. Consulte la documentación del sistema operativo para obtener información sobre cómo llevar a cabo esta operación.

Algunos programas de utilidad, como tar, no pueden hacer frente a archivos de varios gigabytes o terabytes. Antes de habilitar el soporte de archivos grandes, consulte la documentación del sistema operativo para ver información sobre las restricciones en los programas de utilidad que utilice.

Para obtener información sobre la planificación de la cantidad de almacenamiento necesario para las colas, consulte Documentos de rendimiento de MQ para informes de rendimiento específicos de plataforma.

Puede controlar el tamaño de los archivos de cola utilizando un nuevo atributo en colas locales y de modelo. Consulte “Modificación de archivos de cola de IBM MQ” en la página 149 para obtener más información.

Multi Modificación de archivos de cola de IBM MQ

Puede controlar el tamaño de los archivos de cola utilizando un atributo en colas locales y de modelo. Puede visualizar el tamaño actual de un archivo de cola y el tamaño máximo hasta el que puede crecer actualmente (en función del tamaño de bloque en uso actualmente en ese archivo), utilizando dos nuevos atributos de estado de cola.

Atributo utilizado para modificar los archivos de colas

El atributo en las colas locales y de modelo es:

MAXFSIZE
Denota el tamaño máximo del archivo de cola utilizado por la cola, en megabytes.

Puede establecer o visualizar el valor de este atributo utilizando mandatos MQSC, IBM MQ Explorero administrative REST API. También puede visualizar el valor de este atributo en IBM MQ Console.

Consulte MAXFSIZE y “Cambio del tamaño de un archivo de cola de IBM MQ” en la página 150 para obtener más información.

El equivalente PCF de este atributo es **MQIA_MAX_Q_FILE_SIZE**. Consulte Cambiar, copiar y crear cola.

Los dos atributos sobre el estado de la cola son:

CURFSIZE
Muestra el tamaño actual del archivo de cola en megabytes, redondeado hasta el megabyte más cercano.

Puede establecer o visualizar el valor de este atributo utilizando mandatos MQSC, IBM MQ Explorero administrative REST API.

Consulte CURFSIZE para obtener más información.

El equivalente PCF de este atributo es **MQIA_CUR_Q_FILE_SIZE**. Consulte Consultar cola y Consultar cola (respuesta).

CURMAXFS
Indica el tamaño máximo actual hasta el que puede crecer el archivo de cola, redondeado al megabyte más cercano, dado el tamaño de bloque actual en uso en una cola.

Puede establecer o visualizar el valor de este atributo utilizando mandatos MQSC, IBM MQ Explorero administrative REST API.

Consulte CURMAXFS para obtener más información.

El equivalente PCF de este atributo es **MQIA_CUR_MAX_FILE_SIZE**. Consulte Consultar cola y Consultar cola (respuesta).

Tamaño de bloque y granularidad

Los archivos de cola se dividen en segmentos denominados bloques. Para aumentar el tamaño máximo de un archivo de cola, es posible que el gestor de colas tenga que cambiar el tamaño de bloque o la granularidad de la cola.

Si se crea una cola definida recientemente con un valor **MAXFSIZE** grande, la cola se crea con un tamaño de bloque adecuado. Sin embargo, si una cola existente tiene su valor **MAXFSIZE** aumentado, por ejemplo utilizando el mandato MQSC **ALTER QLOCAL** , es posible que sea necesario permitir que la cola se vacíe para que el gestor de colas vuelva a configurar la cola.

Consulte “Cálculo de la cantidad de datos que un archivo de cola IBM MQ puede almacenar” en la página 151 para obtener más información.

Atención: Algunos sistemas de archivos y sistemas operativos tienen límites sobre el tamaño de todo el sistema de archivos o el tamaño de un archivo individual. Debe comprobar los límites en los sistemas que utiliza su empresa.

Referencia relacionada

ALTER QUEUES

DISPLAY QUEUE

DISPLAY QSTATUS

Multi *Cambio del tamaño de un archivo de cola de IBM MQ*

Puede aumentar o disminuir el tamaño máximo de un archivo de cola.

Antes de empezar

Nota: Los pasos de esta tarea requieren que ejecute mandatos MQSC:

- ALW EnAIX, Linux, and Windows , emite comandos MQSC desde un**runmqsc** símbolo del sistema. Ver Ejecutar comandos MQSC de forma interactiva en**runmqsc** y Ejecutar comandos MQSC desde archivos de texto en**runmqsc** .
- IBM i EnIBM i , crea una lista de comandos en un archivo de secuencia de comandos y luego ejecuta el archivo usando el**STRMQMMQSC** dominio. Ver Administración mediante comandos MQSC enIBM i .

Antes de establecer un nuevo tamaño para un archivo de cola, utilice el mandato MQSC DISPLAY QLOCAL para ver el tamaño del archivo de cola que desea cambiar. Por ejemplo, emita el mandato siguiente:

```
DISPLAY QLOCAL(SYSTEM.DEFAULT.LOCAL.QUEUE) MAXFSIZE
```

Recibirá la siguiente salida:

```
AMQ8409I: Display queue details
  QUEUE(SYSTEM.DEFAULT.LOCAL.QUEUE)        TYPE(QLOCAL)
  MAXFSIZE(DEFAULT)
```

que muestra que el tamaño máximo del archivo de cola es el valor predeterminado de 2.088.960 MB.

Acerca de esta tarea

En los procedimientos siguientes se muestra cómo:

- Reducir el tamaño máximo hasta el que puede crecer un archivo de cola.

- Aumentar el tamaño máximo hasta el que puede crecer un archivo de cola.

Atención: debe tener cuidado al aumentar el tamaño de los archivos de cola sin considerar la forma en que están grabadas las aplicaciones y el posible efecto sobre el rendimiento. El acceso a los mensajes de forma aleatoria en un archivo de cola muy grande puede ser muy lento.

Si está considerando aumentar el tamaño máximo de un archivo de cola más allá del valor predeterminado, debe tener cuidado de utilizar selectores de mensajes como, por ejemplo, los

ID de correlación y las series de selector IBM MQ classes for JMS JM 3.0 o IBM MQ classes for Jakarta Messaging . Los archivos de cola más grandes son más adecuados para el acceso primero en entrar, primero en salir a la cola.

Solo se deben tener cantidades muy grandes de datos en archivos de cola individuales en gestores de colas que están configurados para el registro circular o en los que no se ha habilitado la creación de imágenes de soporte para la cola individual.

No debe limitar el tamaño de las colas SYSTEM, ya que esto podría afectar al funcionamiento del gestor de colas.

Procedimiento

1. Reduzca el tamaño de archivo de cola máximo
   a) Emita el siguiente mandato MQSC para crear un archivo local denominado SMALLQUEUE, con un tamaño de 500 gigabytes:

```
DEFINE QLOCAL(SMALLQUEUE) MAXFSIZE(512000)
     2 : DEFINE QLOCAL(SMALLQUEUE) MAXFSIZE(512000)
AMQ8006I: IBM MQ queue created
```

   y recibe el mensaje: AMQ8006I:

   Nota: si configura una cola con un valor menor que la cantidad de datos que ya hay en el archivo, los nuevos mensajes no se podrán poner en la cola.

   Si una aplicación intenta poner un mensaje en un archivo de cola que no tiene espacio suficiente, la aplicación recibe el código de retorno MQRC_Q_SPACE_NOT_AVAILABLE. Cuando se leen suficientes mensajes con lectura destructiva de la cola, las aplicaciones pueden empezar a colocar mensajes nuevos en la cola.
2. Aumente el tamaño máximo del archivo de cola.
   a) Emita el siguiente mandato MQSC para crear un archivo local denominado LARGEQUEUE, con un tamaño de 5 terabytes:

```
DEFINE QLOCAL(LARGEQUEUE) MAXFSIZE(5242880)
     3 : DEFINE QLOCAL(LARGEQUEUE) MAXFSIZE(5242880)
AMQ8006I: IBM MQ queue created
```

Multi *Cálculo de la cantidad de datos que un archivo de cola IBM MQ puede almacenar*

La cantidad de datos que se pueden almacenar en una cola está limitada por el tamaño de los bloques individuales en los que se divide la cola. Utilice los mandatos MQSC para confirmar el tamaño de bloque y la granularidad, y compruebe el tamaño de un archivo de cola.

Antes de empezar

Nota: Los pasos de esta tarea requieren que ejecute mandatos MQSC:

- ALW EnAIX, Linux, and Windows , emite comandos MQSC desde un**runmqsc** símbolo del sistema. Ver Ejecutar comandos MQSC de forma interactiva en**runmqsc** y Ejecutar comandos MQSC desde archivos de texto en**runmqsc** .
- IBM i EnIBM i , crea una lista de comandos en un archivo de secuencia de comandos y luego ejecuta el archivo usando el**STRMQMMQSC** dominio. Ver Administración mediante comandos MQSC enIBM i .

Procedimiento

- Confirme el tamaño de bloque y la granularidad.

  El tamaño de bloque predeterminado es 512 bytes. Para admitir archivos de cola mayores de dos terabytes, el gestor de colas tendrá que aumentar el tamaño de bloque.

  El tamaño de bloque se calcula automáticamente cuando se configura MAXFSIZE para una cola, pero el tamaño de bloque revisado no se puede aplicar a la cola si la cola ya contiene mensajes. Una vez que una cola está vacía, el gestor de colas modifica automáticamente el tamaño de bloque para dar soporte al **MAXFSIZE**configurado.

  El mandato **DISPLAY QSTATUS** tiene un atributo nuevo, CURMAXFS, que le permite confirmar que se ha modificado una cola para utilizar un tamaño de bloque nuevo.

  En el ejemplo siguiente, el valor **CURMAXFS** de 4177920 confirma que el archivo de cola puede crecer actualmente a aproximadamente cuatro terabytes de tamaño. Si el valor de **MAXFSIZE** configurado en la cola es mayor que el valor de **CURMAXFS**, el gestor de colas sigue esperando a que se vacíe la cola antes de volver a configurar el tamaño de bloque del archivo de cola.

  ```
  DISPLAY QSTATUS(LARGEQUEUE)  CURMAXFS
       2 : DISPLAY QSTATUS(LARGEQUEUE)  CURMAXFS
  AMQ8450I: Display queue status details
    QUEUE(LARGEQUEUE)                       TYPE(QUEUE)
    CURMAXFS(4177920)                       CURDEPTH(100000)
  ```

- Compruebe el tamaño de un archivo de cola.

  Puede visualizar el tamaño actual de un archivo de cola en disco, en megabytes, utilizando el atributo CURFSIZE en el mandato **DISPLAY QSTATUS** . Esto puede ser útil en plataformas como IBM MQ Appliance, donde no es posible acceder al sistema de archivos directamente.

  ```
  DISPLAY QSTATUS(SMALLQUEUE)  CURFSIZE
       1 : DISPLAY QSTATUS(SMALLQUEUE)  CURFSIZE
  AMQ8450I: Display queue status details
    QUEUE(SMALLQUEUE)                       TYPE(QUEUE)
    CURDEPTH(4024)                          CURFSIZE(10)
  ```

  Nota: Cuando una cola tiene mensajes eliminados, el atributo **CURFSIZE** no disminuye inmediatamente.

  Normalmente, el espacio en un archivo de cola sólo se libera cuando ninguna aplicación tiene la cola abierta y no hay mensajes pendientes almacenados en la cola. Cualquier truncamiento o compactación necesaria de un archivo de cola cargado por el gestor de colas se produce durante el punto de comprobación, la conclusión del gestor de colas o mientras se graba una imagen de soporte de la cola.

Referencia relacionada
ALTER QUEUES
DISPLAY QSTATUS

Trabajar con colas remotas

Una cola remota es una definición de un gestor de colas local que hace referencia a una cola de un gestor de colas remoto. No es necesario definir una cola remota desde una posición local, pero si lo hace, las

aplicaciones pueden hacer referencia a la cola remota por su nombre definido localmente en lugar de tener que especificar un nombre calificado por el ID del gestor de colas en el que se encuentra la cola remota.

Antes de empezar

Nota: Los pasos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Acerca de esta tarea

Una aplicación se conecta a un gestor de colas local y luego emite una llamada MQOPEN. En la llamada de apertura, el nombre de cola especificado es el de una definición de cola remota existente en el gestor de colas local. La definición de cola remota suministra los nombres de la cola de destino, del gestor de colas de destino y, opcionalmente, de una cola de transmisión. Para transferir un mensaje a una cola remota, la aplicación emite una llamada MQPUT especificando el identificador devuelto en la llamada MQOPEN. El gestor de colas utiliza el nombre de la cola remota y el del gestor de colas remoto en una cabecera de transmisión que se añade al principio del mensaje. Esta información se utiliza para dirigir el mensaje a su destino correcto en la red.

Como administrador, puede controlar el destino del mensaje alterando la definición de cola remota.

Procedimiento

- Coloque un mensaje en una cola propiedad de un gestor de colas remoto.

  La aplicación se conecta a un gestor de colas, por ejemplo, `saturn.queue.manager`. La cola de destino es propiedad de otro gestor de colas.

  En la llamada MQOPEN, la aplicación especifica estos campos:

Valor de campo	Descripción
ObjectName CYAN.REMOTE.QUEUE	Especifica el nombre local del objeto de cola remota. Esto define la cola de destino y el gestor de colas de destino.
ObjectType (Cola)	Identifica este objeto como una cola.
ObjectQmgrName En blanco o `saturn.queue.manager`	Este campo es opcional. Si está en blanco, se presupone el nombre del gestor de colas local. (Es el gestor de colas donde se encuentra la definición de cola remota).

  A continuación, la aplicación emite una llamada MQPUT para colocar un mensaje en esta cola.

  En el gestor de colas local, puede crear una definición local de una cola remota utilizando los siguientes mandatos MQSC:

```
DEFINE QREMOTE (CYAN.REMOTE.QUEUE) +
DESCR ('Queue for auto insurance requests from the branches') +
RNAME (AUTOMOBILE.INSURANCE.QUOTE.QUEUE) +
RQMNAME (jupiter.queue.manager) +
XMITQ (INQUOTE.XMIT.QUEUE)
```

  donde:

  QREMOTE (CYAN.REMOTE.QUEUE)
  Especifica el nombre local del objeto de cola remota. Este es el nombre que las aplicaciones conectadas a este gestor de colas deben especificar en la llamada MQOPEN para abrir la cola AUTOMOBILE.INSURANCE.QUOTE.QUEUE del gestor de colas remoto `jupiter.queue.manager`.

DESCR ('Queue for auto insurance requests from the branches')
Proporciona un texto adicional que describe el uso de la cola.

RNAME (AUTOMOBILE.INSURANCE.QUOTE.QUEUE)
Especifica el nombre de la cola de destino del gestor de colas remoto. Ésta es la cola de destino real para los mensajes enviados por aplicaciones que especifiquen el nombre de cola CYAN.REMOTE.QUEUE. La cola AUTOMOBILE.INSURANCE.QUOTE.QUEUE debe definirse como una cola local en el gestor de colas remoto.

RQMNAME (jupiter.queue.manager)
Especifica el nombre del gestor de colas remoto al que pertenece la cola de destino AUTOMOBILE.INSURANCE.QUOTE.QUEUE.

XMITQ (INQUOTE.XMIT.QUEUE)
Especifica el nombre de la cola de transmisión. Es opcional; si no se especifica el nombre de una cola de transmisión, se utiliza una cola con el mismo nombre que el gestor de colas remoto.

En cualquier caso, la cola de transmisión adecuada debe definirse como una cola local con un atributo **Usage** que especifique que es una cola de transmisión (USAGE(XMITQ) en los mandatos MQSC).

- Colocar mensajes en una cola remota (método alternativo).

  Utilizar una definición local de una cola remota no es la única manera de colocar mensajes en una cola remota. Las aplicaciones pueden especificar el nombre de cola completo, incluido el nombre del gestor de colas remoto, como parte de la llamada MQOPEN. En este caso, no se requiere una definición local de una cola remota. No obstante, esto significa que las aplicaciones deben conocer o tener acceso al nombre del gestor de colas remoto durante la ejecución.
- Utilice otros mandatos con colas remotas.

  Puede utilizar mandatos MQSC para visualizar o alterar los atributos de un objeto cola remota, o puede suprimir el objeto de cola remota. Por ejemplo:

  – Para visualizar los atributos de la cola remota:

  ```
  DISPLAY QUEUE (CYAN.REMOTE.QUEUE)
  ```

  – Para cambiar la cola remota para habilitar las transferencias. Esto no afecta a la cola de destino, sólo a las aplicaciones que especifiquen esta cola remota:

  ```
  ALTER QREMOTE (CYAN.REMOTE.QUEUE) PUT(ENABLED)
  ```

  – Para suprimir esta cola remota. Esto no afecta a la cola de destino, sólo a su definición local:

  ```
  DELETE QREMOTE (CYAN.REMOTE.QUEUE)
  ```

  Nota: Cuando se suprime una cola remota, sólo se suprime la representación local de la cola remota. No se suprime la cola remota en sí ni ningún mensaje que se encuentre en ella.

Las definiciones de cola remota se pueden utilizar como alias

Además de localizar una cola en otro gestor de colas, también puede utilizar una definición local de una cola remota para los alias de gestor de colas y los alias de cola de respuesta. Ambos tipos de alias se resuelven mediante la definición local de una cola remota. Debe definir los canales adecuados para que el mensaje llegue a su destino.

Alias de gestor de colas

Un alias es el proceso por el cual el nombre del gestor de colas de destino, tal como se especifica en un mensaje, es modificado por un gestor de colas en la ruta del mensaje. Los alias de gestor de colas son importantes porque se pueden utilizar para controlar el destino de los mensajes dentro de una red de gestores de colas.

Para hacer esto debe alterar la definición de cola remota del gestor de colas en el punto de control. La aplicación emisora no sabe que el nombre de gestor de colas especificado es un alias.

Para obtener más información sobre los alias de gestor de colas, consulte ¿Qué son los alias?

Alias de cola de respuesta

De forma opcional, una aplicación puede especificar el nombre de una cola de respuesta cuando coloca un *mensaje de solicitud* en una cola.

Si la aplicación que procesa el mensaje extrae el nombre de la cola de respuesta, sabe dónde enviar el *mensaje de respuesta*, si fuera necesario.

Un alias de cola de respuesta es el proceso por el cual una cola de respuesta, tal como se especifica en un mensaje de solicitud, es modificada por un gestor de colas en la ruta del mensaje. La aplicación emisora no sabe que el nombre de cola de respuesta especificado es un alias.

Un alias de cola de respuesta le permite modificar el nombre de la cola de respuesta y, opcionalmente, de su gestor de colas. Esto, a su vez, le permite controlar qué ruta se utiliza para los mensajes de respuesta.

Para obtener más información sobre mensajes de solicitud, mensajes de respuesta y colas de respuesta, consulte las secciones Tipos de mensajes y Cola de respuesta y gestor de colas.

Para obtener más información sobre las alias de cola de respuesta, consulte Alias de cola de respuesta y clústeres.

Trabajar con colas alias

Puede definir una cola alias para hacer referencia indirectamente a otra cola o tema.

Antes de empezar

Nota: Los pasos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Atención: Las listas de distribución no admiten el uso de cola alias que apuntan a objetos de tema. Si una cola de alias apunta a un objeto de tema en una lista de distribución, IBM MQ devuelve `MQRC_ALIAS_BASE_Q_TYPE_ERROR`.

Acerca de esta tarea

La cola a la que una cola alias hace referencia puede ser cualquiera de las siguientes:

- Una cola local (consulte el apartado “Definición de una cola local con DEFINE QLOCAL” en la página 142).
- Una definición local de una cola remota (consulte el apartado “Trabajar con colas remotas” en la página 152).
- Un tema.

Una cola alias no es una cola real, sino una definición que se resuelve en una cola real (o de destino) en tiempo de ejecución. La definición de la cola alias especifica la cola de destino. Cuando una aplicación efectúa una llamada `MQOPEN` a una cola alias, el gestor de colas resuelve el alias en el nombre de la cola de destino.

Una cola alias no se puede resolver en otra cola alias definida localmente. Sin embargo, una cola alias puede resolverse como colas alias definidas en otro lugar de los clústeres de los cuales forme parte el gestor de colas local. En Resolución de nombres hallará más información.

Las colas alias son útiles para:

- Proporcionar a las aplicaciones niveles diferentes de autorizaciones de acceso a la cola de destino.
- Permitir que diferentes aplicaciones trabajen con la misma cola de diferentes modos. (Quizá desee asignar prioridades predeterminadas diferentes o valores de persistencia predeterminados diferentes).

- Simplificar el mantenimiento, la migración y el equilibrio de carga. (Quizá desee cambiar el nombre de la cola de destino sin tener que cambiar la aplicación, que sigue utilizando el alias).

Por ejemplo, suponga que se ha desarrollado una aplicación para colocar mensajes en una cola denominada MY.ALIAS.QUEUE. La aplicación especifica el nombre de esta cola cuando realiza una solicitud MQOPEN e, indirectamente, si transfiere un mensaje a esta cola. La aplicación no sabe que la cola es una cola alias. Para cada llamada MQI que utilice este alias, el gestor de colas determina el nombre de cola real, que podría ser una cola local o una cola remota definida en este gestor de colas.

Al cambiar el valor del atributo TARGET, puede redirigir las llamadas MQI a otra cola, posiblemente en otro gestor de colas. Esto resulta de gran utilidad para el mantenimiento, la migración y el equilibrio de la carga.

Procedimiento

- Defina una cola alias.

  El siguiente mandato MQSC crea una cola alias:

```
DEFINE QALIAS (MY.ALIAS.QUEUE) TARGET (YELLOW.QUEUE)
```

  Este mandato redirige las llamadas MQI que especifican MY.ALIAS.QUEUE a la cola YELLOW.QUEUE. Este mandato no crea la cola de destino; las llamadas MQI fracasan si la cola YELLOW.QUEUE no existe en tiempo de ejecución.

  Si cambia la definición del alias, puede redirigir las llamadas MQI a otra cola. Por ejemplo:

```
ALTER QALIAS (MY.ALIAS.QUEUE) TARGET (MAGENTA.QUEUE)
```

  Este mandato redirige las llamadas MQI a otra cola, MAGENTA.QUEUE.

  También puede utilizar colas alias para que hacer que una sola cola (la cola destino) parezca tener atributos distintos para aplicaciones distintas. Esto se hace definiendo dos alias, uno para cada aplicación. Suponga que tiene dos aplicaciones:

  – La aplicación ALPHA puede transferir mensajes a YELLOW.QUEUE, pero no tiene autorización para obtener mensajes de dicha cola.
  – La aplicación BETA puede obtener mensajes de YELLOW.QUEUE, pero no tiene autorización para colocar mensajes en ella.

  El siguiente mandato MQSC define un alias que se coloca habilitado y se obtiene inhabilitado para la aplicación ALPHA:

```
DEFINE QALIAS (ALPHAS.ALIAS.QUEUE) +
TARGET (YELLOW.QUEUE) +
PUT (ENABLED) +
GET (DISABLED)
```

  El mandato siguiente define un alias que está inhabilitado para operaciones de transferencia y habilitado para operaciones de obtención para la aplicación BETA:

```
DEFINE QALIAS (BETAS.ALIAS.QUEUE) +
TARGET (YELLOW.QUEUE) +
PUT (DISABLED) +
GET (ENABLED)
```

  ALPHA utiliza el nombre de cola ALPHAS.ALIAS.QUEUE en sus llamadas MQI; BETA utiliza el nombre de cola BETAS.ALIAS.QUEUE. Ambas aplicaciones acceden a la misma cola, pero de forma diferente.

  Puede utilizar los atributos LIKE y REPLACE cuando defina alias de colas, del mismo modo que los utiliza con colas locales.

- Utilice otros mandatos con colas alias.

  Puede utilizar los mandatos MQSC adecuados para visualizar o alterar atributos de la cola alias o para suprimir el objeto de cola alias. Por ejemplo:

  Utilice el mandato **DISPLAY QALIAS** para visualizar los atributos de la cola alias:

  ```
  DISPLAY QALIAS (ALPHAS.ALIAS.QUEUE)
  ```

  Utilice el mandato **ALTER QALIAS** para modificar el nombre de cola base al que se resuelve el alias, donde la opción `force` fuerza el cambio incluso si la cola está abierta:

  ```
  ALTER QALIAS (ALPHAS.ALIAS.QUEUE) TARGET(ORANGE.LOCAL.QUEUE) FORCE
  ```

  Use el mandato **DELETE QALIAS** para borrar este alias de cola:

  ```
  DELETE QALIAS (ALPHAS.ALIAS.QUEUE)
  ```

  No se puede suprimir una cola alias si una aplicación tiene actualmente abierta la cola.

Conceptos relacionados
Listas de distribución
Referencia relacionada
ALTER QALIAS
DEFINE QALIAS
DELETE QALIAS

Trabajar con colas modelo

Las colas modelo proporcionan un método práctico para que las aplicaciones puedan crear colas cuando se necesitan.

Antes de empezar

Nota: Los pasos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Acerca de esta tarea

Un gestor de colas crea una *cola dinámica* si recibe una llamada MQI de una aplicación que especifica un nombre de cola que se ha definido como cola modelo. El nombre de la nueva cola dinámica lo genera el gestor de colas cuando se crea la cola. Una *cola modelo* es una plantilla que especifica los atributos de las colas dinámicas creadas a partir de ella.

Procedimiento

- Defina una cola modelo.

  Utilice el mandato MQSC de **DEFINE QMODEL** para definir una cola modelo con un conjunto de atributos de la misma forma que define una cola local. Las colas modelo y las colas locales tienen el mismo conjunto de atributos, excepto que en las colas modelo se puede especificar si las colas dinámicas creadas son temporales o persistentes. (Las colas persistentes se conservan tras reiniciar el gestor de colas, las colas temporales no). Por ejemplo:

  ```
  DEFINE QMODEL (GREEN.MODEL.QUEUE) +
  DESCR('Queue for messages from application X') +
  PUT (DISABLED) +
  GET (ENABLED) +
  NOTRIGGER +
  MSGDLVSQ (FIFO) +
  ```

```
MAXDEPTH (1000) +
MAXMSGL (2000) +
USAGE (NORMAL) +
DEFTYPE (PERMDYN)
```

Este mandato crea una definición de cola modelo. Por el atributo **DEFTYPE** se puede ver que las colas creadas a partir de esta plantilla son colas dinámicas permanentes. Todos los atributos no especificados se copian automáticamente de la cola predeterminada SYSTEM.DEFAULT.MODEL.QUEUE.

Se pueden usar los atributos **LIKE** y **REPLACE** cuando se definen colas modelo de la misma forma en que se usan con las colas locales.

- Utilice otros mandatos con colas modelo.

  Puede utilizar los mandatos MQSC adecuados para visualizar o alterar los atributos de una cola modelo o para suprimir el objeto de cola modelo. Por ejemplo:

  Use el mandato **DISPLAY QUEUE** para visualizar los atributos de la cola modelo:

  ```
  DISPLAY QUEUE (GREEN.MODEL.QUEUE)
  ```

  Use el mandato **ALTER QMODEL** para modificar el modelo a fin de permitir colocaciones en cualquier cola dinámica creada a partir de dicho modelo:

  ```
  ALTER QMODEL (BLUE.MODEL.QUEUE) PUT(ENABLED)
  ```

  Use el mandato **DELETE QMODEL** para borrar esta cola modelo:

  ```
  DELETE QMODEL (RED.MODEL.QUEUE)
  ```

Referencia relacionada
ALTER QMODEL
DEFINE QMODEL
DELETE QMODEL
DISPLAY QUEUE

Trabajar con colas de mensajes no entregados

Normalmente, cada gestor de colas tiene una cola local que utilizará como cola de mensajes no entregados, de modo que los mensajes que no se puedan entregar en su destino correcto se almacenen para su recuperación posterior. Debe informar al gestor de colas sobre la cola de mensajes no entregados y especificar cómo se han de procesar los mensajes encontrados en la cola de mensajes no entregados. La utilización de colas de mensajes no entregados puede afectar la secuencia en que se entregan los mensajes, por lo que puede optar por no utilizarlas.

Antes de empezar

Nota: Los pasos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Acerca de esta tarea

Con el producto se entrega una cola de mensajes no entregados de ejemplo llamada SYSTEM.DEAD.LETTER.QUEUE. Esta cola se crea automáticamente cuando se crea el gestor de colas. Si es necesario, puede modificar esta definición y cambiar su nombre.

Una cola de mensajes no entregados no tiene ningún requisito especial excepto que:

- Debe ser una cola local

- Su atributo MAXMSGL (longitud máxima de mensajes) debe permitir que la cola pueda alojar los mensajes más grandes que el gestor de colas tenga que manejar **más** el tamaño de la cabecera de mensaje no entregado (MQDLH)

La utilización de colas de mensajes no entregados puede afectar la secuencia en que se entregan los mensajes, por lo que puede optar por no utilizarlas.

Procedimiento

- Informe al gestor de colas sobre la cola de mensajes no entregados.

  Para ello, especifique un nombre de cola de mensajes no entregados en el mandato **crtmqm** (`crtmqm -u DEAD.LETTER.QUEUE`, por ejemplo), o utilizando el atributo **DEADQ** en el mandato **ALTER QMGR** para especificar uno más adelante. Debe definir la cola de mensajes no entregados antes de poder utilizarla.
- Especifique cómo se van a procesar los mensajes encontrados en una cola de mensajes no entregados.

  Puede establecer el atributo de canal USEDLQ para determinar si la cola de mensajes no entregados se utiliza cuando los mensajes no se pueden entregar. Este atributo puede configurarse de modo que algunas funciones del gestor de colas utilicen la cola de mensajes no entregados mientras que otras no lo hagan. Para obtener más información sobre el uso del atributo de canal USEDLQ en mandatos MQSC diferentes, consulte DEFINE CHANNEL, DISPLAY CHANNEL, ALTER CHANNELy DISPLAY CLUSQMGR.

  Puede utilizar el manejador de cola de mensajes no entregados de IBM MQ para especificar cómo deben procesarse o eliminarse los mensajes encontrados en una cola de mensajes no entregados. Consulte “Proceso de mensajes en una cola de mensajes no entregados de IBM MQ” en la página 159.

Conceptos relacionados

Colas de mensajes no entregados

Tareas relacionadas

Resolución de problemas de mensajes no entregados

Referencia relacionada

ALTER QMGR

crtmqm (crear gestor de colas)

Proceso de mensajes en una cola de mensajes no entregados de IBM MQ

Para procesar mensajes en una cola de mensajes no entregados (DLQ), utilice el manejador DLQ predeterminado que proporciona IBM MQ. El manejador DLQ compara los mensajes de la DLQ con las entradas de la tabla de reglas que defina.

Acerca de esta tarea

Los gestores de colas, los agentes de canales de mensajes (MCA) y las aplicaciones pueden transferir mensajes a una DLQ. Todos los mensajes de la DLQ deben tener como prefijo una estructura de *cabecera de mensaje no entregado*, MQDLH. Los mensajes transferidos a la DLQ por un gestor de colas o un agente de canales de mensajes tienen siempre esta cabecera; las aplicaciones que transfieran mensajes a la DLQ deben facilitar esta cabecera. El campo *Reason* de la estructura MQDLH contiene un código de razón que indica el motivo por el cual el mensaje está en la DLQ.

Todos los entornos IBM MQ necesitan una rutina para procesar mensajes en la DLQ con regularidad. IBM MQ proporciona una rutina predeterminada, denominada *manejador de cola de mensajes no entregados* (el manejador DLQ), que se invoca utilizando el mandato MQSC **runmqdlq** .

Las instrucciones para procesar mensajes de la DLQ se facilitan al manejador DLQ mediante una *tabla de normas* escrita por el usuario. Es decir, el manejador DLQ compara los mensajes de la DLQ con las entradas de la tabla de reglas; cuando un mensaje DLQ coincide con una de las entradas de la tabla de reglas, el manejador DLQ realiza la acción asociada a dicha entrada.

Tareas relacionadas

Resolución de problemas de mensajes no entregados

Referencia relacionada

Colas de mensajes no entregados

z/OS ALW *Invocación del manejador de cola de mensajes no entregados*

Invoque el manejador de cola de mensajes no entregados (DLQ) utilizando el mandato de control q **runmqdl**. Puede especificar la DLQ que desea procesar y el gestor de colas que desea utilizar de dos formas.

Antes de empezar

Para ejecutar el manejador DLQ, debe tener autorización de acceso a la DLQ propiamente dicha y a todas las colas de mensajes a las que se envíen los mensajes de la DLQ. Además, para que el manejador DLQ pueda transferir mensajes a colas con la autorización del ID de usuario del contexto del mensaje, debe tener autorización para asumir la identidad de otros usuarios.

Acerca de esta tarea

Los ejemplos siguientes se aplican a la DLQ denominada `ABC1.DEAD.LETTER.QUEUE`, propiedad del gestor de colas `ABC1.QUEUE.MANAGER`.

Si no especifica la DLQ o el gestor de colas tal como se indica, se utilizará el gestor de colas predeterminado de la instalación junto con la DLQ del mismo.

El mandato **runmqdlq** toma su entrada de `stdin`. Asocie la tabla de reglas con **runmqdlq** redirigiendo `stdin` desde la tabla de reglas.

Para obtener más información sobre el mandato **runmqdlq**, consulte runmqdlq.

Procedimiento

- Puede nombrar la DLQ y el gestor de colas como parámetros para el mandato **runmqdlq** .

  Por ejemplo, desde el indicador de mandatos:

  ```
  runmqdlq ABC1.DEAD.LETTER.QUEUE ABC1.QUEUE.MANAGER <qrule.rul
  ```

- Puede nombrar la DLQ y el gestor de colas en la tabla de reglas.

  Por ejemplo:

  ```
  INPUTQ(ABC1.DEAD.LETTER.QUEUE) INPUTQM(ABC1.QUEUE.MANAGER)
  ```

Conceptos relacionados

Colas de mensajes no entregados

Tareas relacionadas

Resolución de problemas de mensajes no entregados

z/OS ALW *El manejador DLQ de ejemplo **amqsdlq***

Además del manejador de la cola de mensajes no entregados invocado mediante el mandato **runmqdlq**, IBM MQ proporciona el código fuente de un manejador DLQ de ejemplo, **amqsdlq**, con una función que es similar a la que proporciona **runmqdlq**.

Puede personalizar **amqsdlq** para proporcionar un manejador DLQ que cumpla sus requisitos. Por ejemplo, puede que necesite un manejador DLQ que pueda procesar mensajes sin cabeceras de mensajes no entregados. (Tanto el manejador DLQ predeterminado como el ejemplo, **amqsdlq**, sólo procesan los

mensajes en la DLQ que empiezan por una cabecera de mensaje no entregado, MQDLH. Los mensajes que no empiezan por una MQDLH se identifican como erróneos y permanecen indefinidamente en la DLQ).

MQ_INSTALLATION_PATH representa el directorio de alto nivel en el que está instalado IBM MQ.

En IBM MQ for Windows, el código fuente de amqsdlq se encuentra en el directorio:

```
MQ_INSTALLATION_PATH\tools\c\samples\dlq
```

y la versión compilada se encuentra en el directorio:

```
MQ_INSTALLATION_PATH\tools\c\samples\bin
```

En los sistemas IBM MQ for UNIX y Linux , el origen de **amqsdlq** se suministra en el directorio:

MQ_INSTALLATION_PATH/samp/dlq

y la versión compilada se encuentra en el directorio:

MQ_INSTALLATION_PATH/samp/bin

Se incluye una versión incorporada del programa de ejemplo, denominado **amqsdlqc**. Puede utilizar esta opción para conectarse a un gestor de colas remoto en modalidad de cliente. Para utilizar **amqsdlqc** debe establecer una de las variables de entorno *MQSERVER* , **MQCHLLIB** o **MQCHLTAB** para identificar cómo conectarse al gestor de colas. Por ejemplo:

```
export MQSERVER="SYSTEM.DEF.SVRCONN/TCP/myappliance.co.uk(1414)"
```

z/OS ALW *La tabla de reglas del manejador DLQ*

La tabla de reglas del manejador de la cola de mensajes no entregados define cómo debe el manejador DLQ procesar los mensajes que llegan a la DLQ.

Una tabla de reglas tiene dos tipos de entradas:

- La primera entrada de la tabla, que es opcional, contiene *datos de control*.
- Todas las demás entradas de la tabla son *reglas* que el manejador DLQ debe seguir. Cada regla está formada por un *patrón* (un conjunto de características de mensaje) con el que se compara el mensaje, y por una *acción* que debe realizarse cuando un mensaje de la DLQ coincide con el patrón especificado. En una tabla de reglas debe haber una regla como mínimo.

Cada entrada de la tabla de reglas tiene una o más palabras clave.

Conceptos relacionados

Colas de mensajes no entregados

Tareas relacionadas

Resolución de problemas de mensajes no entregados

z/OS ALW *Datos de control de DLQ*

Puede incluir palabras clave en una entrada de datos de control de una tabla de reglas del manejador de la cola de mensajes no entregados.

Nota:

- La línea vertical (|) separa alternativas, de las que sólo puede especificarse una.
- Todas las palabras clave son opcionales.

INPUTQ (*QueueName*| '' (valor predeterminado))

El nombre de la DLQ que desea procesar:

1. Cualquier valor INPUTQ que especifique como parámetro en el mandato `runmqdlq`, prevalece sobre todos los valores INPUTQ de la tabla de reglas.

2. Si no especifica ningún valor INPUTQ como parámetro en el mandato `runmqdlq`, pero **sí** especifica un valor en la tabla de reglas, se utiliza el valor INPUTQ de la tabla de reglas.
3. Si no se especifica ninguna DLQ o si se especifica INPUTQ(' ') en la tabla de reglas, se utiliza el nombre de la DLQ que pertenece al gestor de colas cuyo nombre se proporciona como parámetro en el mandato `runmqdlq`.
4. Si no especifica ningún valor INPUTQ como parámetro en el mandato `runmqdlq` o como valor en la tabla de reglas, se utiliza la DLQ que pertenece al gestor de colas nombrado en la palabra clave INPUTQM de la tabla de reglas.

INPUTQM (*QueueManagerNombre*| '' (valor predeterminado))
El nombre del gestor de colas al que pertenece la DLQ nombrada en la palabra clave INPUTQ:

1. Cualquier valor INPUTQM que especifique como parámetro en el mandato `runmqdlq`, prevalece sobre todos los valores INPUTQM de la tabla de reglas.
2. Si no especifica ningún valor INPUTQM como parámetro en el mandato `runmqdlq`, se utiliza el valor INPUTQM de la tabla de reglas.
3. Si no especifica ningún gestor de colas o especifica INPUTQM(' ') en la tabla de reglas, se utiliza el gestor de colas predeterminado de la instalación.

RETRYINT (*Intervalo*| 60 (valor predeterminado))
Es el intervalo, en segundos, tras el cual el manejador DLQ debe intentar volver a procesar mensajes de la DLQ que no pudo procesar en el primer intento, y para los cuales se han solicitado varios intentos. De forma predeterminada, el intervalo de reintento es de 60 segundos.

WAIT (YES (valor predeterminado) |NO|*nnn*)
Indica si el manejador DLQ debe esperar a que lleguen más mensajes a la DLQ cuando detecta que no quedan más mensajes que pueda procesar.

SÍ
Hace que el manejador DLQ espere indefinidamente.

NO
Hace que el manejador DLQ termine cuando detecta que la DLQ está vacía o no contiene mensajes que pueda procesar.

nnn
Hace que el manejador DLQ, después de haber detectado que la cola está vacía o no contiene mensajes que pueda procesar, espere *nnn* segundos a que llegue un nuevo trabajo antes de terminar.

Especifique WAIT (YES) en DLQ ocupadas y WAIT (NO) o WAIT (*nnn*) para las DLQ que tengan un nivel de actividad bajo. Si el manejador DLQ tiene autorización para terminar, es aconsejable invocarlo de nuevo mediante la activación. Para obtener más información sobre el desencadenamiento, consulte Inicio de aplicaciones IBM MQ utilizando desencadenantes.

Como alternativa a la inclusión de datos de control en la tabla de reglas, puede facilitar los nombres de la DLQ y su gestor de colas como parámetros de entrada del mandato `runmqdlq`. Si se especifican valores tanto en la tabla de reglas como en la entrada del mandato `runmqdlq`, prevalece el valor especificado en el mandato `runmqdlq`.

Si incluye una entrada de datos de control en la tabla de reglas, dicha entrada ser la **primera** de la tabla.

Reglas de la cola de mensajes no entregados (patrones y acciones)
Una descripción de las palabras clave de coincidencia de patrón (aquellas con las que se comparan los mensajes de la cola de mensajes no entregados) y las palabras clave de acción (aquellas que determinan la forma en que el manejador de la cola de mensajes no entregados debe procesar un mensaje coincidente). También se proporciona una regla de ejemplo.

Palabras clave de coincidencia de patrón

Las palabras clave de coincidencia de patrón, que se utilizan para especificar valores con los que se comparan los mensajes de la DLQ, son las siguientes. (Todas las palabras clave de coincidencia de patrón son opcionales):

APPLIDAT (*ApplIdentityDatos*|* (valor predeterminado))
El valor de *ApplIdentityData* indicado en el descriptor del mensaje, MQMD, del mensaje de la DLQ.

APPLNAME (*PutApplNombre*|* (valor predeterminado))
El nombre de la aplicación que ha emitido las llamadas MQPUT o MQPUT1, tal y como se especifica en el campo *NombAplColoc* del descriptor de mensaje, MQMD, del mensaje en la DLQ.

APPLTYPE (*PutApplTipo*|* (valor predeterminado))
El valor *TipoAplColoc*, especificado en el descriptor de mensaje, MQMD, del mensaje que está en la DLQ.

DESTQ (*QueueName*|* (valor predeterminado))
El nombre de la cola de mensajes a la que está destinado el mensaje.

DESTQM (*QueueManagerNombre*|* (valor predeterminado))
El nombre del gestor de colas de la cola de mensajes a la que está destinado el mensaje.

FEEDBACK (*Comentarios*|* (valor predeterminado))
Cuando el valor de *TipoMsj* es MQFB_REPORT, *Respuesta* describe la naturaleza del informe.

Puede utilizar nombres simbólicos. Por ejemplo, puede utilizar el nombre simbólico MQFB_COA para identificar los mensajes de la DLQ que requieren confirmación de la llegada a sus respectivas colas de destino.

FORMAT (*Format*|* (valor predeterminado))
El nombre que el remitente del mensaje utiliza para describir el formato de los datos del mensaje.

MSGTYPE (*MsgType*|* (valor predeterminado))
El tipo de mensaje del mensaje de la DLQ.

Puede utilizar nombres simbólicos. Por ejemplo, puede utilizar el nombre simbólico MQMT_REQUEST para identificar los mensajes de la DLQ que requieren respuesta.

PERSIST (*Persistencia*|* (valor predeterminado))
El valor de persistencia del mensaje. (La persistencia de un mensaje determina si sigue existiendo después de reiniciar el gestor de colas).

Puede utilizar nombres simbólicos. Por ejemplo, puede utilizar el nombre simbólico MQPER_PERSISTENT para identificar mensajes de la DLQ que son persistentes.

REASON (*ReasonCode*|* (valor predeterminado))
El código de razón que explica por qué se ha colocado el mensaje en la DLQ.

Puede utilizar nombres simbólicos. Por ejemplo, puede utilizar el nombre simbólico MQRC_Q_FULL para identificar los mensajes que se colocaron en la DLQ debido a que las colas de destino correspondientes estaban llenas.

REPLYQ (*QueueName*|* (valor predeterminado))
El nombre de la cola de respuesta especificado en el descriptor del mensaje, MQMD, del mensaje de la DLQ.

REPLYQM (*QueueManagerNombre*|* (valor predeterminado))
El nombre del gestor de colas de la cola de respuesta especificado en el descriptor de mensaje, MQMD, del mensaje de la DLQ.

USERID (*UserIdentifier*|* (valor predeterminado))
El ID del usuario que originó el mensaje que está en la DLQ, tal como se especifica en el descriptor de mensaje, MQMD, del mensaje de la DLQ.

Las palabras clave de acción

Las palabras clave de acción, que se utilizan para describir la forma de procesar un mensaje, son las siguientes:

ACTION (DISCARD|IGNORE|RETRY|FWD)
Es la acción que debe realizarse para cualquier mensaje de la DLQ que coincida con el patrón definido en esta regla.

DISCARD
Hace que el mensaje se suprima de la DLQ.

IGNORE
Hace que el mensaje permanezca en la DLQ.

RETRY
Si el primer intento de transferir el mensaje a la cola de destino no se ejecuta correcta, hace que se vuelva a intentar. La palabra clave RETRY establece el número de intentos realizados para implementar una acción. La palabra clave RETRYINT de los datos de control controla el intervalo entre intentos.

FWD
Hace que se remita el mensaje a la cola mencionada en la palabra clave FWDQ.

Debe especificar la palabra clave ACTION.

FWDQ (*QueueName*|&DESTQ|&REPLYQ)
El nombre de la cola de mensajes a la que debe remitirse el mensaje cuando se solicita la acción ACTION (FWD).

QueueName
El nombre de una cola de mensajes. FWDQ(' ') no es válido.

&DESTQ
Hace que el nombre de la cola se tome del campo *DestQName* de la estructura MQDLH.

&REPLYQ
Toma el nombre de cola del campo *ReplyToQ* del descriptor MQMD del mensaje.

Para evitar mensajes de error cuando una regla que especifica FWDQ (& REPLYQ) coincide con un mensaje con un campo *ReplyToQ* en blanco, especifique REPLYQ (? *) en el patrón del mensaje.

FWDQM (*QueueManagerNombre*| & DESTQM | & REPLYQM | '' (valor predeterminado))
El gestor de colas de la cola a la que debe remitirse un mensaje.

QueueManagerName
El nombre del gestor de colas de la cola a la que debe remitirse un mensaje cuando se solicita la acción ACTION (FWD).

&DESTQM
Toma el nombre del gestor de colas del campo *DestQMgrName* de la estructura MQDLH.

&REPLYQM
Toma el nombre del gestor de colas del campo *ReplyToQMgr* del descriptor MQMD del mensaje.

' '
FWDQM(' '), que es el valor predeterminado, identifica el gestor de colas local.

HEADER (YES (valor predeterminado) |NO)
Indica si la MQDLH debe permanecer en un mensaje para el que se solicita la acción ACTION (FWD). Por omisión, la MQDLH permanece en el mensaje. La palabra clave HEADER no es válida para acciones distintas de FWD.

PUTAUT (DEF (valor predeterminado) | CTX)
La autorización con la que el manejador DLQ debe transferir los mensajes:

DEF
Hace que los mensajes se transfieran con la autorización del propio manejador DLQ.

CTX
Hace que los mensajes se transfieran con la autorización del ID de usuario del contexto del mensaje. Si especifica PUTAUT (CTX), deberá estar autorizado para asumir la identidad de otros usuarios.

RETRY (*RecuentoReintentos*|1 (valor predeterminado))
El número de veces, dentro del rango de 1 a 999.999.999, para intentar una acción (en el intervalo especificado en la palabra clave RETRYINT de los datos de control). El número de intentos realizados por el manejador DLQ para implementar una regla concreta es específico de la instancia actual del manejador DLQ; el número total no se conserva al reiniciar. Si el manejador DLQ se reinicia, la cuenta de intentos efectuados para llevar a cabo una regla se restaura a cero.

Norma de ejemplo

La siguiente es una regla de ejemplo de una tabla de reglas del manejador DLQ:

```
PERSIST(MQPER_PERSISTENT) REASON (MQRC_PUT_INHIBITED) +
ACTION (RETRY) RETRY (3)
```

Esta regla indica al manejador DLQ que intente tres veces entregar en la cola de destino cualquier mensaje persistente que se haya transferido a la DLQ debido a que MQPUT y MQPUT1 estaban inhibidos.

Todas las palabras clave que pueden utilizarse en una regla están descritas en el resto de este apartado. Tenga en cuenta lo siguiente:

- El valor predeterminado de una palabra clave, si lo hay, está subrayado. Para la mayoría de palabras clave, el valor predeterminado es * (asterisco), que coincide con cualquier valor.
- La línea vertical (|) separa alternativas, de las que sólo puede especificarse una.
- Todas las palabras clave excepto ACTION son opcionales.

z/OS ALW *Convenios de la tabla de reglas de la cola de mensajes no entregados*

La sintaxis, estructura y contenido de la tabla de reglas del manejador de la cola de mensajes no entregados debe cumplir estos convenios.

La tabla de reglas debe seguir los siguientes convenios:

- Una tabla de reglas debe contener una regla como mínimo.
- Las palabras clave pueden estar en cualquier orden.
- Una palabra clave sólo puede incluirse una vez en cualquier regla.
- Las palabras clave no son sensibles a las mayúsculas y minúsculas.
- Una palabra clave y el valor de su parámetro deben estar separados de las demás palabras clave por un blanco o una coma como mínimo.
- Al principio o al final de una regla y entre palabras clave, signos de puntuación y valores, puede haber cualquier número de espacios en blanco.
- Cada regla debe empezar en una nueva línea.
- En sistemas Windows, la última regla de la tabla debe finalizar con un carácter de retorno de carro/salto de línea. Esto se puede conseguir pulsando la tecla Intro al final de la regla, para que la última línea de la tabla sea una línea en blanco.
- Por motivos de portabilidad, la longitud significativa de una línea no debe ser superior a 72 caracteres.
- Utilice el signo más (+) en una línea como último carácter distinto del blanco para indicar que la regla continúa en el primer carácter distinto del blanco de la línea siguiente. Utilice el signo menos (-) en una línea como último carácter distinto del blanco para indicar que la regla continúa en el principio de la línea siguiente. Puede haber caracteres de continuación dentro de palabras clave y parámetros.

Por ejemplo:

```
APPLNAME('ABC+
D')
```

tiene como resultado 'ABCD' y

```
APPLNAME('ABC-
D')
```

da como resultado 'ABC D'.

- Puede haber líneas de comentario, que empiezan por un asterisco (*), en cualquier lugar de la tabla de reglas.
- Las líneas en blanco se ignoran.
- Cada entrada de la tabla de reglas del manejador DLQ incluye una o más palabras clave y sus parámetros asociados. Los parámetros deben seguir estas reglas sintácticas:
  - Cada valor de parámetro debe contener, como mínimo, un carácter significativo. Las comillas simples delimitadoras utilizadas en los valores que se encierran entre comillas no se consideran significativas. Por ejemplo, serían válidos estos parámetros:

`FORMAT('ABC')`	3 caracteres significativos
`FORMAT(ABC)`	3 caracteres significativos
`FORMAT('A')`	1 carácter significativo
`FORMAT(A)`	1 carácter significativo
`FORMAT(' ')`	1 carácter significativo

    Estos parámetros no son válidos porque no contienen caracteres significativos:

    ```
    FORMAT('')
    FORMAT( )
    FORMAT()
    FORMAT
    ```

  - Se admiten caracteres comodín. Puede utilizar el signo de interrogación (?) en lugar de cualquier carácter individual, excepto un blanco final; puede utilizar el asterisco (*) en lugar de cero o más caracteres adyacentes. El asterisco (*) y el signo de interrogación (?) se interpretan ***siempre*** como caracteres comodín en los valores de parámetros.
  - No pueden incluirse caracteres comodín en los parámetros de las siguientes palabras clave: ACTION, HEADER, RETRY, FWDQ, FWDQM y PUTAUT.
  - Los espacios en blanco de cola de los valores de parámetros y de los campos correspondientes en los mensajes de la DLQ no son significativos para realizar comparaciones con comodines. No obstante, los espacios en blanco de cabecera e intercalados en series de caracteres que se encierran entre comillas sí son significativos en las comparaciones con comodines.
  - Los parámetros numéricos no pueden incluir el carácter comodín de signo de interrogación (?). Puede utilizar el asterisco (*) en lugar de un parámetro numérico completo, pero no como parte de un parámetro numérico. Por ejemplo, serían válidos estos parámetro numéricos:

`MSGTYPE(2)`	Sólo pueden elegirse mensajes de respuesta
`MSGTYPE(*)`	Puede elegirse cualquier tipo de mensaje
`MSGTYPE('*')`	Puede elegirse cualquier tipo de mensaje

Sin embargo, MSGTYPE('2*') no es válido, porque incluye un asterisco (*) como parte de un parámetro numérico.

- Los parámetros numéricos deben estar comprendidos entre 0 y 999.999.999. Si el valor del parámetro está dentro de este rango, se aceptará, incluso si en ese momento no es válido en el campo asociado a la palabra clave. Puede utilizar nombres simbólicos para parámetros numéricos.
- Si un valor de tipo serie es más corto que el campo de MQDLH o MQMD con el que está relacionada la palabra clave, el valor se rellenará con espacios en blanco hasta alcanzar la longitud del campo. Si el valor, excluyendo los asteriscos, es más largo que la longitud del campo, se diagnosticará un error. Por ejemplo, todos estos valores de tipo serie son válidos para un campo de 8 caracteres:

'ABCDEFGH'	8 caracteres
'A*C*E*G*I'	5 caracteres, excluidos los asteriscos
'*A*C*E*G*I*K*M*O*'	8 caracteres, excluidos los asteriscos

- Escriba series que contengan espacios en blanco, caracteres en minúsculas o caracteres especiales que no sean el punto (.), la barra inclinada (?), el subrayado (_) y el signo de porcentaje (%) entre comillas simples. Los caracteres en minúsculas que no estén entre comillas simples se convierten a mayúsculas. Si la serie incluye una parte entrecomillada, deberán utilizarse dos comillas simples para indicar el principio y el final de la parte entrecomillada. Cuando se calcula la longitud de la serie, cada aparición de comillas dobles se cuenta como un solo carácter.

z/OS ALW *Cómo se procesa la tabla de reglas de la cola de mensajes no entregados*

El manejador de la cola de mensajes no entregados busca en la tabla de reglas una regla donde el patrón coincida con un mensaje de la DLQ.

La búsqueda empieza por la primera regla de la tabla y continúa secuencialmente por la tabla. Cuando el manejador DLQ encuentra una regla con un patrón que coincide, intenta realizar la acción que indica la regla. El manejador DLQ aumenta en 1 la cuenta de intentos de la regla cada vez que intenta aplicarla. Si el primer intento falla, el manejador DLQ vuelve a intentarlo hasta que la cuenta de intentos es igual al número especificado en la palabra clave RETRY. Si todos los intentos fallan, el manejador DLQ busca la siguiente regla coincidente de la tabla.

Este proceso se repite para las siguientes reglas coincidentes hasta que se realiza una acción satisfactoriamente. Cuando se han intentado realizar todas las reglas coincidentes el número de veces especificado en su palabra clave RETRY y todos los intentos han fallado, se presupone ACTION (IGNORE). ACTION (IGNORE) también se presupone si no se encuentra ninguna regla coincidente.

Nota:

1. Sólo se buscan patrones de reglas coincidentes para mensajes de la DLQ que empiecen por MQDLH. Los mensajes que no empiezan por MQDLH se indican periódicamente como erróneos, y permanecen en la DLQ indefinidamente.
2. Todas las palabras clave de patrón pueden adoptar su valor predeterminado, por lo que una regla puede constar únicamente de una acción. No obstante, tenga en cuenta que las reglas que son sólo una acción se aplican a todos los mensajes de la cola que tienen MQDLH y que no se han procesado todavía de acuerdo con otras reglas de la tabla.
3. La tabla de reglas se valida cuando se inicia el manejador DLQ, y los errores se indican en ese momento. Puede efectuar cambios en la tabla de reglas en cualquier momento, pero esos cambios no serán efectivos hasta que se reinicie el manejador DLQ.
4. El manejador DLQ no altera el contenido de los mensajes, de la MQDLH ni del descriptor de mensaje. El manejador DLQ siempre pone mensajes en otras colas con la opción de mensaje MQPMO_PASS_ALL_CONTEXT.
5. Es posible que los errores de sintaxis consecutivos en la tabla de reglas no se reconozcan debido a que la tabla de reglas se ha diseñado para eliminar la generación de errores repetitivos durante la validación.
6. El manejador DLQ abre la DLQ con la opción MQOO_INPUT_AS_Q_DEF.

7. Pueden ejecutarse simultáneamente varias instancias del manejador DLQ para la misma cola, utilizando la misma tabla de reglas. Pero es más normal que exista una relación unívoca entre una DLQ y un manejador DLQ.

Conceptos relacionados

Colas de mensajes no entregados

Tareas relacionadas

Resolución de problemas de mensajes no entregados

z/OS ALW *Ejemplo de tabla de reglas del manejador DLQ*

Un ejemplo de tabla de reglas de la cola de mensajes no entregados para el mandato **runmqdlq**, que contiene una sola entrada de control de datos y varias reglas.

```
*********************************************************************
*    An example rules table for the runmqdlq command        *
*********************************************************************
* Control data entry
* -----------------
* If no queue manager name is supplied as an explicit parameter to
* runmqdlq, use the default queue manager for the machine.
* If no queue name is supplied as an explicit parameter to runmqdlq,
* use the DLQ defined for the local queue manager.
*
inputqm(' ') inputq(' ')

* Rules
* -----
* We include rules with ACTION (RETRY) first to try to
* deliver the message to the intended destination.
* If a message is placed on the DLQ because its destination
* queue is full, attempt to forward the message to its
* destination queue. Make 5 attempts at approximately
* 60-second intervals (the default value for RETRYINT).

REASON(MQRC_Q_FULL) ACTION(RETRY) RETRY(5)

* If a message is placed on the DLQ because of a put inhibited
* condition, attempt to forward the message to its
* destination queue. Make 5 attempts at approximately
* 60-second intervals (the default value for RETRYINT).

REASON(MQRC_PUT_INHIBITED) ACTION(RETRY) RETRY(5)

* The AAAA corporation are always sending messages with incorrect
* addresses. When we find a request from the AAAA corporation,
* we return it to the DLQ (DEADQ) of the reply-to queue manager
* (&REPLYQM).
* The AAAA DLQ handler attempts to redirect the message.

MSGTYPE(MQMT_REQUEST) REPLYQM(AAAA.*) +
ACTION(FWD) FWDQ(DEADQ) FWDQM(&REPLYQM)

* The BBBB corporation never do things by half measures. If
* the queue manager BBBB.1 is unavailable, try to
* send the message to BBBB.2

DESTQM(bbbb.1) +
action(fwd) fwdq(&DESTQ) fwdqm(bbbb.2) header(no)

* The CCCC corporation considers itself very security
* conscious, and believes that none of its messages
* will ever end up on one of our DLQs.
* Whenever we see a message from a CCCC queue manager on our
* DLQ, we send it to a special destination in the CCCC organization
* where the problem is investigated.

REPLYQM(CCCC.*) +
ACTION(FWD) FWDQ(ALARM) FWDQM(CCCC.SYSTEM)
```

```
* Messages that are not persistent run the risk of being
* lost when a queue manager terminates. If an application
* is sending nonpersistent messages, it should be able
* to cope with the message being lost, so we can afford to
```

```
* discard the message.  PERSIST(MQPER_NOT_PERSISTENT) ACTION(DISCARD)
* For performance and efficiency reasons, we like to keep
* the number of messages on the DLQ small.
* If we receive a message that has not been processed by
* an earlier rule in the table, we assume that it
* requires manual intervention to resolve the problem.
* Some problems are best solved at the node where the
* problem was detected, and others are best solved where
* the message originated. We don't have the message origin,
* but we can use the REPLYQM to identify a node that has
* some interest in this message.
* Attempt to put the message onto a manual intervention
* queue at the appropriate node. If this fails,
* put the message on the manual intervention queue at
* this node.

REPLYQM('?*') +
ACTION(FWD) FWDQ(DEADQ.MANUAL.INTERVENTION) FWDQM(&REPLYQM)

ACTION(FWD) FWDQ(DEADQ.MANUAL.INTERVENTION)
```

Conceptos relacionados

Colas de mensajes no entregados

Tareas relacionadas

Resolución de problemas de mensajes no entregados

Referencia relacionada

runmqdlq (ejecutar manejador de cola de mensajes no entregados)

IBM i *Invocación del manejador de cola de mensajes no entregados en IBM i*

En IBM MQ for IBM i, invoque el manejador DLQ estableciendo el mandato **STRMQMDLQ** .

Antes de empezar

Para poder ejecutar el manejador DLQ debe estar autorizado para acceder a la DLQ y a las colas de mensajes a las que se reenvían los mensajes de la DLQ. También debe estar autorizado para asumir la identidad de otros usuarios, para que la DLQ pueda colocar mensajes en colas con la autorización del ID de usuario del contexto del mensaje.

Nota: A menudo es preferible evitar poner mensajes en una DLQ. Para obtener información sobre cuándo utilizar las DLQ y cuándo evitarlas, consulte “Trabajar con colas de mensajes no entregados” en la página 158.

Acerca de esta tarea

Una *cola de mensajes no entregados* (DLQ), denominada a veces *cola de mensajes sin entrega* es una cola que retiene los mensajes que no se pueden entregar a las colas de destino. Cada gestor de colas de una red debe tener una DLQ asociada a él.

Los gestores de colas, los agentes de canal de mensajes y las aplicaciones pueden poner mensajes en la DLQ. Todos los mensajes de la DLQ deben tener como prefijo una estructura de *cabecera de mensaje no entregado*, MQDLH. Los mensajes transferidos a la DLQ por un gestor de colas o por un agente de canal de mensajes tienen siempre una MQDLH. Para las aplicaciones que colocan mensajes en la DLQ, debe proporcionar una MQDLH.

El campo *Reason* de la estructura MQDLH contiene un código de razón que indica el motivo por el cual el mensaje está en la DLQ.

En todos los entornos de IBM MQ, debe haber una rutina que se ejecute periódicamente para procesar los mensajes de la DLQ. IBM MQ proporciona una rutina predeterminada, denominada *manejador de cola de mensajes no entregados* (el manejador DLQ), que se invoca utilizando el mandato **STRMQMDLQ** . Una *tabla de reglas* escrita por el usuario proporciona instrucciones al manejador DLQ para procesar los mensajes que hay en la DLQ. Es decir, el manejador DLQ compara los mensajes de la DLQ con las entradas de la tabla de reglas. Cuando un mensaje DLQ coincide con una entrada de la tabla de reglas, el manejador DLQ realiza la acción asociada a dicha entrada.

Procedimiento

- Invocar el manejador DLQ

  Utilice el mandato **STRMQMDLQ** para invocar el manejador DLQ. El nombre de la DLQ que desea procesar y del gestor de colas que va a utilizar puede indicarse de dos formas:

  – Como parámetros para **STRMQMDLQ** desde el indicador de mandatos. Por ejemplo:

  ```
  STRMQMDLQ UDLMSGQ(ABC1.DEAD.LETTER.QUEUE) SRCMBR(QRULE) SRCFILE(library/QTXTSRC)
  MQMNAME(MY.QUEUE.MANAGER)
  ```

  – En la tabla de reglas. Por ejemplo:

  ```
  INPUTQ(ABC1.DEAD.LETTER.QUEUE)
  ```

  Nota: La tabla de reglas es un miembro dentro de un archivo físico fuente que puede adoptar cualquier nombre.

  Los ejemplos se aplican a la DLQ denominada ABC1.DEAD.LETTER.QUEUE, propiedad del gestor de colas predeterminado.

  Si no especifica la DLQ o el gestor de colas tal como se indica, se utilizará el gestor de colas predeterminado de la instalación junto con la DLQ del mismo.

Conceptos relacionados

Colas de mensajes no entregados

Tareas relacionadas

Resolución de problemas de mensajes no entregados

IBM i *La tabla de reglas del manejador DLQ en IBM i*

La tabla de reglas del manejador de la cola de mensajes no entregados define cómo el manejador DLQ procesa los mensajes que llegan a la cola de mensajes no entregados de IBM i.

La tabla de reglas del manejador DLQ define de qué forma debe el manejador DLQ procesar los mensajes que llegan a la DLQ. Una tabla de reglas tiene dos tipos de entradas:

- La primera entrada de la tabla, que es opcional, contiene *datos de control*.
- Todas las demás entradas de la tabla son *reglas* que el manejador DLQ debe seguir. Cada regla está formada por un *patrón* (un conjunto de características de mensaje) con el que se compara el mensaje, y por una *acción* que debe realizarse cuando un mensaje de la DLQ coincide con el patrón especificado. En una tabla de reglas debe haber una regla como mínimo.

Cada entrada de la tabla de reglas tiene una o más palabras clave.

Datos de control

Este apartado describe las palabras clave que pueden incluirse en una entrada de datos de control de una tabla de reglas del manejador DLQ. Tenga en cuenta lo siguiente:

- El valor predeterminado de una palabra clave, si lo hay, está subrayado.
- La línea vertical (|) separa alternativas. Sólo puede especificar una de ellas.
- Todas las palabras clave son opcionales.

INPUTQ (*QueueName*| '' (valor predeterminado))

El nombre de la DLQ que desea procesar:

1. Cualquier valor UDLMSGQ (o *dft) que especifique como parámetro en el mandato **STRMQMDLQ** altera temporalmente cualquier valor INPUTQ en la tabla de reglas.
2. Si especifica un valor UDLMSGQ en blanco como parámetro del mandato **STRMQMDLQ**, se utiliza el valor INPUTQ de la tabla de reglas.

3. Si especifica un valor UDLMSGQ en blanco como parámetro del mandato **STRMQMDLQ** y un valor INPUTQ en blanco en la tabla de reglas, se utiliza la cola de mensajes no entregados predeterminada del sistema.

INPUTQM (*QueueManagerNombre*| '' (valor predeterminado))
El nombre del gestor de colas que es propietario de la DLQ nombrada en la palabra clave INPUTQ.

Si no especifica ningún gestor de colas o si especifica INPUTQM(' ') en la tabla de reglas, el sistema utiliza el gestor de colas predeterminado de la instalación.

RETRYINT (*Intervalo*| 60 (valor predeterminado))
El intervalo, en segundos, tras el que el manejador DLQ debe intentar volver a procesar mensajes de la DLQ que no se pudieron procesar en el primer intento y de los cuales se han solicitado varios intentos. De forma predeterminada, el intervalo de reintento es de 60 segundos.

WAIT (YES (valor predeterminado) |NO|*nnn*)
Indica si el manejador DLQ debe esperar a que lleguen más mensajes a la DLQ cuando detecta que no quedan más mensajes que pueda procesar.

SÍ
Hace que el manejador DLQ espere indefinidamente

NO
Hace que el manejador DLQ termine en cuanto detecta que la DLQ está vacía o no contiene mensajes que pueda procesar.

nnn
Hace que el manejador DLQ, después de haber detectado que la cola está vacía o no contiene mensajes que pueda procesar, espere *nnn* segundos a que llegue un nuevo trabajo antes de terminar.

Especifique WAIT (YES) en DLQ ocupadas y WAIT (NO) o WAIT (*nnn*) para las DLQ que tengan un nivel de actividad bajo. Si el manejador DLQ tiene autorización para terminar, invóquelo de nuevo utilizando el mecanismo de desencadenamiento.

Como alternativa a incluir datos de control en la tabla de reglas, puede suministrar el nombre de la DLQ como parámetro de entrada del mandato **STRMQMDLQ**. Si hay un valor que esté especificado tanto en la tabla de reglas como en la entrada del mandato **STRMQMDLQ**, el valor que prevalece es el especificado en el mandato **STRMQMDLQ**.

Nota: Si se incluye una entrada de datos de control en la tabla de reglas, debe ser la primera entrada de la tabla.

IBM i *Reglas de la cola de mensajes no entregados (patrones y acciones) en IBM i*
Una descripción de los patrones y acciones para cada una de las reglas de cola de mensajes no entregados de IBM i.

La siguiente es una regla de ejemplo de una tabla de reglas del manejador DLQ:

```
PERSIST(MQPER_PERSISTENT) REASON (MQRC_PUT_INHIBITED) +
ACTION (RETRY) RETRY (3)
```

Esta regla indica al manejador DLQ que haga 3 intentos de entregar en la cola destino cualquier mensaje persistente que se haya puesto en la DLQ debido a que MQPUT y MQPUT1 estaban inhibidos.

Este apartado describe las palabras clave que pueden incluirse en una regla. Tenga en cuenta lo siguiente:

- El valor predeterminado de una palabra clave, si lo hay, está subrayado. Para la mayoría de palabras clave, el valor predeterminado es * (asterisco), que coincide con cualquier valor.
- La línea vertical (|) separa alternativas. Sólo puede especificar una de ellas.
- Todas las palabras clave excepto ACTION son opcionales.

Este apartado empieza con una descripción de las palabras clave de coincidencia de patrón (aquellas con las que se comparan los mensajes de la DLQ). A continuación, se describen las palabras clave de acción (aquellas que determinan de qué forma va a procesar el manejador DLQ el mensaje que coincide).

IBM i *Palabras clave de coincidencia de patrón de DLQ en IBM i*

Las palabras clave de coincidencia de patrón se describen en un ejemplo. Utilice estas palabras clave para especificar valores con los que se comparan los mensajes en la cola de mensajes no entregados de IBM i. Todas las palabras clave de coincidencia con un patrón son opcionales.

APPLIDAT (*ApplIdentityDatos*|* (valor predeterminado))
El valor *DatosIdentidadAplicación* del mensaje que está en la DLQ especificada en el descriptor de mensaje, MQMD.

APPLNAME (*PutApplNombre*|* (valor predeterminado))
El nombre de la aplicación que emitió la llamada MQPUT o MQPUT1, según se indica en el campo *NombAplColoc* del descriptor de mensaje, MQMD, del mensaje que está en la DLQ.

APPLTYPE (*PutApplTipo*|* (valor predeterminado))
El valor *TipoAplColoc* especificado en el descriptor de mensaje, MQMD, del mensaje que está en la DLQ.

DESTQ (*QueueName*|* (valor predeterminado))
El nombre de la cola de mensajes a la que está destinado el mensaje.

DESTQM (*QueueManagerNombre*|* (valor predeterminado))
El nombre del gestor de colas de la cola de mensajes a la que está destinado el mensaje.

FEEDBACK (*Comentarios*|* (valor predeterminado))
Cuando el valor de *TipoMensaje* es MQMT_REPORT, *InformaciónRetorno* describe la naturaleza del informe.

Puede utilizar nombres simbólicos. Por ejemplo, puede utilizar el nombre simbólico MQFB_COA para identificar los mensajes de la DLQ que requieren confirmación de que han llegado a sus respectivas colas de destino.

FORMAT (*Format*|* (valor predeterminado))
El nombre que el remitente del mensaje utiliza para describir el formato de los datos del mensaje.

MSGTYPE (*MsgType*|* (valor predeterminado))
El tipo de mensaje del mensaje de la DLQ.

Puede utilizar nombres simbólicos. Por ejemplo, puede utilizar el nombre simbólico MQMT_REQUEST para identificar los mensajes de la DLQ que requieren respuesta.

PERSIST (*Persistencia*|* (valor predeterminado))
El valor de persistencia del mensaje. (La persistencia de un mensaje determina si sigue existiendo después de reiniciar el gestor de colas).

Puede utilizar nombres simbólicos. Por ejemplo, puede utilizar el nombre simbólico MQPER_PERSISTENT para identificar los mensajes de la DLQ que son persistentes.

REASON (*ReasonCode*|* (valor predeterminado))
El código de razón que explica por qué se ha colocado el mensaje en la DLQ.

Puede utilizar nombres simbólicos. Por ejemplo, puede utilizar el nombre simbólico MQRC_Q_FULL para identificar los mensajes que se colocaron en la DLQ debido a que las colas de destino correspondientes estaban llenas.

REPLYQ (*QueueName*|* (valor predeterminado))
El nombre de la cola de respuesta especificado en el descriptor de mensaje, MQMD, del mensaje que está en la DLQ.

REPLYQM (*QueueManagerNombre*|* (valor predeterminado))
El nombre del gestor de colas de la cola de respuesta especificada en la palabra clave REPLYQ.

USERID (*UserIdentifier*|* (valor predeterminado))
El ID del usuario que originó el mensaje que está en la DLQ, tal como se especifica en el descriptor de mensaje, MQMD.

IBM i *Palabras clave de acción DLQ en IBM i*

Utilice estas palabras clave de acción de la cola de mensajes no entregados para determinar cómo se procesa un mensaje coincidente en la cola de mensajes no entregados de IBM i.

ACTION (DISCARD|IGNORE|RETRY|FWD)
La acción que debe realizarse para los mensajes de la DLQ que coincidan con el patrón definido en esta regla.

DISCARD
Hace que el mensaje se suprima de la DLQ.

IGNORE
Hace que el mensaje se mantenga en la cola de mensajes no entregados (DLQ).

RETRY
Hace que el manejador DLQ intente de nuevo poner el mensaje en la cola destino.

FWD
Hace que el manejador DLQ reenvíe el mensaje a la cola nombrada en la palabra clave FWDQ.

Debe especificar la palabra clave ACTION. El número de intentos efectuados para llevar a cabo una acción está regido por la palabra clave RETRY. La palabra clave RETRYINT de los datos de control controla el intervalo entre intentos.

FWDQ (*QueueName*|&DESTQ|&REPLYQ)
El nombre de la cola de mensajes a la que se reenvía el mensaje al seleccionar la palabra clave ACTION.

QueueName
El nombre de una cola de mensajes. FWDQ(' ') no es válido.

&DESTQ
Hace que el nombre de la cola se tome del campo *DestQName* de la estructura MQDLH.

&REPLYQ
Toma el nombre de cola del campo *ReplyToQ* del descriptor MQMD del mensaje.

Puede especificar REPLYQ (? *) en el patrón de mensaje para evitar mensajes de error, cuando una regla que especifica FWDQ (& REPLYQ) coincide con un mensaje con un campo *ReplyToQ* en blanco.

FWDQM (*QueueManagerNombre*| & DESTQM | & REPLYQM | '' (valor predeterminado))
El gestor de colas de la cola a la que se reenvía un mensaje.

QueueManagerName
El nombre del gestor de colas de la cola a la que se reenvía el mensaje al seleccionar la palabra clave ACTION (FWD).

&DESTQM
Toma el nombre del gestor de colas del campo *DestQMgrName* de la estructura MQDLH.

&REPLYQM
Toma el nombre del gestor de colas del campo *ReplyToQMgr* del descriptor MQMD del mensaje.

' '
FWDQM(' '), que es el valor predeterminado, identifica el gestor de colas local.

HEADER (YES (valor predeterminado) |NO)
Indica si la MQDLH debe permanecer en un mensaje para el que se solicita la acción ACTION (FWD). Por omisión, la MQDLH permanece en el mensaje. La palabra clave HEADER no es válida para acciones distintas de FWD.

PUTAUT (DEF (valor predeterminado) | CTX)
La autorización con la que el manejador DLQ debe transferir los mensajes:

DEF
Transfiere los mensajes con la autorización del propio manejador DLQ.

CTX
Hace que los mensajes se transfieran con la autorización del ID de usuario del contexto del mensaje. Si especifica PUTAUT (CTX), debe poseer autorización para asumir la identidad de otros usuarios.

RETRY (*RecuentoReintentos*|1 (valor predeterminado))
El número de veces, dentro del rango de 1 a 999.999.999, que se intenta una acción (en el intervalo especificado en la palabra clave RETRYINT de los datos de control).

Nota: El número de intentos realizados por el manejador DLQ para implementar una regla concreta es específico de la instancia actual del manejador DLQ; el número total no se conserva al reiniciar. Si reinicia el manejador DLQ, la cuenta de intentos realizados para aplicar una regla se restablece en cero.

IBM i *Convenios de la tabla de reglas de la cola de mensajes no entregados en IBM i*
La tabla de reglas de la cola de mensajes no entregados de IBM i debe seguir los siguientes convenios de sintaxis, estructura y contenido.

- Una tabla de reglas debe contener una regla como mínimo.
- Las palabras clave pueden estar en cualquier orden.
- Una palabra clave sólo se puede incluir una vez en cualquier regla.
- Las palabras clave no son sensibles a las mayúsculas y minúsculas
- Una palabra clave y el valor de su parámetro deben estar separados de las demás palabras clave por un blanco o una coma como mínimo.
- Al principio o al final de una regla y entre palabras clave, signos de puntuación y valores, puede haber cualquier número de espacios en blanco.
- Cada regla debe empezar en una nueva línea.
- Por motivos de portabilidad, la longitud significativa de una línea no debe ser superior a los 72 caracteres.
- Utilice el signo más (+) en una línea como último carácter distinto del blanco para indicar que la regla continúa en el primer carácter distinto del blanco de la línea siguiente. Utilice el signo menos (-) en una línea como último carácter distinto del blanco para indicar que la regla continúa en el principio de la línea siguiente. Puede haber caracteres de continuación dentro de palabras clave y parámetros.

  Por ejemplo:

  ```
  APPLNAME('ABC+
  D')
  ```

  da como resultado 'ABCD'.

  ```
  APPLNAME('ABC-
  D')
  ```

  da como resultado 'ABC D'.
- Puede haber líneas de comentario, que empiezan por un asterisco (*), en cualquier lugar de la tabla de reglas.
- Las líneas en blanco se ignoran.
- Cada entrada de la tabla de reglas del manejador DLQ incluye una o más palabras clave y sus parámetros asociados. Los parámetros deben seguir estas reglas sintácticas:

– Cada valor de parámetro debe contener, como mínimo, un carácter significativo. Las comillas delimitadoras de los valores entre comillas no se consideran significativas. Por ejemplo, serían válidos estos parámetros:

`FORMAT('ABC')`	3 caracteres significativos
`FORMAT(ABC)`	3 caracteres significativos
`FORMAT('A')`	1 carácter significativo
`FORMAT(A)`	1 carácter significativo
`FORMAT(' ')`	1 carácter significativo

Estos parámetros no son válidos porque no contienen caracteres significativos:

```
FORMAT('')
FORMAT( )
FORMAT()
FORMAT
```

– Se admiten caracteres comodín. Puede utilizar el signo de interrogación (?) en lugar de cualquier carácter individual, excepto un blanco final. Puede utilizar el asterisco (*) en lugar de cero o más caracteres adyacentes. El asterisco (*) y el signo de interrogación (?) se interpretan ***siempre*** como caracteres comodín en los valores de parámetros.
– No puede incluir caracteres comodín en los parámetros de estas palabras clave: ACTION, HEADER, RETRY, FWDQ, FWDQM y PUTAUT.
– Los espacios en blanco de cola de los valores de parámetros y de los campos correspondientes en los mensajes de la DLQ no son significativos para realizar comparaciones con comodines. Sin embargo, los espacios en blanco de cabecera e intercalados en series de caracteres entre comillas sí son significativos en las comparaciones con comodines.
– Los parámetros numéricos no pueden incluir el carácter comodín de signo de interrogación (?). Puede incluir un asterisco (*) para que haga las veces de todo un parámetro numérico, pero el asterisco no puede formar parte del parámetro numérico. Por ejemplo, serían válidos estos parámetro numéricos:

`MSGTYPE(2)`	Sólo pueden elegirse mensajes de respuesta
`MSGTYPE(*)`	Puede elegirse cualquier tipo de mensaje
`MSGTYPE('*')`	Puede elegirse cualquier tipo de mensaje

Sin embargo, `MSGTYPE('2*')` no es válido, porque incluye un asterisco (*) como parte de un parámetro numérico.

– Los parámetros numéricos deben estar comprendidos entre 0 y 999.999.999. Si el valor del parámetro está dentro de este rango, se aceptará, incluso si en ese momento no es válido en el campo asociado a la palabra clave. Puede utilizar nombres simbólicos para parámetros numéricos.
– Si un valor de tipo serie es más corto que el campo de MQDLH o MQMD con el que está relacionada la palabra clave, el valor se rellenará con espacios en blanco hasta alcanzar la longitud del campo. Si el valor, excluyendo los asteriscos, es más largo que la longitud del campo, se diagnosticará un error. Por ejemplo, serían válidos todos estos valores de tipo serie para un campo de 8 caracteres:

`'ABCDEFGH'`	8 caracteres
`'A*C*E*G*I'`	5 caracteres, excluidos los asteriscos
`'*A*C*E*G*I*K*M*O*'`	8 caracteres, excluidos los asteriscos

– Las series que contienen espacios en blanco, caracteres en minúsculas o caracteres especiales distintos del punto (.), la barra inclinada (?), el subrayado (_) y el signo de porcentaje (%) deben ir entre comillas simples. Los caracteres en minúsculas que no estén entre comillas se convierten a

mayúsculas. Si la serie incluye una parte entre comillas, se deben utilizar dos comillas simples para indicar el principio y el final de la parte entrecomillada. Cuando se calcula la longitud de la serie, cada aparición de comillas dobles se cuenta como un solo carácter.

IBM i *Cómo se procesa la tabla de reglas DLQ en IBM i*

El manejador de la cola de mensajes no entregados busca en la tabla de reglas una regla con un patrón que coincida con un mensaje en la cola de mensajes no entregados de IBM i.

La búsqueda empieza por la primera regla de la tabla y continúa secuencialmente por la tabla. Cuando se encuentra una regla cuyo patrón coincide, la tabla de reglas intenta realizar la acción indicada por la regla. El manejador DLQ aumenta en una unidad la cuenta de reintentos de la regla cada vez que intenta aplicarla. Si falla el primer intento, el intento se repite hasta que la cuenta de intentos sea igual al número especificado en la palabra clave RETRY. Si todos los intentos fallan, el manejador DLQ busca la siguiente regla coincidente de la tabla.

Este proceso se repite para las siguientes reglas coincidentes hasta que se realiza una acción satisfactoriamente. Cuando se han intentado realizar todas las reglas coincidentes el número de veces especificado en su palabra clave RETRY y todos los intentos han fallado, se presupone ACTION (IGNORE). ACTION (IGNORE) también se presupone si no se encuentra ninguna regla coincidente.

Nota:

1. Sólo se buscan patrones de reglas coincidentes para mensajes de la DLQ que empiecen por MQDLH. Los mensajes que no empiezan por MQDLH se indican periódicamente como erróneos, y permanecen en la DLQ indefinidamente.
2. Todas las palabras clave de patrón pueden tomar su valor predeterminado, por lo que una regla puede constar únicamente de una acción. No obstante, tenga en cuenta que las reglas que son sólo una acción se aplican a todos los mensajes de la cola que tienen MQDLH y que no se han procesado todavía de acuerdo con otras reglas de la tabla.
3. La tabla de reglas se valida al iniciar el manejador DLQ y en ese momento se marcan los errores con un distintivo. (Los mensajes de error que emite el manejador DLQ se describen en el manual Mensajes y códigos de razón). Puede realizar cambios en la tabla de reglas siempre que lo desee, pero los cambios no entrarán en vigor hasta que reinicie el manejador DLQ.
4. El manejador DLQ no altera el contenido de los mensajes, de la MQDLH ni del descriptor de mensaje. El manejador DLQ siempre coloca mensajes en otras colas con la opción de mensaje **MQPMO_PASS_ALL_CONTEXT**.
5. Es posible que no se reconozcan los errores sintácticos consecutivos de la tabla de reglas debido a que la validación de la tabla de reglas elimina la generación de errores repetitivos.
6. El manejador DLQ abre la DLQ con la opción **MQOO_INPUT_AS_Q_DEF** .
7. Pueden ejecutarse simultáneamente varias instancias del manejador DLQ para la misma cola, utilizando la misma tabla de reglas. Pero es más normal que exista una relación unívoca entre una DLQ y un manejador DLQ.

IBM i *Ejemplo de tabla de reglas del manejador DLQ en IBM i*

Ejemplo de código para una tabla de reglas del manejador de la cola de mensajes no entregados en IBM i. Este es un ejemplo de tabla de reglas que contiene una sola entrada de datos de control y varias reglas.

```
*********************************************************************
*    An example rules table for the STRMQMDLQ command        *
*********************************************************************
* Control data entry
* ------------------
* If no queue manager name is supplied as an explicit parameter to
* STRMQMDLQ, use the default queue manager for the machine.
* If no queue name is supplied as an explicit parameter to STRMQMDLQ,
* use the DLQ defined for the local queue manager.
*
inputqm(' ') inputq(' ')
```

```
* Rules
* -----
* We include rules with ACTION (RETRY) first to try to
* deliver the message to the intended destination.

* If a message is placed on the DLQ because its destination
* queue is full, attempt to forward the message to its
* destination queue. Make 5 attempts at approximately
* 60-second intervals (the default value for RETRYINT).

REASON(MQRC_Q_FULL) ACTION(RETRY) RETRY(5)

* If a message is placed on the DLQ because of a put inhibited
* condition, attempt to forward the message to its
* destination queue. Make 5 attempts at approximately
* 60-second intervals (the default value for RETRYINT).

REASON(MQRC_PUT_INHIBITED) ACTION(RETRY) RETRY(5)

* The AAAA corporation is always sending messages with incorrect
* addresses. When we find a request from the AAAA corporation,
* we return it to the DLQ (DEADQ) of the reply-to queue manager
* (&REPLYQM).
* The AAAA DLQ handler attempts to redirect the message.

MSGTYPE(MQMT_REQUEST) REPLYQM(AAAA.*) +
ACTION(FWD) FWDQ(DEADQ) FWDQM(&REPLYQM)

* The BBBB corporation never does things by half measures. If
* the queue manager BBBB.1 is unavailable, try to
* send the message to BBBB.2

DESTQM(bbbb.1) +
action(fwd) fwdq(&DESTQ) fwdqm(bbbb.2) header(no)

* The CCCC corporation considers itself very security
* conscious, and believes that none of its messages
* will ever end up on one of our DLQs.
* Whenever we see a message from a CCCC queue manager on our
* DLQ, we send it to a special destination in the CCCC organization
* where the problem is investigated.

REPLYQM(CCCC.*) +
ACTION(FWD) FWDQ(ALARM) FWDQM(CCCC.SYSTEM)

* Messages that are not persistent run the risk of being
* lost when a queue manager terminates. If an application
* is sending nonpersistent messages, it must be able
* to cope with the message being lost, so we can afford to
* discard the message.

PERSIST(MQPER_NOT_PERSISTENT) ACTION(DISCARD)

* For performance and efficiency reasons, we like to keep
* the number of messages on the DLQ small.
* If we receive a message that has not been processed by
* an earlier rule in the table, we assume that it
* requires manual intervention to resolve the problem.
* Some problems are best solved at the node where the
* problem was detected, and others are best solved where
* the message originated. We do not have the message origin,
* but we can use the REPLYQM to identify a node that has
* some interest in this message.
* Attempt to put the message onto a manual intervention
* queue at the appropriate node. If this fails,
* put the message on the manual intervention queue at
* this node.

REPLYQM('?*') +
ACTION(FWD) FWDQ(DEADQ.MANUAL.INTERVENTION) FWDQM(&REPLYQM)

ACTION(FWD) FWDQ(DEADQ.MANUAL.INTERVENTION)
```

Asegurarse de que todos los mensajes DLQ se han procesado

El manejador de la cola de mensajes no entregados mantiene un registro de todos los mensajes de la DLQ que se han visto pero no eliminado. Asegúrese de que la DLQ contiene el menor número de mensajes posible.

Acerca de esta tarea

Si se utiliza el manejador DLQ como filtro para extraer un pequeño subconjunto de mensajes de la DLQ, el manejador DLQ sigue llevando un registro de los mensajes de la DLQ que no ha procesado. El manejador DLQ tampoco puede garantizar que se vayan a ver los nuevos mensajes que lleguen a la DLQ, incluso si la DLQ está definida como "primero en entrar, primero en salir" (FIFO). Si la cola no está vacía, se realizan nuevas exploraciones periódicas de la DLQ para comprobar todos los mensajes.

Por estas razones, debe asegurarse de que la DLQ contenga el menor número de mensajes posible. Si a los mensajes que no se pueden descartar ni reenviar a otras colas (sea cual sea el motivo) se les permite acumularse en la cola, la carga de trabajo del manejador DLQ aumenta y la propia DLQ corre el riesgo de llenarse.

Para permitir que el manejador DLQ vacíe la DLQ, tome las medidas siguientes:

Procedimiento

- Para los mensajes que de otro modo ignoraría, utilice una acción que mueva los mensajes a otra cola.

  Intente no utilizar el mandato **ACTION (IGNORE)** , que deja mensajes en la DLQ-y recuerde que **ACTION (IGNORE)** se presupone para los mensajes que no están explícitamente direccionado por otras reglas de la tabla. En su lugar, utilice una acción que mueva los mensajes a otra cola. Por ejemplo:

  ```
  ACTION (FWD) FWDQ (IGNORED.DEAD.QUEUE) HEADER (YES)
  ```

- Haga que la regla final de la tabla sea "catch-all" para procesar mensajes que no hayan sido abordados por reglas anteriores de la tabla.

  Por ejemplo, la regla final de la tabla puede ser parecida a esto:

  ```
  ACTION (FWD) FWDQ (REALLY.DEAD.QUEUE) HEADER (YES)
  ```

  Esto reenvía los mensajes que pasan a la regla final de la tabla a la cola REALLY.DEAD.QUEUE, donde se pueden procesar manualmente. Si no tiene una regla de este tipo, es probable que los mensajes permanezcan en la DLQ indefinidamente.

Trabajar con temas administrativos

Utilice los mandatos MQSC para gestionar temas administrativos.

Consulte la sección Mandatos MQSC para obtener información detallada acerca de estos mandatos.

Conceptos relacionados

Objetos de tema administrativo

Tareas relacionadas

“Definición de un tema administrativo” en la página 179
Utilice el mandato MQSC **DEFINE TOPIC** para crear un tema administrativo. Al definir un tema administrativo, puede establecer cada atributo de tema.

“Visualizar atributos de objeto de tema administrativo” en la página 179
Utilice el mandato MQSC **DISPLAY TOPIC** para visualizar un objeto de tema administrativo.

“Cambiar atributos de tema administrativo” en la página 180
Puede cambiar los atributos de tema de dos maneras, utilizando el mandato **ALTER TOPIC** o el mandato **DEFINE TOPIC** con el atributo **REPLACE** .

“Copiar una definición de tema administrativo” en la página 181
Puede copiar una definición de tema utilizando el atributo LIKE en el mandato **DEFINE**.

“Supresión de una definición de tema administrativo” en la página 181
Puede utilizar el mandato MQSC **DELETE TOPIC** para suprimir un tema administrativo.

Definición de un tema administrativo

Utilice el mandato MQSC **DEFINE TOPIC** para crear un tema administrativo. Al definir un tema administrativo, puede establecer cada atributo de tema.

Antes de empezar

Nota: El ejemplo de esta tarea requiere que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Acerca de esta tarea

Los atributos de tema que no se definen explícitamente se heredan del tema administrativo predeterminado, SYSTEM.DEFAULT.TOPIC, creado al realizar la instalación del sistema.

Ejemplo

Por ejemplo, el mandato **DEFINE TOPIC** que sigue, define un tema denominado ORANGE.TOPIC con estas características:

- Se resuelve en la serie de tema ORANGE. Para obtener información sobre cómo se pueden utilizar las series de temas, consulte Combinación de series de temas.
- Los atributos establecidos en ASPARENT utilizan el atributo tal como está definido en el tema padre de este tema. Esta acción se repite ene árbol de temas hasta que se encuentra el tema raíz SYSTEM.BASE.TOPIC. Para obtener más información, consulte Árboles de temas.

```
DEFINE TOPIC (ORANGE.TOPIC) +
TOPICSTR (ORANGE) +
DEFPRTY(ASPARENT) +
NPMSGDLV(ASPARENT)
```

Nota:

- Con la excepción del valor de la serie de tema, todos los valores de atributo que se muestran son los valores predeterminados. Se muestran aquí sólo como ejemplo ilustrativo. Puede omitirlos si está seguro de que los valores predeterminados son los que desea o no han sido modificados. Consulte también “Visualizar atributos de objeto de tema administrativo” en la página 179.
- Si ya tiene un tema administrativo con el nombre ORANGE.TOPIC en el mismo gestor de colas, este mandato falla. Utilice el atributo REPLACE si desea sobrescribir la definición existente de un tema, pero consulte también “Cambiar atributos de tema administrativo” en la página 180.

Referencia relacionada

DEFINE TOPIC

Visualizar atributos de objeto de tema administrativo

Utilice el mandato MQSC **DISPLAY TOPIC** para visualizar un objeto de tema administrativo.

Antes de empezar

Nota: Los ejemplos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Ejemplo

Este mandato muestra todos los temas:

```
DISPLAY TOPIC(ORANGE.TOPIC)
```

Puede visualizar atributos de forma selectiva especificándolos individualmente con el mandato **DISPLAY TOPIC** . Por ejemplo:

```
DISPLAY TOPIC(ORANGE.TOPIC) +
TOPICSTR +
DEFPRTY +
NPMSGDLV
```

Este mandato muestra los tres atributos especificados:

```
AMQ8633: Display topic details.
     TOPIC(ORANGE.TOPIC)                                     TYPE(LOCAL)
     TOPICSTR(ORANGE)                                        DEFPRTY(ASPARENT)
     NPMSGDLV(ASPARENT)
```

Para visualizar los valores ASPARENT del tema tal como se utilizan en el tiempo de ejecución, utilice el mandato **DISPLAY TPSTATUS** . Por ejemplo, utilice:

```
DISPLAY TPSTATUS(ORANGE) DEFPRTY NPMSGDLV
```

El mandato muestra los siguientes detalles:

```
AMQ8754: Display topic status details.
TOPICSTR(ORANGE)            DEFPRTY(0)
NPMSGDLV(ALLAVAIL)
```

Al definir un tema administrativo, éste toma todos los atributos que no especifique explícitamente del tema administrativo predeterminado, que se llama SYSTEM.DEFAULT.TOPIC. Para saber cuáles son estos atributos predeterminados, utilice el siguiente mandato:

```
DISPLAY TOPIC (SYSTEM.DEFAULT.TOPIC)
```

Referencia relacionada

DISPLAY TOPIC

DISPLAY TPSTATUS

Cambiar atributos de tema administrativo

Puede cambiar los atributos de tema de dos maneras, utilizando el mandato **ALTER TOPIC** o el mandato **DEFINE TOPIC** con el atributo **REPLACE** .

Antes de empezar

Nota: Los ejemplos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Ejemplo

Si, por ejemplo, desea cambiar la prioridad predeterminada de los mensajes entregados a un tema denominado ORANGE.TOPIC, para que sea 5, utilice uno de los mandatos siguientes:

- Utilizando el mandato **ALTER**:

  ```
  ALTER TOPIC(ORANGE.TOPIC) DEFPRTY(5)
  ```

  Este mandato cambia un único atributo, el de la prioridad predeterminada del mensaje entregado a este tema a 5; todos los demás atributos permanecen intactos.

- Utilizando el mandato **DEFINE**:

```
DEFINE TOPIC(ORANGE.TOPIC) DEFPRTY(5) REPLACE
```

  Este mandato cambia la prioridad predeterminada de los mensajes entregados a este tema. Todos los demás atributos reciben sus valores predeterminados.

Si modifica la prioridad de los mensajes enviados a este tema, los mensajes existentes no se ven afectados. Sin embargo, cualquier mensaje nuevo utiliza la prioridad especificada si no la proporciona la aplicación de publicación.

Referencia relacionada

ALTER TOPIC

DISPLAY TOPIC

Copiar una definición de tema administrativo

Puede copiar una definición de tema utilizando el atributo LIKE en el mandato **DEFINE**.

Antes de empezar

Nota: Los ejemplos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Ejemplo

El mandato siguiente crea un tema, MAGENTA.TOPIC, con los mismos atributos que el tema original, ORANGE.TOPIC, en lugar de los del tema administrativo predeterminado del sistema. Especifique el nombre del tema que debe copiarse exactamente cómo lo especificó al crear el tema. Si el nombre contiene caracteres en minúscula, especifique el nombre entre comillas simples.

```
DEFINE TOPIC (MAGENTA.TOPIC) +
LIKE (ORANGE.TOPIC)
```

También puede utilizar esta forma del mandato **DEFINE** para copiar una definición de tema, pero realice los cambios en losa tributos del original. Por ejemplo:

```
DEFINE TOPIC(BLUE.TOPIC) +
TOPICSTR(BLUE) +
LIKE(ORANGE.TOPIC)
```

También puede copiar los atributos del tema BLUE.TOPIC en el tema GREEN.TOPIC y especificar que cuando las aplicaciones no puedan entregarse a la cola de suscriptor correcta no se coloquen en la cola de mensajes no entregados. Por ejemplo:

```
DEFINE TOPIC(GREEN.TOPIC) +
TOPICSTR(GREEN) +
LIKE(BLUE.TOPIC) +
USEDLQ(NO)
```

Referencia relacionada

DEFINE TOPIC

Supresión de una definición de tema administrativo

Puede utilizar el mandato MQSC **DELETE TOPIC** para suprimir un tema administrativo.

Antes de empezar

Nota: El ejemplo de esta tarea requiere que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Ejemplo

```
DELETE TOPIC(ORANGE.TOPIC)
```

Las aplicaciones ya no podrán abrir el tema para su publicación o realizar nuevas suscripciones utilizando el nombre del objeto, ORANGE.TOPIC. Las aplicaciones de publicación que tienen el tema abierto pueden continuar publicando la serie de tema resuelta. Las suscripciones ya realizadas en este tema continúan recibiendo publicaciones después de que se haya suprimido el tema.

Las aplicaciones que no hacen referencia a este objeto de tema, pero que utilizan la serie de tema resuelta representada por este objeto de tema, 'NARANJA' en este ejemplo, siguen funcionando. En este caso heredan las propiedades de un objeto de tema superior en el árbol de temas. Para obtener más información, consulte Árboles de temas.

Referencia relacionada
DELETE TOPIC

Trabajar con suscripciones

Utilice los mandatos MQSC para gestionar suscripciones.

Acerca de esta tarea

Las suscripciones pueden ser de tres tipos, definidos en el atributo **SUBTYPE**:

ADMIN
Definida administrativamente por un usuario.

PROXY
Suscripción creada internamente para direccionar publicaciones entre gestores de colas.

API
Creada mediante programación, por ejemplo, con la llamada MQI MQSUB.

Consulte la sección Mandatos MQSC para obtener información detallada acerca de estos mandatos.

Definir una suscripción administrativa

Utilice el mandato MQSC **DEFINE SUB** para crear una suscripción administrativa. También puede utilizar el valor predeterminado definido en la definición de suscripción local predeterminada. O bien puede modificar las características de suscripción a partir de las de la suscripción local predeterminada SYSTEM.DEFAULT.SUB creada al instalar el sistema.

Antes de empezar

Nota: Los ejemplos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Ejemplo

El mandato **DEFINE SUB** siguiente define una suscripción denominada ORANGE con estas características:

- Suscripción duradera, lo que significa que persiste tras el reinicio del gestor de colas, con una caducidad ilimitada.

- Recibe las publicaciones creadas en la serie de tema ORANGE, con las prioridades de mensaje tal como las establecen las aplicaciones de publicación.
- Las publicaciones entregadas para esta suscripción se envían a la cola local SUBQ, que debe definirse antes de definir la suscripción.

```
DEFINE SUB (ORANGE) +
TOPICSTR (ORANGE) +
DESTCLAS (PROVIDED) +
DEST (SUBQ) +
EXPIRY (UNLIMITED) +
PUBPRTY (ASPUB)
```

Nota:

- La suscripción y el nombre de la serie de tema no es tienen que coincidir.
- Salvo los valores del destino y la serie de tema, todos los valores de atributo que se muestran son los valores predeterminados. Se muestran aquí sólo como ejemplo ilustrativo. Puede omitirlos si está seguro de que los valores predeterminados son los que desea o no han sido modificados. Consulte también “Visualización de atributos de suscripciones” en la página 183.
- Si ya tiene una suscripción local con el nombre ORANGE en el mismo gestor de colas, este mandato falla. Utilice el atributo **REPLACE** si desea sobrescribir la definición existente de una cola, pero consulte también “Cambiar atributos de suscripciones locales” en la página 184.
- Si la cola SUBQ no existe, este mandato falla.

Referencia relacionada

DEFINE SUB

Visualización de atributos de suscripciones

Puede utilizar el mandato **DISPLAY SUB** para visualizar los atributos configurados de cualquier suscripción conocida por el gestor de colas.

Antes de empezar

Nota: Los ejemplos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Ejemplo

```
DISPLAY SUB(ORANGE)
```

Puede visualizar de forma selectiva los atributos especificándolos individualmente. Por ejemplo:

```
DISPLAY SUB(ORANGE) +
        SUBID +
        TOPICSTR +
        DURABLE
```

Este mandato muestra los tres atributos especificados de la siguiente manera:

```
AMQ8096: IBM MQ subscription inquired.
     SUBID(414D5120414141202020202020202020EE921E4E20002A03)
     SUB(ORANGE)                                                    TOPICSTR(ORANGE)
     DURABLE(YES)
```

TOPICSTR es la serie de tema resuelta en la que opera este suscriptor. Cuando se define una suscripción para utilizar un objeto de tema, la serie de tema de dicho objeto se utiliza como prefijo para la serie de tema proporcionada al realizar la suscripción. SUBID es un identificador exclusivo asignado por el gestor de colas cuando se crea una suscripción. Se trata de un atributo útil que se debe visualizar para mostrar

porque algunos nombres de suscripciones pueden ser largos o estar en un juego de caracteres diferentes para los que podría resultar poco práctico.

Un método alternativo para visualizar las suscripciones es utilizar el SUBID:

```
DISPLAY SUB +
        SUBID(414D5120414141202020202020202020EE921E4E20002A03) +
        TOPICSTR +
        DURABLE
```

Este mandato ofrece la misma salida que antes:

```
AMQ8096: IBM MQ subscription inquired.
 SUBID(414D5120414141202020202020202020EE921E4E20002A03)
   SUB(ORANGE)                                                          TOPICSTR(ORANGE)
   DURABLE(YES)
```

Las suscripciones de proxy en un gestor de colas no se visualizan de forma predeterminada. Para visualizarlas, especifique el valor PROXY o ALL para **SUBTYPE**.

Puede utilizar el mandato DISPLAY SBSTATUS para ver los atributos de tiempo de ejecución. Por ejemplo, utilice el mandato:

```
DISPLAY SBSTATUS(ORANGE) NUMMSGS
```

Se visualiza la siguiente salida:

```
AMQ8099: IBM MQ subscription status inquired.
   SUB(ORANGE)
   SUBID(414D5120414141202020202020202020EE921E4E20002A03)
   NUMMSGS(0)
```

Al definir una suscripción administrativa, ésta toma los atributos que no se especifican explícitamente de la suscripción predeterminada, que se denomina SYSTEM.DEFAULT.SUB. Para saber cuáles son estos atributos predeterminados, utilice el siguiente mandato:

```
DISPLAY SUB (SYSTEM.DEFAULT.SUB)
```

Referencia relacionada

DISPLAY SUB

Cambiar atributos de suscripciones locales

Puede cambiar los atributos de suscripción de dos maneras, utilizando el mandato **ALTER SUB** o el mandato **DEFINE SUB** con el atributo **REPLACE** .

Antes de empezar

Nota: Los ejemplos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Ejemplo

Si desea cambiar la prioridad de los mensajes entregados a una suscripción denominada ORANGE para que sea 5, utilice uno de los mandatos siguientes:

- Utilizando el mandato **ALTER**:

```
ALTER SUB(ORANGE) PUBPRTY(5)
```

Este mandato cambia un solo atributo, el de la prioridad de los mensajes entregados a esta suscripción a 5; todos los demás atributos permanecen intactos.

- Utilizando el mandato **DEFINE**:

```
DEFINE SUB(ORANGE) PUBPRTY(5) REPLACE
```

Este mandato cambia la prioridad de los mensajes entregados a esta suscripción y todos los demás atributos que tienen a los que se asignan los valores predeterminados.

Si modifica la prioridad de los mensajes enviados a esta suscripción, los mensajes existentes no se verán afectados. Sin embargo, los nuevos mensajes tienen la prioridad especificada.

Referencia relacionada

ALTER SUB

DEFINE SUB

Copiar un definición de suscripción local

Puede copiar una definición de suscripción utilizando el atributo **LIKE** en el mandato **DEFINE**.

Antes de empezar

Nota: Los ejemplos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Ejemplo

```
DEFINE SUB(BLUE) +
       LIKE(ORANGE)
```

You can also copy the attributes of the sub REAL to the sub THIRD.SUB, and specify that the **correlID** of delivered publications is THIRD, rather than the publishers **correlID**. Por ejemplo:

```
DEFINE SUB(THIRD.SUB) +
       LIKE(BLUE) +
       DESTCORL(ORANGE)
```

Referencia relacionada

DEFINE SUB

Supresión de una suscripción local

Puede utilizar el mandato MQSC **DELETE SUB** para suprimir una suscripción local.

Antes de empezar

Nota: Los ejemplos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Ejemplo

```
DELETE SUB(ORANGE)
```

También puede suprimir una suscripción utilizando el SUBID:

```
DELETE SUB SUBID(414D5120414141202020202020202020EE921E4E20002A03)
```

Referencia relacionada
DELETE SUB

Comprobación de mensajes en una suscripción

Cuando se define una suscripción, ésta se asocia a una cola. Los mensajes publicados que coinciden con la suscripción se colocan en esta cola. Utilice los mandatos MQSC para comprobar si hay mensajes en cola actualmente para una suscripción.

Antes de empezar

Nota: Los pasos de esta tarea requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Acerca de esta tarea

Tenga en cuenta que los siguientes mandatos MQSC sólo muestran las suscripciones que han recibido mensajes.

Para comprobar los mensajes que actualmente se encuentran en la cola para una suscripción, siga estos pasos:

Procedimiento

1. Para comprobar si hay mensajes en cola para un tipo de suscripción DISPLAY SBSTATUS(*sub_name*) NUMMSGS, consulte “Visualización de atributos de suscripciones” en la página 183.
2. Si el valor de **NUMMSGS** es mayor que cero, identifique la cola asociada con la suscripción escribiendo DISPLAY SUB(*sub_name*)DEST.
3. Utilizando el nombre de la cola devuelta puede ver los mensajes siguiendo la técnica que se describe en “Examinar colas con el programa de ejemplo” en la página 147.

Referencia relacionada
DISPLAY SBSTATUS

Trabajar con servicios

Los objetos de servicio son una forma de gestionar los procesos adicionales como parte de un gestor de colas. Con los servicios, puede definir programas que se inician y se detienen cuando se inicia o se detiene el gestor de colas. Los servicios de IBM MQ siempre se inician bajo el ID de usuario del usuario que ha iniciado el gestor de colas.

Acerca de esta tarea

Los objetos de servicio pueden ser de los tipos siguientes:

Servidor
Un servidor es un objeto de servicio que tiene el parámetro **SERVTYPE** especificado como SERVER. Un objeto de servicio de servidor es la definición de un programa que se ejecuta cuando se inicia un gestor de colas especificado. Los objetos de servicio de servidor definen programas que normalmente se ejecutan durante un largo periodo de tiempo. Por ejemplo, un objeto de servicio de servidor se puede utilizar para ejecutar un proceso de supervisor desencadenante, como **runmqtrm**.

Sólo se puede ejecutar una instancia de un objeto de servicio de servidor al mismo tiempo. El estado de los objetos de servicio del servidor que están en ejecución se puede supervisar con el mandato MQSC, **DISPLAY SVSTATUS**.

Mandato
Un mandato es un objeto de servicio que tiene el parámetro **SERVTYPE** especificado como COMMAND. Los objetos de servicio de mandato son similares a los objetos de servicio de servidor, aunque se

pueden ejecutar varias instancias de un objeto de servicio de mandato simultáneamente y su estado no se puede supervisar con el mandato MQSC **DISPLAY SVSTATUS**.

Si se ejecuta el mandato MQSC, **STOP SERVICE**, no se realiza ninguna comprobación para determinar si el programa que inició el mandato MQSC, **START SERVICE**, sigue activo antes de detener el programa.

Referencia relacionada

DEFINE SERVICE
DISPLAY SVSTATUS
START SERVICE
STOP SERVICE

Definición de un objeto de servicio

Puede definir un objeto de servicio con el mandato MQSC **DEFINE SERVICE**.

Antes de empezar

Nota: Esta tarea requiere que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Procedimiento

- Defina un objeto de servicio con el mandato MQSC **DEFINE SERVICE**.

  Los atributos que necesita definir son los siguientes:

  SERVTYPE
  Define el tipo del objeto de servicio. Los valores posibles son los siguientes:

  SERVIDOR
  Objeto de servicio del servidor.

  Sólo se puede ejecutar una instancia de un objeto de servicio de servidor al mismo tiempo. El estado de los objetos de servicio de servidor se puede supervisar utilizando el mandato MQSC, **DISPLAY SVSTATUS**.

  Mandato
  Objeto de servicio del mandato.

  Se pueden ejecutar varias instancias de un objeto de servicio de mandato al mismo tiempo. El estado de un objeto de servicio de mandato no se pueden supervisar.

  STARTCMD
  El programa que se ejecuta para iniciar el servicio. Debe especificarse una vía de acceso completa al programa.

  STARTARG
  Argumentos que se han pasado al programa de inicio.

  STDERR
  Especifica la vía de acceso a un archivo al que se debería redirigir la salida de error estándar (stderr) del programa de servicio.

  STDOUT
  Especifica la vía de acceso a un archivo al que se debería redirigir la salida estándar (stdout) del programa de servicio.

  STOPCMD
  El programa que se ejecuta para detener el servicio. Debe especificarse una vía de acceso completa al programa.

  STOPARG
  Argumentos que se han pasado al programa de detención.

CONTROL
Especifica cómo se debe iniciar y detener el servicio.

MANUAL
El servicio no se debe iniciar ni detener de forma automática. Se controla mediante los mandatos **START SERVICE** y **STOP SERVICE**. Éste es el valor predeterminado.

QMGR
El servicio que se define se debe iniciar y detener al mismo tiempo que se inicia y se detiene el gestor de colas.

STARTONLY
El servicio debe iniciarse al mismo tiempo que se inicia el gestor de colas, pero no tiene que detenerse cuando se detiene el gestor de colas.

Tareas relacionadas

"Gestión de servicios" en la página 188

El gestor de colas puede iniciar y detener automáticamente una instancia de un objeto de servicio, o puede iniciarse y detenerse utilizando los mandatos MQSC **START SERVICE** y **STOP SERVICE**.

Referencia relacionada

DEFINE SERVICE
DISPLAY SVSTATUS
START SERVICE
STOP SERVICE

Gestión de servicios

El gestor de colas puede iniciar y detener automáticamente una instancia de un objeto de servicio, o puede iniciarse y detenerse utilizando los mandatos MQSC **START SERVICE** y **STOP SERVICE**.

Antes de empezar

Nota: Esta tarea requiere que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Procedimiento

- Establezca el parámetro **CONTROL** en el gestor de colas para iniciar o detener una instancia de un objeto de servicio automáticamente, o utilice los mandatos MQSC **START SERVICE** y **STOP SERVICE** para hacerlo manualmente.

  Cuando se inicia una instancia de un objeto de servicio, se graba un mensaje en las anotaciones de error del gestor de colas, que contiene el nombre del objeto de servicio y el ID de proceso del proceso iniciado. A continuación se muestra una entrada de registro de ejemplo para un objeto de servicio de servidor que se inicia:

```
02/15/2005 11:54:24 AM - Process(10363.1) User(mqm) Program(amqzmgr0)
Host(HOST_1) Installation(Installation1)
VRMF(7.1.0.0) QMgr(A.B.C)
AMQ5028: The Server 'S1' has started. ProcessId(13031).

EXPLANATION:
The Server process has started.
ACTION:
None.
```

  A continuación se muestra una entrada de registro de ejemplo para un objeto de servicio de mandatos que se inicia:

```
02/15/2005 11:53:55 AM - Process(10363.1) User(mqm) Program(amqzmgr0)
Host(HOST_1) Installation(Installation1)
VRMF(7.1.0.0) QMgr(A.B.C)
```

```
AMQ5030: The Command 'C1' has started. ProcessId(13030).

EXPLANATION:
The Command has started.
ACTION:
None.
```

Cuando se detiene una instancia de un servicio de servidor, se graba un mensaje en las anotaciones de error del gestor de colas, que contiene el nombre del servicio y el ID de proceso del proceso que finaliza. A continuación se muestra una entrada de registro de ejemplo para un objeto de servicio de servidor que se detiene:

```
02/15/2005 11:54:54 AM - Process(10363.1) User(mqm) Program(amqzmgr0)
Host(HOST_1) Installation(Installation1)
VRMF(7.1.0.0) QMgr(A.B.C)
AMQ5029: The Server 'S1' has ended. ProcessId(13031).

EXPLANATION:
The Server process has ended.
ACTION:
None.
```

Tareas relacionadas

"Definición de variables de entorno adicionales en el archivo service.env" en la página 189
Cuando se inicia un servicio, el entorno en el que se inicia el proceso de servicio se hereda del entorno del gestor de colas. Es posible definir variables de entorno adicionales que se deben establecer en el entorno del proceso de servicio añadiendo las variables que desea definir en uno de los archivos de alteración temporal del entorno `service.env`.

Referencia relacionada

STOP SERVICE (detener un servicio) en Multiplatforms

START SERVICE (iniciar un servicio) en Multiplatforms

Definición de variables de entorno adicionales en el archivo service.env

Cuando se inicia un servicio, el entorno en el que se inicia el proceso de servicio se hereda del entorno del gestor de colas. Es posible definir variables de entorno adicionales que se deben establecer en el entorno del proceso de servicio añadiendo las variables que desea definir en uno de los archivos de alteración temporal del entorno `service.env`.

Acerca de esta tarea

Puede añadir las variables de entorno a dos archivos posibles:

- El archivo de ámbito de máquina `service.env`
- El archivo `service.env` del ámbito del gestor de colas

Los dos archivos se procesan, si están disponibles, y las definiciones del archivo del ámbito de gestor de colas tienen preferencia sobre las del archivo del ámbito de máquina.

Puede especificar cualquier variable de entorno en el archivo `service.env` . Por ejemplo, si el servicio IBM MQ ejecuta varios mandatos, puede ser útil establecer la variable de usuario **PATH** en el archivo `service.env` .

Nota: Los valores en los que establece la variable no pueden ser variables de entorno; por ejemplo, CLASSPATH= *%CLASSPATH%* es incorrecto. De forma similar, en Linux PATH= *$PATH* `:/opt/mqm/bin` daría resultados inesperados.

CLASSPATH debe estar en mayúsculas y la sentencia de vía de acceso de clases sólo puede contener literales. Algunos servicios (telemetría por ejemplo) establecen su propia vía de acceso de clase. Se le añade el **CLASSPATH** definido en `service.env` .

El formato de las variables definidas en el archivo `service.env` es una lista de pares de variables de nombre y valor. Cada variable debe definirse en una línea nueva, y cada variable se toma tal como se define explícitamente, incluidos los espacios en blanco.

Procedimiento

- Añada variables de entorno al archivo `service.env` de ámbito de máquina.
  Este archivo se encuentra en:
  - Linux AIX `/var/mqm` en sistemas AIX and Linux.
  - Windows El directorio de datos seleccionado durante la instalación en los sistemas Windows.
- Añada variables de entorno al archivo `service.env` del ámbito del gestor de colas.
  Este archivo se encuentra en el directorio de datos del gestor de colas. Por ejemplo, la ubicación del archivo de alteración temporal de entorno para un gestor de colas denominado QMNAME es:
  - Linux AIX En sistemas AIX and Linux, `/var/mqm/qmgrs/QMNAME/service.env`
  - Windows En sistemas Windows, `C:\ProgramData \IBM \MQ\qmgrs\QMNAME\service.env`

Ejemplo de un archivo `service.env`

```
#********************************************************************#
#*                                                                  *#
#* <N_OCO_COPYRIGHT>                                                *#
#* Licensed Materials - Property of IBM                             *#
#*                                                                  *#
#* 63H9336                                                          *#
#* (C) Copyright IBM Corporation 2005, 2024.                        *#
#*                                                                  *#
#* <NOC_COPYRIGHT>                                                  *#
#*                                                                  *#
#********************************************************************#
#********************************************************************#
#* Module Name: service.env                                         *#
#* Type       : IBM MQ service environment file                     *#
#* Function   : Define additional environment variables to be set   *#
#*              for SERVICE programs.                               *#
#* Usage      : <VARIABLE>=<VALUE>                                  *#
#*                                                                  *#
#********************************************************************#
MYLOC=/opt/myloc/bin
MYTMP=/tmp
TRACEDIR=/tmp/trace
MYINITQ=ACCOUNTS.INITIATION.QUEUE
```

Tareas relacionadas

“Utilización de inserciones sustituibles en definiciones de servicio” en la página 190
Puede sustituir señales en la definición de un objeto de servicio. Los tokens que se sustituyan se reemplazan automáticamente por su texto ampliado cuando se ejecute el programa de servicio.

Referencia relacionada

Descripciones de variables de entorno

Utilización de inserciones sustituibles en definiciones de servicio

Puede sustituir señales en la definición de un objeto de servicio. Los tokens que se sustituyan se reemplazan automáticamente por su texto ampliado cuando se ejecute el programa de servicio.

Acerca de esta tarea

Las señales de sustitución se pueden tomar de la siguiente lista de señales comunes, o de cualquier variable definida en el archivo, `service.env`.

Procedimiento

- Si desea utilizar inserciones reemplazables, inserte la señal entre caracteres + en cualquiera de las series de **STARTCMD**, **STARTARG**, **STOPCMD**, **STOPARG**, **STDOUT** o **STDERR**.

  Para obtener ejemplos de esto, consulte “Utilización de un objeto de servicio de servidor” en la página 191 y “Utilización de un objeto de servicio de mandato” en la página 193.

  A continuación se muestran tokens comunes que se pueden utilizar para sustituir tokens en la definición de un objeto de servicio:

  MQ_INSTALL_PATH
  La ubicación donde se ha instalado IBM MQ.

  MQ_DATA_PATH
  La ubicación del directorio de datos de IBM MQ:

  – Linux AIX En sistemas AIX and Linux, la ubicación del directorio de datos de IBM MQ es `/var/mqm/`

  – Windows En sistemas Windows, la ubicación del directorio de datos de IBM MQ es el directorio de datos seleccionado durante la instalación de IBM MQ

  QMNAME
  El nombre del gestor de colas actual.

  MQ_SERVICE_NAME
  El nombre del servicio.

  MQ_SERVER_PID
  Esta señal solo la pueden utilizar los argumentos **STOPARG** y **STOPCMD**.

  Para objetos de servicio de servidor, esta señal se reemplaza con el ID de proceso del proceso iniciado por los argumentos **STARTCMD** y **STARTARG**. De lo contrario, esta señal se reemplaza por 0.

  MQ_Q_MGR_DATA_PATH
  La ubicación del directorio de datos del gestor de colas.

  MQ_Q_MGR_DATA_NAME
  El nombre transformado del gestor de colas. Para obtener más información sobre la transformación de nombres, consulte Comprender los nombres de archivo IBM MQ.

Utilización de un objeto de servicio de servidor

Estos ejemplos muestran cómo definir, utilizar y modificar un objeto de servicio de servidor para iniciar un supervisor desencadenante u otro programa.

Antes de empezar

Nota: Estos ejemplos requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Estos ejemplos se escriben con caracteres de separador de vía de acceso de estilo UNIX , excepto cuando se indique lo contrario.

Procedimiento

1. **Defina un objeto de servicio de servidor, utilizando el mandato MQSC de `DEFINE SERVICE` :**

```
DEFINE SERVICE(S1) +
CONTROL(QMGR) +
SERVTYPE(SERVER) +
STARTCMD('+MQ_INSTALL_PATH+bin/runmqtrm') +
STARTARG('-m +QMNAME+ -q ACCOUNTS.INITIATION.QUEUE') +
```

```
STOPCMD('+MQ_INSTALL_PATH+bin/amqsstop') +
STOPARG('-m +QMNAME+ -p +MQ_SERVER_PID+')
```

Donde:

+MQ_INSTALL_PATH+ es una señal que representa el directorio de instalación.

+QMNAME+ es una señal que representa el nombre del gestor de colas.

ACCOUNTS.INITIATION.QUEUE es la cola de inicio.

amqsstop es un programa de ejemplo que se proporciona con IBM MQ y que solicita al gestor de colas que interrumpa todas las conexiones del ID de proceso. amqsstop genera mandatos PCF, por lo que el servidor de mandatos debe estar en ejecución.

+MQ_SERVER_PID+ es un token que representa el ID de proceso que se pasa al programa de parada.

Consulte "Utilización de inserciones sustituibles en definiciones de servicio" en la página 190 para ver una lista de las señales habituales.

2. **Una instancia del objeto de servicio de servidor se ejecuta la próxima vez que se inicia el gestor de colas. Sin embargo, puede iniciar una instancia del objeto de servicio de servidor inmediatamente con el mandato MQSC de START SERVICE :**

```
START SERVICE(S1)
```

3. **Mostrar el estado del proceso de servicio del servidor, utilizando el mandato MQSC de DISPLAY SVSTATUS :**

```
DISPLAY SVSTATUS(S1)
```

4. **Modifique el objeto de servicio del servidor y seleccione las actualizaciones reiniciando manualmente el proceso de servicio del servidor, utilizando el mandato MQSC de ALTER SERVICE .**

   El objeto de servicio de servidor se modifica para que la cola de inicio se especifique como JUPITER.INITIATION.QUEUE.

```
ALTER SERVICE(S1) +
STARTARG('-m +QMNAME+ -q JUPITER.INITIATION.QUEUE')
```

   Nota: Un servicio en ejecución no recoge ninguna actualización de su definición de servicio hasta que se reinicia.

5. **Reinicie el proceso de servicio del servidor para que se recoja la modificación, utilizando los mandatos MQSC STOP SERVICE y START SERVICE :**

```
STOP SERVICE(S1)
```

   Seguidos de:

```
START SERVICE(S1)
```

   El proceso de servicio de servidor se reinicia e identifica las modificaciones realizadas en el paso "4" en la página 192.

   Nota: El mandato MQSC, **STOP SERVICE**, solo se puede utilizar si se especifica un argumento **STOPCMD** en la definición de servicio.

Más ejemplos de pasar argumentos

- **Defina un objeto de servicio de servidor para iniciar un programa llamado `runserv` cuando se inicia un gestor de colas.**

Hágalo utilizando el mandato MQSC de **DEFINE SERVICE** .

Este ejemplo se ha escrito con caracteres separadores de vías de acceso de estilo Windows.

Uno de los argumentos que se pasa al programa de inicio es una serie que contiene un espacio. Este argumento debe pasarse como una sola serie de caracteres; Para conseguirlo, se utilizan comillas dobles tal como se muestra en el mandato siguiente para definir el objeto de servicio de mandatos.

```
DEFINE SERVICE(S1) SERVTYPE(SERVER) CONTROL(QMGR) +
STARTCMD('C:\Program Files\Tools\runserv.exe') +
STARTARG('-m +QMNAME+ -d "C:\Program Files\Tools\"') +
STDOUT('C:\Program Files\Tools\+MQ_SERVICE_NAME+.out')


DEFINE SERVICE(S4) +
CONTROL(QMGR) +
SERVTYPE(SERVER) +
STARTCMD('C:\Program Files\Tools\runserv.exe') +
STARTARG('-m +QMNAME+ -d "C:\Program Files\Tools\"') +
STDOUT('C:\Program Files\Tools\+MQ_SERVICE_NAME+.out')
```

Donde:

+QMNAME+ es una señal que representa el nombre del gestor de colas.

`"C:\Program Files\Tools\"` es una serie que contiene un espacio, que se pasará como una única serie.

- **Defina un objeto de servicio de servidor que se pueda utilizar para iniciar automáticamente el supervisor desencadenante cuando se inicie el gestor de colas.**

  Hágalo utilizando el mandato MQSC de **DEFINE SERVICE** .

```
DEFINE SERVICE(TRIG_MON_START) +
CONTROL(QMGR) +
SERVTYPE(SERVER) +
STARTCMD('runmqtrm') +
STARTARG('-m +QMNAME+ -q +IQNAME+')
```

  Donde:

  +QMNAME+ es una señal que representa el nombre del gestor de colas.

  +IQNAME+ es una variable de entorno que define el usuario en uno de los archivos service.env y que representa el nombre de la cola de inicio.

Referencia relacionada

ALTER SERVICE
DEFINE SERVICE
DISPLAY SVSTATUS
START SERVICE
STOP SERVICE

Utilización de un objeto de servicio de mandato

Estos ejemplos muestran cómo definir un objeto de servicio de mandatos para iniciar un programa que graba entradas en el registro del sistema del sistema operativo cuando se inicia o se detiene un gestor de colas.

Antes de empezar

Nota: Estos ejemplos requieren que ejecute el mandato MQSC de **DEFINE SERVICE** . La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Estos ejemplos se escriben con caracteres de separador de vía de acceso de estilo UNIX .

Acerca de esta tarea

En los ejemplos siguientes:

`logger` es un programa de ejemplo que se proporciona con IBM MQ que puede grabar entradas en el registro del sistema del sistema operativo.

+QMNAME+ es una señal que representa el nombre del gestor de colas.

Procedimiento

- Defina un objeto de servicio de mandatos para iniciar un programa que graba entradas en el registro del sistema del sistema operativo cuando se inicia o detiene un gestor de colas:

```
DEFINE SERVICE(S2) +
CONTROL(QMGR) +
SERVTYPE(COMMAND) +
STARTCMD('/usr/bin/logger') +
STARTARG('Queue manager +QMNAME+ starting') +
STOPCMD('/usr/bin/logger') +
STOPARG('Queue manager +QMNAME+ stopping')
```

- Defina un objeto de servicio de mandatos para iniciar un programa que grabe entradas en el registro del sistema del sistema operativo sólo cuando se detenga un gestor de colas:

```
DEFINE SERVICE(S3) +
CONTROL(QMGR) +
SERVTYPE(COMMAND) +
STOPCMD('/usr/bin/logger') +
STOPARG('Queue manager +QMNAME+ stopping')
```

Referencia relacionada

DEFINE SERVICE

Gestión de objetos para desencadenamiento

Estos ejemplos muestran cómo iniciar una aplicación automáticamente cuando se cumplen determinadas condiciones en una cola. Por ejemplo, es posible que desee iniciar una aplicación cuando el número de mensaje de una cola alcanza un número especificado. Este recurso se denomina *desencadenamiento*. Deberá definir los objetos que dan soporte al desencadenamiento.

Antes de empezar

Nota: Estos ejemplos requieren que ejecute mandatos MQSC. La forma de hacer esto varía según la plataforma. Ver AdministrarIBM MQ usando comandos MQSC .

Estos ejemplos se escriben con caracteres de separador de vía de acceso de estilo UNIX .

Acerca de esta tarea

Para obtener una descripción detallada del desencadenamiento, consulte Inicio de aplicaciones IBM MQ utilizando desencadenantes.

Procedimiento

- Defina una cola de aplicación para el desencadenamiento.

  Una cola de aplicación es una cola local que las aplicaciones utilizan para el envío de mensajes, mediante la MQI. El desencadenamiento requiere definir varios atributos de cola en la cola de aplicación.

La activación en sí se habilita mediante el atributo **Trigger** (TRIGGER en los mandatos MQSC). En este ejemplo, se va a generar un suceso desencadenante cuando haya 100 mensajes con una prioridad 5 o superior en la cola local MOTOR.INSURANCE.QUEUE, como se indica a continuación:

```
DEFINE QLOCAL (MOTOR.INSURANCE.QUEUE) +
PROCESS (MOTOR.INSURANCE.QUOTE.PROCESS) +
MAXMSGL (2000) +
DEFPSIST (YES) +
INITQ (MOTOR.INS.INIT.QUEUE) +
TRIGGER +
TRIGTYPE (DEPTH) +
TRIGDPTH (100)+
TRIGMPRI (5)
```

donde:

QLOCAL (MOTOR.INSURANCE.QUEUE)
Es el nombre de la cola de aplicación que se define.

PROCESS (MOTOR.INSURANCE.QUOTE.PROCESS)
Es el nombre de la definición de proceso que describe la aplicación que se va a iniciar mediante un programa supervisor desencadenante.

MAXMSGL (2000)
Especifica la longitud máxima de los mensajes de la cola.

DEFPSIST (YES)
Especifica que los mensajes de esta cola son persistentes de forma predeterminada.

INITQ (MOTOR.INS.INIT.QUEUE)
Es el nombre de la cola de inicio a la que el gestor de colas va a transferir el mensaje desencadenante.

TRIGGER
Es el valor del atributo de desencadenamiento.

TRIGTYPE (DEPTH)
Especifica que se genera un suceso desencadenante cuando el número de mensajes de la prioridad obligatoria (TRIGMPRI) alcanza el número especificado en TRIGDPTH.

TRIGDPTH (100)
Especifica el número de mensajes necesarios para generar un suceso desencadenante.

TRIGMPRI (5)
Es la prioridad de los mensajes que el gestor de colas va a incluir en el recuento para decidir si va a generar un suceso desencadenante. Sólo se incluyen en el recuento los mensajes con prioridad 5 o superior.

- Definir una cola de inicio

  Cuando se produce un suceso desencadenante, el gestor de colas coloca un mensaje desencadenante en la cola de inicio especificada en la definición de la cola de aplicación. Las colas de inicio no tienen valores especiales, pero puede utilizar como guía la siguiente definición de la cola local MOTOR.INS.INIT.QUEUE:

  ```
  DEFINE QLOCAL(MOTOR.INS.INIT.QUEUE) +
  GET (ENABLED) +
  NOSHARE +
  NOTRIGGER +
  MAXMSGL (2000) +
  MAXDEPTH (1000)
  ```

- Definir un proceso

  Utilice el mandato DEFINE PROCESS para crear una definición de proceso. Una definición de proceso define la aplicación que se va a utilizar para procesar los mensajes de la cola de aplicación. La definición de cola de aplicación nombra el proceso que se va a utilizar y, de este modo, asocia la cola de aplicación con la aplicación que se va a utilizar para procesar sus mensajes. Esto se hace mediante

el atributo PROCESS en la cola de aplicación MOTOR.INSURANCE.QUEUE. El siguiente mandato MQSC define el proceso necesario, MOTOR.INSURANCE.QUOTE.PROCESS, identificado en este ejemplo:

```
DEFINE PROCESS (MOTOR.INSURANCE.QUOTE.PROCESS) +
DESCR ('Insurance request message processing') +
APPLTYPE (UNIX) +
APPLICID ('/u/admin/test/IRMP01') +
USERDATA ('open, close, 235')
```

Donde:

MOTOR.INSURANCE.QUOTE.PROCESS
Es el nombre de la definición de proceso.

DESCR ('Insurance request message processing')
Describe el programa de aplicación relacionado con esta definición. Este texto Aparece cuando utiliza el mandato DISPLAY PROCESS y sirve para ayudarle a identificar lo que hace el proceso. Si utiliza espacios en la serie, debe colocarla entre comillas simples.

APPLTYPE (UNIX)
Es el tipo de aplicación que debe iniciarse.

APPLICID ('/u/admin/test/IRMP01')
Es el nombre del archivo ejecutable de la aplicación, especificado como nombre de archivo totalmente calificado. En sistemas Windows, un valor típico de APPLICID sería `c:\appl\test\irmp01.exe`.

USERDATA ('open, close, 235')
Son datos definidos por el usuario, que la aplicación puede utilizar.

- Visualizar atributos de una definición de proceso

  Utilice el mandato DISPLAY PROCESS para examinar los resultados de la definición. Por ejemplo:

```
DISPLAY PROCESS (MOTOR.INSURANCE.QUOTE.PROCESS)


24 : DISPLAY PROCESS (MOTOR.INSURANCE.QUOTE.PROCESS) ALL
AMQ8407: Display Process details.
DESCR ('Insurance request message processing')
APPLICID ('/u/admin/test/IRMP01')
USERDATA (open, close, 235)
PROCESS (MOTOR.INSURANCE.QUOTE.PROCESS)
APPLTYPE (UNIX)
```

  También puede utilizar el mandato MQSC ALTER PROCESS para modificar una definición de proceso existente, y el mandato DELETE PROCESS para suprimir una definición de proceso.

Utilización del programa de utilidad dmpmqmsg entre dos sistemas

El programa de utilidad **dmpmqmsg** (anteriormente *qload*) le permite copiar o mover el contenido de una cola, o sus mensajes, a un archivo.

Visión general

El archivo que cree con **dmpmqmsg** se puede guardar según sea necesario y se puede utilizar en algún momento posterior para volver a cargar los mensajes en la cola.

Importante:

1. El archivo tiene un formato específico que comprende el programa de utilidad. Sin embargo, el archivo también pueden leerlo los usuarios, de modo que puede actualizarlo en un editor antes de volver a cargarlo. Si edita el archivo no debe modificar su formato.
2. El programa de utilidad **dmpmqmsg** se suministra con el conjunto de archivos de tiempo de ejecución para AIX, Linux, and Windows, por lo que está disponible tanto en el servidor como en el cliente de IBM MQ .

Los usos posibles son:

- Guardar en un archivo los mensajes que están en una cola. Posiblemente para fines de archivado y para volver a cargarlo en una cola posteriormente.
- Volver a cargar una cola con los mensajes previamente guardados en un archivo.
- Eliminar los mensajes antiguos de una cola.
- 'Reproducir' los mensajes de prueba desde una ubicación almacenada, incluso manteniendo el tiempo correcto entre los mensajes, si es necesario.

Atención: El SupportPac MO03 utilizaba el parámetro **-l** para especificar enlace local o de cliente. El parámetro **-l** se ha sustituido por el parámetro **-c**.

-P se utiliza ahora para la información de página de códigos en lugar del parámetro **-c**.

Consulte dmpmqmsg para obtener más información sobre el mandato y los parámetros disponibles.

Ejemplo de utilización del programa de utilidad dmpmqmsg en Linux, mediante una máquina de Windows

Tiene un gestor de colas en una máquina de Linux que tiene mensajes en una cola (*Q1*) que desea mover a otra cola (*Q2*) del mismo gestor de colas. Desea iniciar el programa de utilidad **dmpmqmsg** desde una máquina Windows.

La cola (*Q1*) tiene cuatro mensajes que se han añadido mediante la aplicación **amqsput** (gestor de colas local) de ejemplo o la aplicación **amqsputc** (gestor de colas remoto).

En la máquina de Linux puede ver:

```
display ql(Q1) CURDEPTH
        2 : display ql(Q1) CURDEPTH
AMQ8409: Display Queue details.
     QUEUE(Q1)
TYPE(QLOCAL)
     CURDEPTH(4)
```

Establezca la variable de entorno MQSERVER para que apunte al gestor de colas en Linux. Por ejemplo:

```
set MQSERVER=SYSTEM.DEF.SVRCONN/TCP/veracruz.x.com(1414)
```

donde *veracruz* es el nombre de la máquina.

Ejecute el programa de utilidad **dmpmqmsg** para leer en la cola, *Q1*, y almacenar la salida en `c:\temp\mqqload.txt`.

Conéctese como un cliente remoto al gestor de colas, *QM_VER*, que se ejecuta en el host y el puerto Linux establecidos por MQSERVER. Puede establecer la conexión como cliente remoto utilizando el atributo: `-c`.

```
dmpmqmsg -m QM_VER -i Q1 -f c:\temp\mqqload.txt -c
Read      - Files:    0   Messages:       4   Bytes:           22
Written - Files:    1   Messages:       4   Bytes:           22
```

El archivo de salida `c:\temp\mqqload.txt` contiene texto con un formato que el programa de utilidad **dmpmqmsg** entiende.

En la máquina de Windows, emita el mandato **dmpmqmsg** (utilizando la opción `-o` en lugar de la opción `-i`) para cargar la cola (*Q2*) en la máquina de Linux desde un archivo en la máquina de Windows:

```
dmpmqmsg -m QM_VER -o Q2 -f c:\temp\mqqload.txt -c
Read      - Files:    1   Messages:       4   Bytes:           22
Written - Files:    0   Messages:       4   Bytes:           22
```

En la máquina de Linux, observe que ahora hay cuatro mensajes en la cola que se han restaurado a partir del archivo.

```
display ql(Q2) CURDEPTH
        6 : display ql(Q2) CURDEPTH
AMQ8409: Display Queue details.
     QUEUE(Q2)
TYPE(QLOCAL)
     CURDEPTH(4)
```

En la máquina de Linux,

Suprima los mensajes de la cola original.

```
clear qlocal(Q1)
        4 : clear qlocal(Q1)
AMQ8022: IBM MQ queue cleared.
```

Confirme que no haya más mensajes en la cola original:

```
display ql(Q1) CURDEPTH
        5 : display ql(Q1) CURDEPTH
AMQ8409: Display Queue details.
     QUEUE(Q1)
TYPE(QLOCAL)
     CURDEPTH(0)
```

Consulte dmpmqmsg para ver una descripción del mandato y sus parámetros.

Conceptos relacionados

“Ejemplos de utilización del programa de utilidad dmpmqmsg” en la página 198
Formas sencillas en las que puede utilizar el programa de utilidad **dmpmqmsg** (anteriormente **qload**).

Ejemplos de utilización del programa de utilidad dmpmqmsg

Formas sencillas en las que puede utilizar el programa de utilidad **dmpmqmsg** (anteriormente **qload**).

Descargar una cola en un archivo

Utilice las siguientes opciones de la línea de mandatos para guardar en un archivo los mensajes que están en una cola:

```
dmpmqmsg -m QM1 -i Q1 -f c:\myfile
```

Este mandato realiza una copia de los mensajes de la cola y los guarda en el archivo especificado.

Descargar una cola en una serie de archivos

Puede descargar una cola en una serie de archivos utilizando un carácter `insert` en el nombre de archivo. En esta modalidad cada mensaje se graba en un archivo nuevo:

```
dmpmqmsg -m QM1 -i Q1 -f c:\myfile%n
```

Este mandato descarga la cola en los archivos, `myfile1`, `myfile2`, `myfile3`, etc.

Cargar una cola desde un archivo

Para volver a cargar una cola con los mensajes que ha guardado en "Descargar una cola en un archivo" en la página 198, utilice las opciones siguientes en la línea de mandatos:

```
dmpmqmsg -m QM1 -o Q1 -f c:\myfile%n
```

Este mandato descarga la cola en los archivos, `myfile1`, `myfile2`, `myfile3`, etc.

Cargar una cola desde una serie de archivos

Puede cargar una cola desde una serie de archivos utilizando un carácter `insert` en el nombre de archivo. En esta modalidad cada mensaje se graba en un archivo nuevo:

```
dmpmqmsg -m QM1 -o Q1 -f c:\myfile%n
```

Este mandato carga la cola en los archivos, `myfile1`, `myfile2`, `myfile3`, etc.

Copiar los mensajes de una cola a otra cola

Sustituya el parámetro del archivo en "Descargar una cola en un archivo" en la página 198, con otro nombre de cola y utilice las opciones siguientes:

```
dmpmqmsg -m QM1 -i Q1 -o Q2
```

Este mandato permite copiar los mensajes de una cola en otra cola.

Copiar los primeros 100 de una cola a otra cola

Utilice el mandato del ejemplo anterior y añada la opción `-r#100` :

```
dmpmqmsg -m QM1 -i Q1 -o Q2 -r#100
```

Mover los mensajes de una cola a otra cola

Una variación de "Cargar una cola desde un archivo" en la página 199. Observe la diferencia entre utilizar **`-i`** (minúsculas) que solo examina una cola y **`-I`** (mayúsculas) que obtiene de forma destructiva de una cola:

```
dmpmqmsg -m QM1 -I Q1 -o Q2
```

Mover de una cola a otra cola los mensajes con una antigüedad de más de un día

Este ejemplo muestra cómo se utiliza la selección de antigüedad. Se pueden seleccionar mensajes con una antigüedad mayor que o menor que, o en un rango de antigüedad.

```
dmpmqmsg -m QM1 -I Q1 -o Q2 -T1440
```

Mostrar la antigüedad de los mensajes que están actualmente en la cola

Utilice las opciones siguientes en la línea de mandatos:

```
dmpmqmsg -m QM1 -i Q1 -f stdout -dT
```

Trabajar con el archivo de mensajes

Después de descargar el mensaje desde una cola, como en el paso "Descargar una cola en un archivo" en la página 198, es posible que desee editar el archivo.

También es posible que desee cambiar el formato del archivo para utilizar una de las opciones de visualización que no ha especificado en el momento de descargar la cola.

Puede utilizar la utilidad **dmpmqmsg** para volver a procesar el archivo en el formato necesario incluso después de haber tenido lugar la descarga de la cola. Utilice las opciones siguientes en la línea de mandatos.

```
dmpmqmsg -f c:\oldfile -f c:\newfile -dA
```

Consulte dmpmqmsg para ver una descripción del mandato y sus parámetros.

Trabajar con objetos de IBM MQ remotos

Puede administrar objetos IBM MQ en gestor de colas remotos utilizando mandatos MQSC, mandatos PCF o la administrative REST API. Antes de poder utilizar cualquiera de estos métodos, debe definir colas de transmisión y canales entre el gestor de colas local y el gestor de colas remoto, para que los mandatos se puedan enviar al gestor de colas remoto y las respuestas recibidas por el gestor de colas local. De forma alternativa, puede configurar un clúster de gestor de colas y, después, utilizar los mismos métodos de administración remota.

Acerca de esta tarea

Para preparar los gestores de colas para la administración remota, debe configurar los objetos siguientes en el gestor de colas local:

- Un escucha
- Una cola de transmisión que tiene el nombre del gestor de colas remoto.
- Un canal emisor que tiene los detalles de la conexión para el gestor de colas remoto.
- Un canal receptor que tiene el mismo nombre que el canal emisor en el gestor de colas remoto.

También debe configurar los objetos siguientes en el gestor de colas remoto:

- Un escucha
- Una cola de transmisión que tiene el nombre del gestor de colas local.
- Un canal emisor que tiene los detalles de conexión para el gestor de colas local.
- Un canal receptor que tiene el mismo nombre que el canal emisor en el gestor de colas local.

Si desea más información sobre cómo configurar estos objetos, consulte "Configuración de gestores de colas para la administración remota" en la página 201.

De forma alternativa, puede configurar un clúster de gestores de colas. Un *clúster* es un grupo de gestores de colas configurados de forma que puedan comunicarse directamente entre ellos a través de una sola red, sin definiciones complejas de colas de transmisión, canales y colas. Los clústeres pueden configurarse con facilidad y, normalmente, contienen gestores de colas que están relacionados lógicamente de algún modo y necesitan compartir datos o aplicaciones. Incluso los clústeres más pequeños reducen los costes de administración del sistema.

El establecimiento de una red de gestores de colas en un clúster requiere menos definiciones que el establecimiento de un entorno de gestión de colas distribuidas tradicional. Como es necesario efectuar menos definiciones, la red puede configurarse o modificarse más rápida y fácilmente y el riesgo de cometer errores en las definiciones disminuye.

Para configurar un clúster, se necesita una definición de clúster emisor (CLUSSDR) y una definición de clúster receptor (CLUSRCVR) para cada gestor de colas. No se necesita ninguna definición de cola de transmisión ni de cola remota. Los principios de la administración remota son los mismos cuando

se utilizan dentro de un clúster, pero las definiciones propiamente dichas son significativamente más sencillas.

Si desea más información sobre cómo configurar un clúster, consulte Configuración de un clúster de gestores de colas.

Procedimiento

- Para obtener información sobre cómo administrar objetos remotos de IBM MQ, consulte los subtemas siguientes:
  - “Configuración de gestores de colas para la administración remota” en la página 201
  - “Gestión del servidor de mandatos para la administración remota” en la página 205
  - “Emisión de mandatos MQSC en un gestor de colas remoto” en la página 206
  - “Conversión de datos entre juegos de caracteres codificados” en la página 208

Configuración de gestores de colas para la administración remota

Puede administrar un gestor de colas remoto desde un gestor de colas local utilizando los mandatos administrative REST API, MQSC o PCF. El gestor de colas remoto podría estar en el mismo sistema, en una instalación diferente, o en un sistema distinto con el mismo entorno, o un entorno diferente de IBM MQ. Antes de poder administrar un gestor de colas de forma remota desde un gestor de colas local, debe crear un canal emisor y receptor, un escucha y una cola de transmisión en cada gestor de colas. Estos canales y colas permiten que los mandatos se envíen al gestor de colas remoto y que las respuestas se reciban en el gestor de colas local. El procedimiento para crear estas colas y canales es el mismo si desea utilizar los mandatos administrative REST API, MQSC o PCF.

Antes de empezar

- El procedimiento siguiente utiliza los gestores de colas de ejemplo `source.queue.manager` y `target.queue.manager`. Debe crear e iniciar estos gestores de colas en el sistema para seguir estos pasos, o sustituir sus propios nombres de gestor de colas en los pasos relevantes.
- El procedimiento siguiente utiliza TCP/IP como tipo de transporte. Debe saber la dirección IP de ambos sistemas para completar esta tarea.
- El procedimiento siguiente crea escuchas que utilizan los puertos de red 1818 en el sistema local y 1819 en el sistema remoto. Puede utilizar otros puertos, pero debe sustituir los valores de puerto en los pasos apropiados.
- Debe ejecutar los mandatos en el procedimiento localmente o a través de un recurso de red como, por ejemplo, Telnet.

Acerca de esta tarea

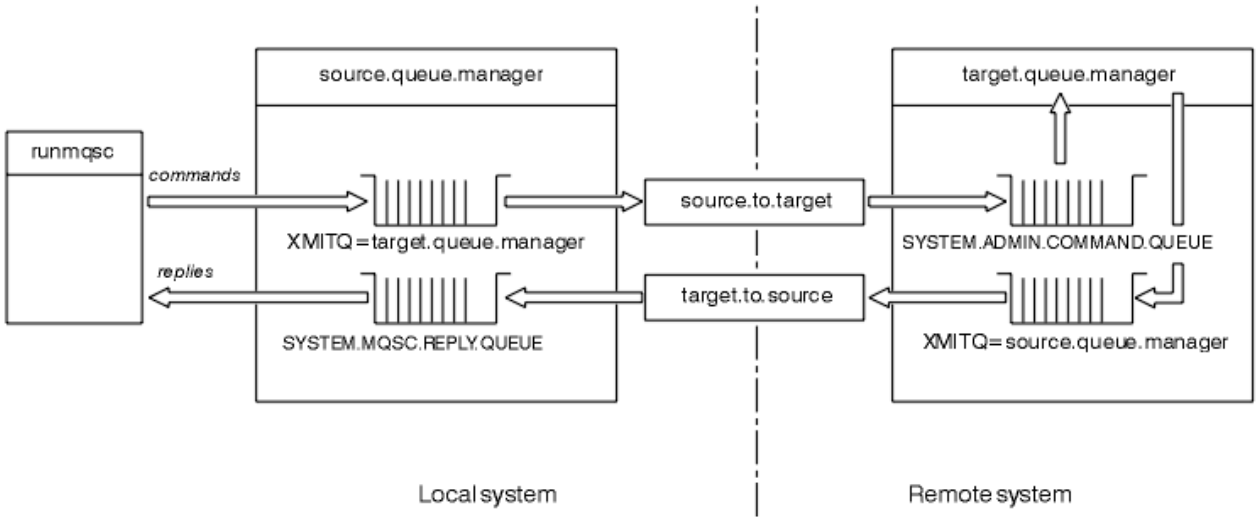


Figura 15. Preparación de canales y colas para la administración remota

Figura 15 en la página 202 muestra la configuración gestores de colas, colas y canales que necesita para la administración remota.

- El objeto `source.queue.manager` es el gestor de colas de origen desde el cual puede emitir mandatos administrative REST API, MQSC o PCF y al que se devuelven los resultados de estos mandatos.
- El objeto `target.queue.manager` es el nombre del gestor de colas de destino, que procesa los mandatos y genera todos los mensajes de operador.
- Los mandatos se colocan en la cola de transmisión que tiene el mismo nombre que el gestor de colas remoto. En este caso, `target.queue.manager`. Una cola de transmisión es una cola local especializada que contiene temporalmente mensajes antes de que el MCA los extraiga y los envíe al gestor de colas remoto.
- Los mandatos se envían mediante el canal `source.to.target` a la `SYSTEM.ADMIN.COMMAND.QUEUE` en el gestor de colas remoto. Cada extremo del canal tiene una definición distinta. Un extremo es un emisor y el otro es un receptor. Las dos definiciones deben tener el mismo nombre y, juntas, formar un solo canal de mensajes.
- La salida del mandato en la cola de transmisión remota que tiene el mismo nombre que el gestor de colas local desde el cual se envió el mandato. En este caso, `source.queue.manager`.
- La salida se envía a través del canal `target.to.source` a una cola de respuesta apropiada, donde se toma y el mandato original genera la salida.

Procedimiento

1. En el gestor de colas del sistema remoto, asegúrese de que la cola del mandato `SYSTEM.ADMIN.COMMAND.QUEUE` está presente. Esta cola se crea de forma predeterminada cuando se crea un gestor de colas.
2. En el sistema remoto, compruebe que el servidor de mandatos se está ejecutando en el gestor de colas. Si el servidor de mandatos no se está ejecutando, la administración remota no es posible.
   a) Inicie **runmqsc** para el gestor de colas. Por ejemplo, para el gestor de colas `target.queue.manager`, especifique el mandato siguiente:

   ```
   runmqsc target.queue.manager
   ```

   b) Muestre el estado del servidor de mandatos especificando el mandato siguiente:

```
DISPLAY QMSTATUS CMDSERV
```

c) Salga del indicador de mandatos de **runmqsc** especificando el mandato siguiente:

```
end
```

d) Si el servidor de mandatos no se ha iniciado, inícielo. Por ejemplo, para el gestor de colas `target.queue.manager`, especifique el mandato siguiente:

```
strmqcsv target.queue.manager
```

3. Defina los canales, el escucha y la cola de transmisión en el gestor de colas local:

a) Inicie **runmqsc** para el gestor de colas. Por ejemplo, para el gestor de colas `source.queue.manager`, especifique el mandato siguiente:

```
runmqsc source.queue.manager
```

b) Defina el canal emisor. Este canal emisor debe tener el mismo nombre que el canal receptor en el gestor de colas remoto. Por ejemplo, especifique el mandato MQSC siguiente, sustituyendo el valor para **CONNAME** con la dirección IP para el gestor de colas remoto y el número de puerto del escucha:

```
DEFINE CHANNEL ('source.to.target') +
CHLTYPE(SDR) +
CONNAME (localhost:1819) +
XMITQ ('target.queue.manager') +
TRPTYPE(TCP)
```

c) Defina el canal receptor. Este canal receptor debe tener el mismo nombre que el canal emisor en el gestor de colas remoto. Por ejemplo, especifique el mandato siguiente:

```
DEFINE CHANNEL ('target.to.source') +
CHLTYPE(RCVR) +
TRPTYPE(TCP)
```

d) Defina el escucha en el gestor de colas local. Por ejemplo, especifique el mandato siguiente:

```
DEFINE LISTENER ('source.queue.manager') +
TRPTYPE (TCP) +
PORT (1818)
```

e) Defina la cola de transmisión en el gestor de colas local. Esta cola de transmisión debe tener el mismo nombre que el gestor de colas remoto. Por ejemplo, especifique el mandato siguiente:

```
DEFINE QLOCAL ('target.queue.manager') +
USAGE (XMITQ)
```

f) Inicie el escucha. Por ejemplo, especifique el mandato siguiente:

```
START LISTENER ('source.queue.manager')
```

g) Salga del indicador de mandatos de **runmqsc** especificando el mandato siguiente:

```
end
```

4. Defina los canales, el escucha y la cola de transmisión en el gestor de colas remoto:

a) Inicie **runmqsc** para el gestor de colas. Por ejemplo, para el gestor de colas `target.queue.manager`, especifique el mandato siguiente:

```
runmqsc target.queue.manager
```

b) Defina el canal emisor. Este canal emisor debe tener el mismo nombre que el canal receptor en el gestor de colas local. Por ejemplo, especifique el mandato MQSC siguiente, sustituyendo el valor para **CONNAME** con la dirección IP para el gestor de colas local y el número de puerto para el escucha:

```
DEFINE CHANNEL ('target.to.source') +
CHLTYPE(SDR) +
CONNAME (localhost:1818) +
XMITQ ('source.queue.manager') +
TRPTYPE(TCP)
```

c) Defina el canal receptor. Este canal receptor debe tener el mismo nombre que el canal emisor en el gestor de colas local. Por ejemplo, especifique el mandato siguiente:

```
DEFINE CHANNEL ('source.to.target') +
CHLTYPE(RCVR) +
TRPTYPE(TCP)
```

d) Defina el escucha. Por ejemplo, especifique el mandato siguiente:

```
DEFINE LISTENER ('target.queue.manager') +
TRPTYPE (TCP) +
PORT (1819)
```

e) Defina la cola de transmisión. Esta cola de transmisión debe tener el mismo nombre que el gestor de colas local. Por ejemplo, especifique el mandato siguiente:

```
DEFINE QLOCAL ('source.queue.manager') +
USAGE (XMITQ)
```

f) Inicie el escucha. Por ejemplo, especifique el mandato siguiente:

```
START LISTENER ('target.queue.manager')
```

g) Salga del mandato **runmqsc** especificando el mandato siguiente:

```
end
```

5. Inicie el canal emisor en el sistema local.

a) Inicie **runmqsc** para el gestor de colas. Por ejemplo, para el gestor de colas `source.queue.manager`, especifique el mandato siguiente:

```
runmqsc source.queue.manager
```

b) Inicie el canal emisor. Por ejemplo, especifique el mandato siguiente:

```
START CHANNEL ('source.to.target')
```

c) Salga del mandato **runmqsc** especificando el mandato siguiente:

```
end
```

6. Inicie el canal emisor en el sistema remoto.

a) Inicie runmqsc para el gestor de colas. Por ejemplo, para el gestor de colas `target.queue.manager`, especifique el mandato siguiente:

```
runmqsc target.queue.manager
```

b) Inicie el canal emisor. Por ejemplo, especifique el mandato siguiente:

```
START CHANNEL ('target.to.source')
```

c) Salga del mandato **runmqsc** especificando el mandato siguiente:

```
end
```

7. Pruebe que la configuración se ha completado correctamente enviando un mandato MQSC desde el sistema local al gestor de colas remoto:
   a) Inicie el indicador de mandatos de **runmqsc** para el gestor de colas remoto desde el sistema local. Por ejemplo, especifique el mandato siguiente:

```
runmqsc -w 30 -m source.queue.manager target.queue.manager
```

   b) Muestre las colas en el gestor de colas remoto especificando el mandato siguiente:

```
DISPLAY QUEUE (*)
```

   En caso de éxito, se muestra una lista de colas del gestor de colas remoto.

   c) Si estos pasos no funcionan, compruebe que los canales en ambos sistemas están en un estado de ejecución. Si los canales no se están ejecutando y no se inician, compruebe que los canales y las colas de transmisión se han configurado correctamente y que el servidor de mandatos se está ejecutando. Por ejemplo, compruebe que se ha especificado el CONNAME correcto para los canales emisores y que las colas de transmisión tienen los nombres correctos. Asimismo, consulte los registros del gestor de colas para ver excepciones de seguridad que podrían ayudar a resolver el problema.

Resultados

Los gestores de colas se han configurado para administrar de forma remota el gestor de colas remoto desde el sistema local.

Qué hacer a continuación

- Obtenga más información sobre la administración remota utilizando mandatos MQSC: “Emisión de mandatos MQSC en un gestor de colas remoto” en la página 206
- Obtenga más información sobre cómo escribir programas de administración utilizando mandatos PCF: “Utilización de los formatos de mandato programable de IBM MQ” en la página 27.
- Obtenga más información sobre cómo utilizar la administrative REST API para la administración remota: “Administración remota mediante REST API” en la página 81.

Gestión del servidor de mandatos para la administración remota

Cada gestor de colas tiene un servidor de mandatos asociado. El servidor de mandatos procesa todos los mandatos entrantes procedentes de gestores de colas remotos o los mandatos PCF procedentes de aplicaciones. Presenta los mandatos al gestor de colas para su proceso y devuelve un código de terminación o un mensaje de operador. Puede iniciar, detener y mostrar el estado del servidor de mandatos. Es indispensable tener un servidor de mandatos para toda la administración que implique mandatos PCF, la MQAI y también para la administración remota.

Antes de empezar

En función del valor del atributo del gestor de colas, **SCMDSERV**, el servidor de mandatos se inicia automáticamente cuando se inicia el gestor de colas o debe iniciarse manualmente. Si el servidor de mandatos se inicia automáticamente, no puede utilizar los mandatos strmqcsv o endmqcsv para iniciar y detener el servidor de mandatos. Puede cambiar el valor del atributo **SCMDSERV** utilizando el mandato MQSC **ALTER QMGR**. Por omisión, el servidor de mandatos se inicia automáticamente.

Al detener un gestor de colas también se detiene el servidor de mandatos asociado al mismo.

Procedimiento

- Muestre el estado del servidor de mandatos.
  a) Inicie el indicador de mandatos de **runmqsc** para el gestor de colas adecuado especificando el mandato siguiente:

```
runmqsc target.queue.manager
```

  donde `target.queue.manager` es el gestor de colas para el cual se está mostrando el servidor de mandatos.
  b) Visualice el estado del servidor de mandatos especificando el mandato MQSC siguiente:

```
DISPLAY QMSTATUS CMDSERV
```

  c) Salga del indicador de mandatos de **runmqsc** especificando el mandato siguiente:

```
end
```

- Si el servidor de mandatos no está establecido para iniciarse automáticamente, inicie el servidor de mandatos especificando el mandato siguiente:

```
strmqcsv target.queue.manager
```

  donde `target.queue.manager` es el gestor de colas para el cual se está iniciando el servidor de mandatos.
- Si el servidor de mandatos no está establecido para iniciarse automáticamente, detenga el servidor de mandatos especificando el mandato siguiente:

```
endmqcsv target.queue.manager
```

  donde `target.queue.manager` es el gestor de colas para el cual se está deteniendo el servidor de mandatos.

  De forma predeterminada, el servidor de mandatos se detiene de una forma controlada. Puede detener el servidor de mandatos inmediatamente añadiendo el código `-i` al mandato.

Emisión de mandatos MQSC en un gestor de colas remoto

Después de configurar gestores de colas para la administración remota, puede utilizar un formato concreto del mandato **runmqsc** en un sistema local para ejecutar mandatos MQSC en un gestor de colas remoto. Cada mandato se envía como un PCF de escape a la cola de mandatos, SYSTEM.ADMIN.COMMAND.QUEUE, del gestor de colas remoto. Se reciben respuestas en la cola SYSTEM.MQSC.REPLY.QUEUE.

Antes de empezar

Debe completar los pasos en “Configuración de gestores de colas para la administración remota” en la página 201 para configurar canales, colas de transmisión y el servidor de mandatos antes de poder administrar remotamente un gestor de colas utilizando mandatos MQSC.

Procedimiento

1. Asegúrese de que el servidor de mandatos se está ejecutando en el gestor de colas remoto.

   Para obtener más información sobre cómo iniciar el servidor de mandatos en un gestor de colas, consulte el apartado “Gestión del servidor de mandatos para la administración remota” en la página 205.
2. En el gestor de colas de origen, puede ejecutar mandatos MQSC de una de estas dos maneras:

- De forma interactiva, iniciando **runmqsc** con los mandatos siguientes:
  - Si el gestor de colas remoto está en z/OS, especifique el mandato siguiente:

```
runmqsc -w 30 -x -m source.queue.manager target.queue.manager
```

  - Si el gestor de colas remoto está en Multiplatforms, especifique el mandato siguiente:

```
runmqsc -w 30 -m source.queue.manager target.queue.manager
```

- Desde una línea de mandatos:
  a. Coloque los mandatos MQSC que se van a ejecutar en el sistema remoto en un archivo de texto, un mandato por línea.
  b. Verifique los mandatos MQSC en el gestor de colas local utilizando el código -v en el mandato **runmqsc**. El código -v comprueba que los mandatos son válidos, pero no los ejecuta. Tenga en cuenta que algunos mandatos podrían fallar si son aplicables al gestor de colas remoto, pero no son aplicables al gestor de colas local.

```
runmqsc -v source.queue.manager < myCmdFile.in > results.out
```

     `myCmdFile.in` contiene los mandatos MQSC que se deben comprobar y el archivo `results.out` contiene los resultados de verificación de los mandatos.
  c. Ejecute el archivo de mandatos en el gestor de colas remoto especificando uno de los mandatos siguientes:
     - Si el gestor de colas remoto está en z/OS, especifique el mandato siguiente:

```
runmqsc -w 30 -x -m source.queue.manager target.queue.manager < myCmdFile.in >
results.out
```

     - Si el gestor de colas remoto está en Multiplatforms, especifique el mandato siguiente:

```
runmqsc -w 30 -m source.queue.manager target.queue.manager < myCmdFile.in >
results.out
```

Los parámetros utilizados son los parámetros siguientes:

-w *segundos*
Especifica que los mandatos MQSC se ejecutan en la modalidad directa, donde los mandatos se colocan en la cola de entrada del servidor de mandatos y se ejecutan en orden.

La variable *segundos* especifica el periodo de tiempo de espera, en segundos, de una respuesta del gestor de colas remoto. Las respuestas recibidas después de este periodo de tiempo se descartan, pero los mandatos MQSC se siguen ejecutando en el gestor de colas remoto. El mensaje siguiente se genera en el gestor de colas local cuando el mandato excede el tiempo de espera:

```
AMQ8416: MQSC timed out waiting for a response from the command server.
```

Cuando acabe de emitir mandatos MQSC, el gestor de colas local mostrará cualquier respuesta con tiempo de espera excedido que haya llegado y descartará todas las respuestas posteriores.

-x
Especifica que el gestor de colas remoto es un gestor de colas z/OS.

-m *NombreGestorColasLocal*
Especifica el nombre del gestor de colas local que desea utilizar para enviar mandatos al gestor de colas remoto.

Qué hacer a continuación

Si tiene dificultades para ejecutar mandatos MQSC de forma remota:

- Compruebe que se está ejecutando el gestor de colas remoto.
- Compruebe que el servidor de mandatos se está ejecutando en el sistema remoto.
- Compruebe que el intervalo de desconexión del canal no ha caducado. Por ejemplo, si se ha iniciado un canal, pero se ha concluido al cabo de un tiempo. Esto es especialmente importante si inicia manualmente los canales.
- Asegúrese de que las solicitudes que se envían desde el gestor de colas local tienen sentido en el gestor de colas de destino. Por ejemplo, las solicitudes que incluyen parámetros que no están soportados en el gestor de colas remoto.
- Vea también Resolución de problemas con mandatos MQSC.

Conversión de datos entre juegos de caracteres codificados

El gestor de colas puede convertir datos de mensaje en formatos definidos de IBM MQ (también conocidos como formatos incorporados) de un juego de caracteres codificado a otro, siempre que ambos juegos de caracteres estén relacionados con un único idioma o un grupo de idiomas similares.

Por ejemplo, está soportada la conversión entre los juegos de caracteres codificados con los identificadores (CCSID) 850 y 500, porque ambos se aplican a idiomas de Europa Occidental.

Para las conversiones de caracteres de nueva línea (NL) EBCDIC a ASCII, consulte Stanza de todos los gestores de colas del archivo mqs.ini y la variable de entorno **AMQ_CONVEBCDICNEWLINE** .

Las conversiones soportadas están definidas en Procesamiento de conversión de datos.

La conversión entre los CCSID 37 y 500 está soportada en IBM MQ Appliance, Windows, Linuxy macOS.

Cuando un gestor de colas no puede convertir mensajes en formatos incorporados

El gestor de colas no puede convertir automáticamente mensajes en formatos incorporados si sus CCSID respectivos representan grupos de idiomas nacionales distintos. Por ejemplo, la conversión entre el CCSID 850 y el CCSID 1025 (que es un juego de caracteres codificado EBCDIC para idiomas que utilizan el alfabeto cirílico) no está soportada porque muchos de los caracteres de uno de los juegos de caracteres codificados no pueden representarse en el otro. Si tiene una red de gestores de colas que trabajan en distintos idiomas nacionales y la conversión de datos entre algunos de los juegos de caracteres codificados no está soportada, puede habilitar una conversión predeterminada.

Para las plataformas a las que se aplica `ccsid_part2.tbl` , consulte “Especificar la conversión de datos predeterminada” en la página 211 Utilización de `ccsid_part2.tbl` para obtener más información. La conversión de datos predeterminada en plataformas distintas de las a las que se aplica el archivo `ccsid_part2.tbl` se describe en “Conversión de datos predeterminada” en la página 209.

Soporte de conversión de datos Unicode ampliado

El producto soporta todos los caracteres Unicode definidos en el estándar Unicode 8.0 en la conversión de datos. Esto incluye soporte completo para UTF-16, incluyendo pares de sustitución (un par de caracteres UTF-16 de 2 bytes en el rango de `X'D800'` a `X'DFFF'` que representa un punto de código Unicode por encima de U+FFFF).

También se admite la combinación de secuencias de caracteres en casos en los que se correlacione un carácter compuesto anteriormente en un CCSID con una secuencia de caracteres de combinación en otro CCSID.

La conversión de datos a y desde Unicode y los CCSID 1388, 1390, 1399, 4933, 5488 y 16884 se ha ampliado en algunas plataformas para dar soporte a todos los puntos de código definidos actualmente para estos CCSID, incluyendo aquellos que se correlacionan con puntos de código en planos suplementarios de Unicode.

En el caso de los CCSID 1390, 1399 y 16884, esto incluye caracteres definidos en el estándar JIS X 0213 (JIS2004).

También se ha añadido soporte para la conversión a y desde Unicode y seis nuevos CCSID (de 1374 a 1379).

Archivo ccsid_part2.tbl

Se proporciona un archivo adicional, `ccsid_part2.tbl`.

El archivo `ccsid_part2.tbl` tiene prioridad sobre el archivo `ccsid.tbl` y:

- Permite añadir o modificar entradas CCSID
- Especificar la conversión de datos predeterminada
- Especificar datos para diferentes niveles de mandatos

`ccsid_part2.tbl` sólo es aplicable a las plataformas siguientes:

- Linux - todas las versiones
- Windows

Windows En IBM MQ for Windows, `ccsid_part2.tbl` se encuentra en el directorio *MQDataRoot*`\conv\table` de forma predeterminada. Además, en IBM MQ for Windows, registra todos los conjuntos de códigos soportados.

Linux En IBM MQ for Linux, `ccsid_part2.tbl` se encuentra en el directorio *MQDataRoot*`/conv/table`, y los conjuntos de código soportado se almacenan en las tablas de conversión que proporciona IBM MQ.

Aunque el archivo `ccsid_part2.tbl` sustituye al archivo `ccsid.tbl` existente en versiones anteriores de IBM MQ para proporcionar información de CCSID adicional, el archivo `ccsid.tbl` sigue siendo analizado por IBM MQ y, por tanto, no se debe suprimir.

Para obtener más información, consulte "El archivo ccsid_part2.tbl" en la página 210.

Archivo `ccsid.tbl`

En plataformas distintas de aquellas a las que se aplica `ccsid_part2.tbl`, el archivo `ccsid.tbl` se utiliza para los fines siguientes:

- AIX En AIX, el sistema operativo mantiene el conjunto de códigos soportado internamente.
- Especifica cualquier conjunto de códigos adicional. Para especificar conjuntos de códigos adicionales, debe editar `ccsid.tbl` (en el archivo se proporciona orientación sobre cómo hacerlo).
- Especifica cualquier conversión de datos predeterminada.

Puede actualizar la información registrada en `ccsid.tbl`; es posible que desee hacerlo si, por ejemplo, un release futuro del sistema operativo da soporte a conjuntos de caracteres codificados adicionales.

Conversión de datos predeterminada

Si configura canales entre dos máquinas en las que no se da soporte normalmente a la conversión de datos, deberá habilitar la conversión de datos predeterminada para que funcionen los canales.

En plataformas distintas de aquellas a las que se aplica `ccsid_part2.tbl`, para habilitar la conversión de datos predeterminada, edite el archivo `ccsid.tbl` para especificar un CCSID EBCDIC predeterminado y un CCSID ASCII predeterminado. Las instrucciones correspondientes están incluidas en el archivo. Debe hacer esto en todas las máquinas que se conectarán utilizando los canales. Reinicie el gestor de colas para que el cambio surta efecto.

El proceso de conversión de datos predeterminada es el siguiente:

- Si la conversión entre los CCSID de origen y de destino no está soportada, pero los CCSID de los entornos de origen y de destino son ambos EBCDIC o son ambos ASCII, los datos de tipo carácter se pasan a la aplicación de destino sin realizar ninguna conversión.
- Si un CCSID representa un juego de caracteres codificado ASCII, y el otro representa un juego de caracteres codificado EBCDIC, IBM MQ convierte los datos utilizando los CCSID de conversión de datos predeterminados definidos en ccsid.tbl.

Nota: Intente limitar los caracteres que se convierten a aquellos que tengan los mismos valores de código en el juego de caracteres codificado especificado para el mensaje y en el juego de caracteres codificado predeterminado. Si utiliza sólo el conjunto de caracteres que es válido para los nombres de objeto de IBM MQ (tal como se define en Denominación de objetos de IBM MQ) en general, cumplirá este requisito. Se producen excepciones con los CCSID EBCDIC 290, 930, 1279 y 5026 utilizados en Japón, ya que los caracteres en minúsculas tienen códigos distintos de los utilizados en otros CCSID EBCDIC.

Conversión de mensajes en formatos definidos por el usuario

El gestor de colas no puede convertir mensajes en formatos definidos por el usuario de un juego de caracteres codificado a otro. Si necesita convertir datos en un formato definido por el usuario, debe facilitar una salida de conversión de datos para cada formato de este tipo. No utilice identificadores CCSID predeterminados para convertir datos de tipo carácter en formatos definidos por el usuario. Si desea más información sobre cómo convertir los datos en formatos definidos por el usuario y sobre cómo escribir las salidas de la conversión de datos, consulte Escritura de salidas de conversión de datos.

Cambio del CCSID del gestor de colas

Cuando haya utilizado el atributo **CCSID** del mandato **ALTER QMGR** para cambiar el CCSID del gestor de colas, detenga y reinicie el gestor de colas para asegurarse de que todas las aplicaciones en ejecución, incluidos el servidor de mandatos y los programas de canal, se detengan y se reinicien.

Esto es necesario porque cualquier aplicación que esté en ejecución cuando se cambia el CCSID del gestor de colas sigue utilizando el CCSID existente.

El archivo `ccsid_part2.tbl`

El archivo `ccsid_part2.tbl` se utiliza para proporcionar información CCSID adicional. El archivo `ccsid_part2.tbl` sustituye el archivo `ccsid.tbl` que se ha utilizado antes de IBM MQ 9.0.

Nota: El archivo `ccsid.tbl`, que se utilizaba antes de IBM MQ 9.0 para suministrar información de CCSID adicional, continúa siendo analizado por IBM MQ y no debe suprimirse. Sin embargo, las entradas en `ccsid_part2.tbl` tienen prioridad sobre otras entradas de `ccsid.tbl`.

Deberá utilizar `ccsid_part2.tbl` en lugar de `ccsid.tbl` porque `ccsid_part2.tbl`:

- Contiene soporte para los valores de codificación Unicode. A partir de IBM MQ 9.0, admite todos los caracteres Unicode definidos en el estándar Unicode 8.0 en la conversión de datos, incluido soporte completo para UTF-16. Para obtener más información, consulte “Conversión de datos entre juegos de caracteres codificados” en la página 208.
- Le permite especificar la versión de las entradas de CCSID, para que las entradas sólo sean aplicables a los niveles de mandatos seleccionados.

Puede utilizar el archivo `ccsid_part2.tbl` para:

- Añadir o modificar entradas CCSID
- Especificar la conversión de datos predeterminada
- Especificar datos para diferentes niveles de mandatos

El archivo `ccsid_part2.tbl` sólo es aplicable a las plataformas siguientes:

- Linux - todas las versiones
- Windows

La ubicación del archivo `ccsid_part2.tbl` depende de la plataforma:

- Linux El directorio *MQDataRoot*`/conv/table` en todas las versiones de Linux.
- Windows El directorio *MQDataRoot*`\conv\table` en Windows.

Adición o modificación de entradas de CCSID

Una entrada del archivo `ccsid_part2.tbl` tiene el formato siguiente:

```
<CCSID number> <Base CCSID> <DBCS CodePage> <SBCS CodePage>
<Type> <Encoding> <ACRI> <Name>
```

Una entrada de ejemplo para el CCSID 1200 (UTF-16) es:

```
  1200 1200 1200 1200 3 8 0 UTF-16
```

Nota: Para obtener más detalles sobre el valor para ACRI, consulte el comentario del archivo `ccsid_part2.tbl`.

En el formato de `ccsid_part2.tbl`:

El tipo puede ser igual a:
1=SBCS

2=DBCS

3=MBCS

El cifrado puede ser igual a:
1=EBCDIC

2 = ASCII

3 = ISO

4 = UCS-2

5 = UTF-8

6 = Euc

7 = GB18030

8 = UTF-16

9 = UTF-32

Al editar el archivo:

- Puede especificar un comentario utilizando el símbolo # al comienzo de una línea. Esto evita que IBM MQ intente analizar la línea.
- No puede proporcionar comentarios en la misma línea.
- Debe asegurarse de que no crea líneas en blanco.
- No debe añadir nuevas entrada al final del archivo.

  Las nuevas entradas de CCSID deben añadirse antes de la información de tabla de ACRI.

Especificar la conversión de datos predeterminada

Puede definir los CCSID de conversión predeterminados, que se utilizan para convertir entre CCSID ASCII o similares y EBCDIC, si no se admite ninguna conversión entre dos CCSID.

Si se habilita esta función, se usa la conversión predeterminada en las cabeceras de mensajes y transmisión, y también se puede utilizar en la conversión de datos de usuario.

Las conversiones predeterminadas se habilitan mediante la creación de dos líneas similares a las siguientes:

```
default       0      500     1     1    0
default       0      850     1     2    0
```

La primera línea establece el valor predeterminado para CCSID EBCDIC en 500 y la segunda línea establece el valor predeterminado para CCSID ASCII y similares en 850.

Especificación de datos para distintos niveles de mandatos

Para especificar entradas de CCSID para distintos niveles de mandatos de IBM MQ, utilice un símbolo de dos puntos seguido por el nivel de mandatos (o niveles de mandatos) de IBM MQ al que desee que se aplique la siguiente sección.

El número representa el nivel de mandatos mínimo en el que debe ejecutarse el gestor de colas o el cliente. Por ejemplo, si el gestor de colas actual está en el nivel de mandatos 900, y detecta un distintivo de nivel de mandatos 800 o 900, se leen los CCSID.

No obstante, un gestor de colas a nivel 800 pasará por alto los CCSID de la sección 900.

El nivel de mandatos especificado se aplica a todas las entradas CCSID encontradas tras un distintivo de nivel de mandatos, hasta que se encuentre un nuevo distintivo de nivel de mandatos.

Si necesita establecer el nivel de mandatos en todos los niveles de mandatos, especifique el número cero.

Al analizar `ccsid_part2.tbl` por primera vez, IBM MQ trata todos los CCSID encontrados como válidos para todos los niveles de mandatos de IBM MQ.

Se empiezan a utilizar versiones únicamente cuando IBM MQ encuentra el primer distintivo de nivel de mandatos.

El fragmento de código siguiente muestra un ejemplo que utiliza versiones:

```
# Comment Block
# End of Comment Block
# Because no command level flag is specified and we're at the start of the file
# the following CCSIDs will be read on all versions
    819   819       0      819     1    3     0    ISO8859-1
    923   923       0      923     1    3     0    ISO8859-15
   1051  1051       0     1051     1    3     0    IBM-1051
# The colon :900 below shows that the CCSIDs after will only be for MQ cmd level 900 and above
:900
   8629   437       0      437     1    2     0    IBM-437
  12725   437       0      437     1    2     0    IBM-437
  16821   437       0      437     1    2     0    IBM-437
  20917   437       0      437     1    2     0    IBM-437
# The colon :0 below shows that the CCSIDs after will be for all version of MQ
:0
   4946   850       0      850     1    2     0    IBM-850
  33618   850       0      850     1    2     0    IBM-850
  61697   850       0      850     1    2     0    IBM-850
  61698   850       0      850     1    2     0    IBM-850
```

Administración de Managed File Transfer

Utilice los mandatos de Managed File Transfer para administrar Managed File Transfer. También puede utilizar IBM MQ Explorer para realizar algunas de las tareas administrativas.

Inicie la transferencia colocando un mensaje en una cola de mandatos de agente

También puede iniciar una transferencia de archivos colocando un mensaje de transferencia de archivos en la cola de mandatos del agente de origen. Un nombre de cola de mandatos de ejemplo es `SYSTEM.FTE.COMMAND.AGENT01`. Debe asegurarse de que el mensaje llegue a la cola de mandatos del agente de origen correcto; si el mensaje lo recibe un agente que no coincide con la información de origen del XML, el mensaje es rechazado.

La solicitud de transferencia XML debe ajustarse al esquema `FileTransfer.xsd` y utilizar el elemento <request> como elemento raíz. Para obtener información sobre la estructura y el contenido de un mensaje de petición de transferencia, consulte Formato de mensaje de petición de transferencia de archivos. El modo de colocar el mensaje de solicitud de transferencia en la cola de mandatos de un agente depende de la tarea. Por ejemplo, puede utilizar la API IBM MQ Java para colocar un mensaje en la cola de forma programática.

Conceptos relacionados

"Transferencia de datos de archivos a mensajes" en la página 269
Puede utilizar la característica de archivo a mensaje de Managed File Transfer para transferir datos desde un archivo a un único mensaje o varios mensajes, en una cola de IBM MQ.

"Transferir datos de mensajes a archivos" en la página 285
La función de mensaje a archivo de Managed File Transfer le permite transferir datos de uno o más mensajes en una cola de IBM MQ a un archivo , un conjunto de datos (en z/OS) o un espacio de archivos de usuario. Si tiene una aplicación que crea o procesa mensajes de IBM MQ, puede utilizar la función de mensaje a archivo de Managed File Transfer para transferir estos mensajes a un archivo en cualquier sistema de la red de Managed File Transfer.

"El puente de protocolo" en la página 296
El puente de protocolo permite que la red de Managed File Transfer (MFT) acceda a los archivos almacenados en un servidor de archivos fuera de la red de MFT, bien en su dominio local o bien en una ubicación remota. Este servidor de archivos puede utilizar los protocolos de red FTP, FTPS o SFTP. Cada servidor de archivos necesita al menos un agente dedicado. El agente dedicado se conoce como el agente de puente de protocolo. Un agente de puente puede interactuar con varios servidores de archivos.

"Cómo trabajar con MFT desde IBM Integration Bus" en la página 336
Puede trabajar con Managed File Transfer desde IBM Integration Bus utilizando los nodos FTEOutput y FTEInput.

"Recuperación y reinicio de MFT" en la página 337
Si el agente o el gestor de colas están no disponibles por algún motivo, por ejemplo, debido a un fallo en el suministro o en la red, Managed File Transfer se recupera tal como se indica a continuación en los siguientes escenarios:

Tareas relacionadas

"Iniciar un agente de MFT" en la página 214
Para poder utilizar un agente de Managed File Transfer para una transferencia de archivos, antes debe iniciar el agente.

"Inicio de una nueva transferencia de archivos" en la página 221
Puede iniciar una nueva de transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos y puede elegir transferir un único archivo o varios archivos de un grupo.

"Supervisión de transferencias de archivos que están en curso" en la página 228
Puede supervisar una transferencia de archivos que está en curso utilizando la pestaña **Transferencia de archivos gestionada - Progreso de la transferencia actual** en IBM MQ Explorer. Esta transferencia de archivos se puede iniciar desde IBM MQ Explorer o desde la línea de mandatos. la pestaña también muestra el progreso de las transferencias planificadas en el punto de inicio de las transferencias planificadas.

"Visualización del estado de transferencias de archivos en el Registro de transferencias" en la página 231
Puede ver los detalles de las transferencias de archivos utilizando el **Registro de transferencias** en IBM MQ Explorer. Puede tratarse de transferencias iniciadas desde la línea de mandatos o desde IBM MQ Explorer. También puede personalizar lo que aparece en el **Registro de transferencias**.

"Supervisión de recursos de MFT" en la página 233
Puede supervisar recursos de Managed File Transfer; por ejemplo, una cola o un directorio. Cuando se cumple una condición en este recurso, el supervisor de recursos inicia una tarea, por ejemplo una transferencia de archivos. Puede crear un supervisor de recursos utilizando el mandato **`fteCreateMonitor`** o la vista **Supervisores** en el plug-in de Managed File Transfer para IBM MQ Explorer.

"Cómo trabajar con plantillas de transferencia de archivos" en la página 266

Las plantillas de transferencia de archivos se pueden utilizar para almacenar valores de transferencia de archivos comunes para transferencias repetitivas y complejas. Cree una plantilla de transferencia desde la línea de mandatos utilizando el mandato **fteCreateTemplate** o utilice IBM MQ Explorer para crear una plantilla de transferencia mediante el asistente **Crear nueva plantilla para transferencia de archivos** o guarde una plantilla mientras crea una transferencia de archivos marcando el recuadro de selección **Guardar valores de transferencia como plantilla**. La ventana **Plantillas de transferencia** muestra todas las plantillas de transferencia que ha creado en la red de Managed File Transfer.

"Listar agentes de MFT" en la página 219
Puede listar los agentes registrados Managed File Transfer con un gestor de colas específico mediante la línea de mandatos o bien mediante IBM MQ Explorer.

"Detener un agente de MFT" en la página 220
Puede detener un agente Managed File Transfer desde la línea de mandatos. Cuando detiene un agente, desactiva el agente y permite que el agente complete la transferencia de archivos actual antes de que se detenga. También puede especificar el parámetro **-i** en la línea de mandatos para detener inmediatamente un agente. Cuando el agente se ha detenido, no puede utilizar ese agente para transferir archivos hasta que lo reinicie.

Configuración de un registrador de MFT

Referencia relacionada

Directrices para transferir archivos

Iniciar un agente de MFT

Para poder utilizar un agente de Managed File Transfer para una transferencia de archivos, antes debe iniciar el agente.

Acerca de esta tarea

Puede iniciar un Managed File Transfer Agent desde la línea de mandatos. En este cso, el proceso del agente se detiene cuando finaliza sesión en el sistema.

ALW En AIX, Linux, and Windows, puede configurar un agente para que se continúe ejecutando cuando cierre sesión del sistema y pueda continuar recibiendo las transferencias de archivos.

z/OS En z/OS, puede configurar el agente para que se inicie como una tarea inicia desde JCL, sin necesidad de una sesión interactiva.

Tenga en cuenta que si un agente encuentra un error irrecuperable cuando se está ejecutando, se genera una primera FDC (Failure Data Capture) y se detiene el agente.

Procedimiento

- Para iniciar un agente desde la línea de mandatos, utilice el mandato **fteStartAgent**.
  Para obtener más información, consulte **fteStartAgent**.
- ALW
  Para configurar un agente, de modo que se continúe ejecutando cuando finaliza sesión en el sistema:
  - Windows En Windows, configure el agente para que se ejecute como un servicio de Windows. Para obtener más información, consulte "Inicio de un agente MFT como un servicio de Windows" en la página 215.
  - Linux AIX En AIX and Linux, configure el agente para que se inicie automáticamente durante un rearranque utilizando un archivo de script. Para obtener más información, consulte "Inicio de un agente de MFT en el arranque del sistema AIX and Linux" en la página 217.
- z/OS

En z/OS, configure el agente para que se inicie como una tarea iniciada desde JCL sin necesidad de una sesión interactiva.

Para obtener más información, consulte “Starting an MFT agent on z/OS” en la página 219.

Tareas relacionadas

“Listar agentes de MFT” en la página 219
Puede listar los agentes registrados Managed File Transfer con un gestor de colas específico mediante la línea de mandatos o bien mediante IBM MQ Explorer.

“Detener un agente de MFT” en la página 220
Puede detener un agente Managed File Transfer desde la línea de mandatos. Cuando detiene un agente, desactiva el agente y permite que el agente complete la transferencia de archivos actual antes de que se detenga. También puede especificar el parámetro **-i** en la línea de mandatos para detener inmediatamente un agente. Cuando el agente se ha detenido, no puede utilizar ese agente para transferir archivos hasta que lo reinicie.

Referencia relacionada

MFT Valores de estado de agente
fteStartAgent

Windows Inicio de un agente MFT como un servicio de Windows

Puede iniciar un agente como un servicio de Windows, de modo que cuando finalice la sesión de Windows, el agente continúa ejecutándose y puede recibir transferencias de archivos.

Acerca de esta tarea

En Windows, cuando inicia un agente desde la línea de mandatos, el proceso de agente se ejecuta utilizando el nombre de usuario que ha utilizado para iniciar la sesión en Windows. Cuando finaliza la sesión del sistema, el proceso del agente se detiene. Para impedir que el agente se detenga, puede configurar un agente para que se ejecute como un servicio de Windows. La ejecución como un servicio de Windows también permite configurar agentes para que se inicien automáticamente cuando se inicia o se reinicia el entorno de Windows.

Realice los pasos siguientes para iniciar un agente que se ejecute como un servicio de Windows. Debe estar ejecutando Managed File Transfer en una de las versiones de Windows soportadas para ejecutar el agente como un servicio de Windows. Para ver la lista de entornos soportados, consulte los Requisitos del sistema para IBM MQ.

Los pasos exactos dependen de si ya ha creado un agente o de si lo está creando. Ambas opciones se describen en los pasos siguientes.

Procedimiento

1. Si está creando un agente de Managed File Transfer , utilice el mandato **fteCreateAgent**, **fteCreateCDAgent**o **fteCreateBridgeAgent** . Especifique el parámetro **-s** para ejecutar el agente como un servicio de Windows. En el ejemplo siguiente, se crea el agente AGENT1, que tiene un gestor de colas del agente QMGR1. El servicio de Windows se ejecuta con el nombre de usuario `fteuser`, que tiene asociada la contraseña `ftepassword`.

   ```
   fteCreateAgent -agentName AGENT1 -agentQMgr QMGR1 -s -su fteuser -sp ftepassword
   ```

   Puede especificar opcionalmente un nombre para el servicio después del parámetro **-s**. Si no especifica un nombre, el servicio se denomina `mqmftAgent`*AGENTQMGR*, donde *AGENT* es el nombre de agente que ha especificado y *QMGR* es el nombre del gestor de colas de agente. En este ejemplo, el nombre predeterminado para el servicio es `mqmftAgentAGENT1QMGR1`.

   Nota: La cuenta de usuario Windows que especifique utilizando el parámetro **-su** debe tener los derechos **Log on as a service** . Para obtener información sobre cómo configurarlo, consulte

Resolución de problemas de un agente o registrador de MFT que se ejecuta como un servicio de Windows.

Para obtener más información, consulte fteCreateAgent, fteCreateCDAgent: crear un Connect:Direct agente de puenteo fteCreateBridgeAgent (crear y configurar un agente de puente de protocolo MFT).

2. Si ha seguido el paso anterior para crear un agente, ejecute los mandatos MQSC generados por el mandato **fteCreateAgent**, **fteCreateCDAgent**, o **fteCreateBridgeAgent** . Estos mandatos crean las colas de IBM MQ que el agente necesita.

   Por ejemplo, para un agente denominado *AGENT1*, un gestor de colas del agente denominado *QMGR1* y un gestor de colas de coordinación denominado *COORDQMGR1*, ejecute el mandato siguiente:

   ```
   runmqsc QMGR1 MQ_DATA_PATH\mqft\config\COORDQMGR1\agents\AGENT1\AGENT1_create.mqsc
   ```

3. Si no ha seguido los pasos anteriores para crear un agente y en su lugar desea configurar un agente existente para que se ejecute como servicio de Windows, detenga primero el agente si está en ejecución y, a continuación, modifique la configuración.

   a) El ejemplo siguiente utiliza un agente denominado AGENT1. Ejecute el siguiente mandato:

   ```
   fteStopAgent AGENT1
   ```

   b) Utilice el mandato **fteModifyAgent** para configurar el agente para que se ejecute como un servicio de Windows:

   ```
   fteModifyAgent -agentName AGENT1 -s -su fteuser -sp ftepassword
   ```

   Si desea más información, consulte fteModifyAgent: ejecutar un agente MFT como un servicio Windows.

4. Inicie el agente utilizando el mandato **fteStartAgent**. De forma alternativa, puede utilizar la herramienta de Servicios de Windows, que está disponible en las herramientas administrativas en el Panel de control, que se selecciona desde el menú de inicio del escritorio de Windows para iniciar el servicio.

   ```
   fteStartAgent AGENT1
   ```

   El servicio continúa ejecutándose aunque haya finalizado la sesión en Windows. Para garantizar que el servicio también se reinicia cuando se reinicia Windows después de una conclusión, el campo **Tipo de inicio** en la herramienta Servicios de Windows se establece en **Automático** de forma predeterminada. Cámbielo por **Manual** si no desea que el servicio se reinicie cuando se reinicie Windows.

5. Opcional: Para detener el agente, utilice el mandato fteStopAgent o utilice la herramienta Windows Services. Por ejemplo, desde la línea de mandatos, ejecute el mandato siguiente:

   ```
   fteStopAgent AGENT1
   ```

   - Cuando se ejecuta el mandato **fteStopAgent** como un servicio, el mandato siempre se ejecuta utilizando el parámetro **-i**, independientemente de si se ha especificado este parámetro. El parámetro **-i** detiene el agente inmediatamente sin completar las transferencias que están en curso. Esto se debe a una limitación del servicio de Windows.

Qué hacer a continuación

Si tiene problemas al iniciar el servicio Windows , consulte Resolución de problemas de un agente o registrador de MFT que se ejecuta como un servicio de Windows. En este tema también se describe la ubicación de los archivos de registro de servicio de Windows.

Tareas relacionadas

“Listar agentes de MFT” en la página 219

Puede listar los agentes registrados Managed File Transfer con un gestor de colas específico mediante la línea de mandatos o bien mediante IBM MQ Explorer.

"Detener un agente de MFT" en la página 220
Puede detener un agente Managed File Transfer desde la línea de mandatos. Cuando detiene un agente, desactiva el agente y permite que el agente complete la transferencia de archivos actual antes de que se detenga. También puede especificar el parámetro **-i** en la línea de mandatos para detener inmediatamente un agente. Cuando el agente se ha detenido, no puede utilizar ese agente para transferir archivos hasta que lo reinicie.

Referencia relacionada

fteCreateAgent (crea un agente de MFT)

fteCreateCDAgent (crear un agente de puente Connect:Direct)

fteCreateBridgeAgent (crear y configurar un agente de puente de protocolo de MFT)

fteModifyAgent (ejecutar un agente de MFT como un servicio de Windows)

Información relacionada

El archivo MFT `agent.properties`

Linux AIX Inicio de un agente de MFT en el arranque del sistema AIX and Linux

Un Managed File Transfer Agent puede configurarse para que se inicie durante el arranque del sistema en AIX and Linux. Cuando finaliza la sesión, el agente continúa en ejecución y puede recibir transferencias de archivos.

Cuando haya creado y configurado un agente utilizando uno de estos mandatos Managed File Transfer ; **fteCreateAgent**, **fteCreateCDAgent**o **fteCreateBridgeAgent**, puede configurarlo para que se inicie automáticamente durante un rearranque en máquinas AIX and Linux utilizando un archivo de script que simplemente ejecute el mandato siguiente:

```
su -l mqmft_user -c mq_install_root/bin/fteStartAgent agent_name
```

Donde *mq_install_root* es el directorio raíz de la instalación de Managed File Transfer necesaria, el valor predeterminado es: `/opt/mqm` y *nombre_agente* es el nombre del Managed File Transfer Agent que se va a iniciar. El uso de este archivo de script varía en función del sistema operativo específico. Por ejemplo, hay opciones adicionales disponibles en Linux.

Linux

Linux

Para los sistemas Linux hay varias formas de iniciar aplicaciones durante el proceso de arranque del sistema. En general, considere realizar los pasos siguientes:

1. Cree un archivo denominado `/etc/rc.mqmft` con contenido:

   ```
   #!/bin/sh
   su -l mqmft_user"-c mq_install_root/bin/fteStartAgent agent_name"
   ```

   Donde *mqmft_user* es el ID de usuario bajo el que se va a ejecutar el proceso de agente. Este ID de usuario debe ser miembro del grupo mqm.
2. Convierta el archivo en ejecutable, por ejemplo:

   ```
   chmod 755 /etc/rc.mqmft
   ```

3. A continuación, añada la siguiente línea a /etc/inittab:

```
mqmft:5:boot:/etc/rc.mqmft
```

Otras formas de iniciar un agente durante el arranque en Linux incluye añadir las líneas de script al archivo /etc/rc.d/rc.local o en Linux SuSe, añadiendo las líneas de script al archivo /etc/init.d/boot.local . Debe seleccionar el método que funcione mejor en su entorno. A continuación se muestra más información sobre otras maneras de iniciar un agente durante el arranque en distribuciones de Linux específicas que están soportadas:

SLES 10 y 11

En los sistemas SUSE Linux Enterprise Server (SLES) 10 y 11, siga estos pasos:

1. Como ID de usuario root del sistema, cree su propio archivo /etc/init.d/rc.rclocal.
2. Añada las líneas siguientes al archivo rc.rclocal:

```
#!/bin/sh
### BEGIN INIT INFO
# Provides: rc.rclocal
# Required-Start: $network $syslog
# Required-Stop: $network $syslog
# Default-Stop: 0 1 2 6
# Description: MQMFT agent startup
### END INIT INFO
su -l mqmft_user"-c mq_install_root/bin/fteStartAgent agent_name"
```

3. Ejecute los mandatos siguientes:

```
chmod 755 rc.rclocal

chkconfig --add rc.rclocal
```

Inicio de agentes de Managed File Transfer en Linux con systemd

Linux

Lleve a cabo el procedimiento siguiente:

1. Cree un archivo en la carpeta del sistema /etc/systemd/ y denomine, por ejemplo, <agentname>.service. Añada el siguiente contenido, donde <agentname> es MFT_AGT_LNX_0.

```
# vi /etc/systemd/system/MFT_AGT_LNX_0.service
[Unit]
Description=IBM MQ MFT MFT_AGT_LNX_0
[Service]
ExecStart=/opt/mqm/bin/fteStartAgent MFT_AGT_LNX_0
ExecStop=/opt/mqm/bin/fteStopAgent MFT_AGT_LNX_0
Type=forking
User=mqm
Group=mqm
KillMode=none
```

2. Para habilitar el servicio, ejecute los mandatos siguientes:

```
# systemctl enable MFT_AGT_LNX_0
# systemctl daemon-reload
```

3. Para iniciar el agente y comprobar su estado, ejecute los mandatos siguientes:

```
# systemctl start MFT_AGT_LNX_0
# systemctl status MFT_AGT_LNX_0
```

Tareas relacionadas

"Detener un agente de MFT" en la página 220
Puede detener un agente Managed File Transfer desde la línea de mandatos. Cuando detiene un agente, desactiva el agente y permite que el agente complete la transferencia de archivos actual antes de que se detenga. También puede especificar el parámetro **-i** en la línea de mandatos para detener inmediatamente un agente. Cuando el agente se ha detenido, no puede utilizar ese agente para transferir archivos hasta que lo reinicie.

Referencia relacionada

fteCreateAgent

fteCreateCDAgent: crear un agente de puente Connect:Direct

fteCreateBridgeAgent (crear y configurar un agente de puente de protocolo de MFT)

z/OS Starting an MFT agent on z/OS

On z/OS, in addition to running the **fteStartAgent** command from a z/OS UNIX System Services session, you can start an agent as a started task from JCL without the need for an interactive session.

A started task is used because it runs under a specific user ID and is not affected by users logging off.

Note: Started tasks are typically run under an administrative user that might not have log-on privileges and so it is not possible to log on to the z/OS system as the user that the agent is running under. The **fteStartAgent**, **fteStopAgent**, **fteSetAgentTraceLevel** commands, and the **fteShowAgentDetails** command with the **-d** parameter specified, cannot be issued for that agent.

You can use the agent property **adminGroup** with Managed File Transfer agents on z/OS. You can define a security manager group, for example MFTADMIN and then add the started task userid and administrator TSO ids to this group. Edit the agent properties file and set the **adminGroup** property to be the name of this security manager group.

```
adminGroup=MFTADMIN
```

Members of this group can then issue the **fteStartAgent**, **fteStopAgent**, and **fteSetAgentTraceLevel** commands, and the **fteShowAgentDetails** command with the **-d** parameter specified, for the agent that is running as a started task.

For more information, see the **adminGroup** property in The MFT `agent.properties` file.

As a Java application, an agent is a z/OS UNIX System Services application that you can run from JCL by using the BFGAGSTP member, from a generated Managed File Transfer command PDSE library data set for an agent. For more information about how to create an MFT command PDSE library data set, and customize it for the required agent, see Creating an MFT Agent or Logger command data set.

Related concepts

Enabling MFT agents to connect to remote z/OS queue managers

Related reference

"Stopping an MFT agent on z/OS" on page 220
If you are running a Managed File Transfer Agent on z/OS as a started task from JCL, the agent accepts the z/OS operator commands **MODIFY** and **STOP**, in addition to the **fteStopAgent** command.

The MFT `agent.properties` file

Listar agentes de MFT

Puede listar los agentes registrados Managed File Transfer con un gestor de colas específico mediante la línea de mandatos o bien mediante IBM MQ Explorer.

Acerca de esta tarea

Para listar agentes mediante la línea de mandatos, consulte el mandato fteListAgents.

Para listar agentes utilizando IBM MQ Explorer, en la vista de navegador pulse **Agentes** bajo el nombre del gestor de colas de coordinación.

Si un agente no aparece en la lista del mandato **fteListAgents** o no se muestra en IBM MQ Explorer, utilice el diagrama de flujo de diagnósticos en el tema siguiente para localizar y arreglar el problema: ¿Qué hacer si el agente MFT no está listado por el mandato **fteListAgents**?.

Referencia relacionada

FteListAgents: listar los agentes de MFT para un gestor de colas de coordinación

Valores de estado de agente MFT

fteShowAgentDetails

Detener un agente de MFT

Puede detener un agente Managed File Transfer desde la línea de mandatos. Cuando detiene un agente, desactiva el agente y permite que el agente complete la transferencia de archivos actual antes de que se detenga. También puede especificar el parámetro **-i** en la línea de mandatos para detener inmediatamente un agente. Cuando el agente se ha detenido, no puede utilizar ese agente para transferir archivos hasta que lo reinicie.

Antes de empezar

Si desea comprobar los nombres de los agentes asociados con un gestor de colas, puede listar los agentes utilizando IBM MQ Explorer o la línea de mandatos; consulte el mandato fteListAgents.

Acerca de esta tarea

Para detener un agente desde la línea de mandatos, consulte fteStopAgent.

Si un agente se detiene de forma controlada utilizando **fteStopAgent**, el agente no acepta ninguna nueva solicitud de transferencia gestionada y espera a que se completen las transferencias en curso antes de que se concluya realmente. A partir de IBM MQ 9.3.0, para mostrar que el agente sigue en un estado transitorio y, por lo tanto, todavía no se ha cerrado y no se puede reiniciar, el agente entra en el estado DETENIÉNDOSE hasta que se completen las transferencias en curso. Este estado aparece en la salida de los mandatos **fteListAgents** y **fteShowAgentDetails** , y en las consultas deMFT REST API, y en la vista **Agentes** del plugin MFT de IBM MQ Explorer.

Windows Si ha configurado su agente para que se ejecute como servicio de Windows, ejecutando el mandato **fteStopAgent** también se detiene el servicio de Windows. O bien, puede detener el agente deteniendo el servicio mediante la herramienta Servicios de Windows. Para obtener más información, consulte el tema “Inicio de un agente MFT como un servicio de Windows” en la página 215.

Tareas relacionadas

“Iniciar un agente de MFT” en la página 214
Para poder utilizar un agente de Managed File Transfer para una transferencia de archivos, antes debe iniciar el agente.

Referencia relacionada

MFT Valores de estado de agente

fteStopAgent

“Stopping an MFT agent on z/OS” en la página 220
If you are running a Managed File Transfer Agent on z/OS as a started task from JCL, the agent accepts the z/OS operator commands **MODIFY** and **STOP**, in addition to the **fteStopAgent** command.

z/OS Stopping an MFT agent on z/OS

If you are running a Managed File Transfer Agent on z/OS as a started task from JCL, the agent accepts the z/OS operator commands **MODIFY** and **STOP**, in addition to the **fteStopAgent** command.

A started task is used because it runs under a specific user ID and is not affected by users logging off.

Note: Started tasks are typically run under an administrative user that might not have log-on privileges and so it is not possible to log on to the z/OS system as the user that the agent is running under. The **fteStartAgent**, **fteStopAgent**, **fteSetAgentTraceLevel** commands, and the **fteShowAgentDetails** command with the **-d** parameter specified, cannot be issued for that agent.

You can use the agent property **adminGroup** with Managed File Transfer agents on z/OS. You can define a security manager group, for example MFTADMIN and then add the started task userid and administrator TSO ids to this group. Edit the agent properties file and set the **adminGroup** property to be the name of this security manager group.

```
adminGroup=MFTADMIN
```

Members of this group can then issue the **fteStartAgent**, **fteStopAgent**, and **fteSetAgentTraceLevel** commands, and the **fteShowAgentDetails** command with the **-d** parameter specified, for the agent that is running as a started task.

For more information, see the **adminGroup** property in The MFT `agent.properties` file .

Controlled agent shutdown by using the z/OS MODIFY command (F)

The **MODIFY** command allows you to stop an agent in a controlled way as an alternative to the **fteStopAgent** command. The agent completes any transfers currently in progress but the agent does not start any new transfers.

For example:

```
F job_name,APPL=STOP
```

where *job_name* is the job that the agent process is running under.

Immediate agent shutdown by using the z/OS STOP command (P)

The **STOP** command is equivalent to an immediate stop by using the **fteStopAgent** command with the **-i** parameter. The agent is stopped immediately even if the agent is currently transferring a file.

For example:

```
P job_name
```

where *job_name* is the job that the agent process is running under.

Related concepts

Enabling MFT agents to connect to remote z/OS queue managers

Related reference

“Starting an MFT agent on z/OS” on page 219

On z/OS, in addition to running the **fteStartAgent** command from a z/OS UNIX System Services session, you can start an agent as a started task from JCL without the need for an interactive session.

The MFT `agent.properties` file

Inicio de una nueva transferencia de archivos

Puede iniciar una nueva de transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos y puede elegir transferir un único archivo o varios archivos de un grupo.

Acerca de esta tarea

Para iniciar una nueva transferencia de archivos desde la línea de mandatos, consulte el mandato fteCreateTransfer.

Para iniciar una nueva transferencia de archivos utilizando el asistente **Crear nueva transferencia de archivos gestionada** en IBM MQ Explorer, siga los pasos que se indican a continuación:

Procedimiento

1. En la vista Navegador, pulse **Transferencia de archivos gestionada**. En la vista Contenido, aparece **Central de transferencias de archivos gestionadas**.
2. En la vista Navegador, aparecen todos los gestores de colas de coordinación. Expanda el nombre del gestor de colas de coordinación en el que está registrado el agente que desea utilizar para la transferencia. Si está conectado actualmente a un gestor de colas de coordinación que no sea el que desea utilizar para la transferencia, pulse el botón derecho del ratón en dicho nombre de gestor de colas de coordinación en la vista Navegador y pulse **Desconectar**. A continuación, pulse el botón derecho del ratón en el nombre del gestor de colas de coordinación que desea utilizar y pulse **Conectar**.
3. Inicie el asistente **Crear nueva transferencia de archivos gestionada** utilizando cualquiera de los dos métodos siguientes:
   a) Pulse el botón derecho del ratón en el nombre de cualquiera de los nodos siguientes en la vista Navegador: el gestor de colas de coordinación relevante, **Plantillas de transferencia**, **Registro de transferencias** o **Transferencias pendientes**. A continuación, pulse **Nueva transferencia** para iniciar el asistente.
   b) Pulse **Archivo** > **Nuevo** > **Otros** > **Asistentes de transferencia de archivos gestionada** > **Asistente de nueva transferencia**
4. Siga las instrucciones de los paneles del asistente. También se proporciona ayuda según contexto para cada panel. Para acceder a la ayuda según contexto en Windows, pulse F1. En Linux, pulse `Ctrl+F1` o `Shift+F1`.

Conceptos relacionados

“Utilización de archivos de definición de transferencia” en la página 223
Puede especificar un archivo de definición de transferencia para crear una transferencia de archivos. El archivo de definición de transferencia es un archivo XML que define parte o toda la información requerida para crear la transferencia.

Tareas relacionadas

“Creación de una transferencia planificada de archivos” en la página 225
Puede planificar una nueva transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos. La transferencia planificada puede contener archivos individuales o varios archivos en un grupo. Puede realizar una transferencia de archivos planificada una vez o repetirla varias veces.

“Desencadenamiento de una transferencia de archivos” en la página 227
Puede establecer determinadas condiciones desencadenantes en una transferencia de archivos que se deben cumplir para que se pueda producir la transferencia. Si las condiciones desencadenantes no se cumplen, la transferencia de archivos no se produce y se somete un mensaje de registro opcionalmente para registrar el hecho de que no se produjo la transferencia. A continuación, se descarta la petición de transferencia de archivos. Por ejemplo, puede configurar una transferencia de archivos que se produzca sólo si un archivo nombrado en el sistema donde está ubiicado el agente de origen sobrepasa un tamaño especificado, o si un determinado archivo nombrado existe en el sistema donde está ubicado el agente de origen. Puede configurar una transferencia de archivos desencadenada desde IBM MQ Explorer o desde la línea de mandatos.

“Establecimiento de un tiempo de espera para la recuperación de transferencias estancadas” en la página 337
Puede establecer un tiempo de espera de recuperación de transferencia para las transferencias de archivos estancadas que se aplique a todas las transferencias para un agente de origen. También puede establecer un tiempo de espera de recuperación de transferencia para una transferencia individual. Si establece una cantidad de tiempo específica, en segundos, durante la cual un agente de origen sigue intentando recuperar una transferencia de archivos estancada y la transferencia no se realiza correctamente cuando el agente alcanza el tiempo de espera, la transferencia falla.

Referencia relacionada

fteCreateTransfer: iniciar una nueva transferencia de archivos

Formato de mensaje de solicitud de transferencia de archivos

Directrices para transferir archivos

Utilización de archivos de definición de transferencia

Puede especificar un archivo de definición de transferencia para crear una transferencia de archivos. El archivo de definición de transferencia es un archivo XML que define parte o toda la información requerida para crear la transferencia.

Los archivos de definición de transferencia resultan útiles si desea especificar varios archivos de origen y varios archivos de destino en una sola operación de transferencia. Puede utilizar un archivo de definición de transferencia para someter una transferencia de archivos compleja. Puede reutilizar y compartir el archivo de definición de transferencia.

Puede utilizar dos formatos para un archivo de definición de transferencia, y aunque estos formatos varían ligeramente, ambos se ajustan al esquema de `FileTransfer.xsd`. Puede encontrar este esquema en el directorio `samples\schema` de la instalación de Managed File Transfer.

Los siguientes dos formatos de archivos de definición de transferencia están soportados:

- Una definición de los archivos de origen y destino de una transferencia. Esta definición utiliza un elemento **transferSpecifications** como raíz.
- Una definición de la transferencia completa, incluyendo archivos de origen y destino y los agentes de origen y destino. Esta definición utiliza un elemento **request** como raíz.
  - Se pueden generar archivos con este formato desde el mandato **fteCreateTransfer** utilizando el parámetro **-gt**.

El ejemplo siguiente muestra un formato de archivo de definición de transferencia que especifica únicamente los archivos de origen y destino para una transferencia:

```
<?xml version="1.0" encoding="UTF-8"?>
<transferSpecifications xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="FileTransfer.xsd">
  <item checksumMethod="MD5" mode="text">
    <source recursive="false" disposition="leave">
      <file>textTransferTest.txt</file>
    </source>
    <destination type="directory" exist="overwrite">
      <file>c:\targetfiles</file>
    </destination>
  </item>
</transferSpecifications>
```

Para enviar este formato de archivo de definición de transferencia deberá especificar los agentes de origen y destino en la línea de mandatos:

```
fteCreateTransfer -sa AGENT1 -sm agent1qm -da AGENT2 -dm agent2qm -td
  c:\definitions\example1.xml
```

El siguiente ejemplo es un formato de archivo de definición de transferencia que especifica toda la información requerida para una transferencia:

```
<?xml version="1.0" encoding="UTF-8"?>
<request version="3.00" xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="FileTransfer.xsd">
  <managedTransfer>
    <originator>
      <hostName>example.com.</hostName>
      <userID>fteuser</userID>
    </originator>
    <sourceAgent agent="AGENT1" QMgr="agent1qm"/>
    <destinationAgent agent="AGENT2" QMgr="agent2qm"/>
    <transferSet>
      <item mode="binary" checksumMethod="MD5">
```

```
        <source recursive="false" disposition="leave">
          <file>c:\sourcefiles\*.jpg</file>
        </source>
        <destination type="directory" exist="error">
          <file>/targetfiles/images</file>
        </destination>
      </item>
    </transferSet>
  </managedTransfer>
</request>
```

Puede generar un archivo con este formato utilizando el parámetro **-gt** en el mandato **fteCreateTransfer**. Cuando envía un archivo de definición de transferencia con este formato, no necesita especificar nada más en la línea de mandatos:

```
fteCreateTransfer -td c:\definitions\example2.xml
```

Puede alterar temporalmente la información del agente de origen y destino en la línea de mandatos pasando los parámetros normales además del archivo de definición de transferencia. Por ejemplo:

```
fteCreateTransfer -da AGENT9 -dm agent9qm -td c:\definitions\example2.xml
```

Este ejemplo utiliza las opciones de línea de mandatos para alterar temporalmente el agente de destino definido dentro del archivo de definición de transferencia con **AGENT9** y el gestor de colas de destino definido en el archivo de definición de transferencia como **agent9qm**.

Los dos formatos descritos pueden contener uno o más elementos <item>. Para obtener más información sobre el elemento <item> consulte Formato del mensaje de solicitud de transferencia de archivos. Cada uno de estos elementos de transferencia define un par de archivo de origen y destino con atributos adicionales para controlar el comportamiento de la transferencia. Por ejemplo, puede especificar el comportamiento siguiente:

- Si la transferencia utiliza una suma de comprobación
- Si la transferencia es de texto o binaria
- Si desea suprimir el archivo de origen después de que haya finalizado la transferencia
- Si desea sobrescribir el archivo de destino si el archivo existe

Una ventaja de utilizar archivos de definición de transferencia es que puede especificar opciones adicionales que no están disponibles desde la línea de mandatos. Por ejemplo, cuando esté llevando a cabo transferencias de mensaje a archivo, puede especificar el atributo groupId utilizando un archivo de definición de transferencia. Este atributo especifica el ID de grupo de IBM MQ de los mensajes que se leen de la cola. Otra ventaja de utilizar archivos de definición de transferencia es que puede especificar distintas opciones para cada par de archivos. Por ejemplo, puede especificar si se utiliza una suma de comprobación, o si el archivo se transfiere en modalidad de texto o binaria, individualmente para cada archivo. Si utiliza la línea de mandatos, se aplican las mismas opciones para cada archivo de una transferencia.

Por ejemplo:

```
<item checksumMethod="none" mode="binary">
  <source disposition="leave">
    <file>c:\sourcefiles\source1.doc</file>
  </source>
  <destination type="file" exist="error">
    <file>c:\destinationfiles\destination1.doc</file>
  </destination>
</item>

<item checksumMethod="MD5" mode="text">
  <source disposition="delete">
    <file>c:\sourcefiles\source2.txt</file>
  </source>
  <destination type="file" exist="overwrite">
    <file encoding="UTF8" EOL="CRLF">c:\destinationfiles\destination2.txt</file>
  </destination>
```

```
</item>

<item checksumMethod="none" mode="text">
  <source recursive="false" disposition="leave">
    <file>c:\originfiles\source3.txt</file>
  </source>
  <destination type="file" exist="overwrite">
    <file>c:\targetfiles\destination3.txt</file>
  </destination>
</item>
```

z/OS Se pueden usar elementos para transferir un archivo de un sistema distribuido a un sistema z/OS:

z/OS

```
<item checksumMethod="none" mode="text">
  <source recursive="false" disposition="leave">
    <file>textTransferTest.txt</file>
  </source>
  <destination type="dataset" exist="overwrite">
    <file encoding="IBM-1047">//TEXT.TRANS.TEST</file>
  </destination>
</item>
```

z/OS Este ejemplo transfiere el archivo `textTransferTest.txt` del agente de origen al conjunto de datos `//TEXT.TRANS.TEST` en el agente de destino en modalidad de texto. Esta transferencia convierte los datos de origen de la codificación predeterminada del agente de origen (no se especifica ningún atributo de codificación de origen) a la página de códigos: IBM-1047.

Creación de una transferencia planificada de archivos

Puede planificar una nueva transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos. La transferencia planificada puede contener archivos individuales o varios archivos en un grupo. Puede realizar una transferencia de archivos planificada una vez o repetirla varias veces.

Acerca de esta tarea

Puede configurar una planificación de transferencia de archivos para que se realice una vez o para que se realice en los siguientes intervalos:

- Cada minuto
- Cada hora
- Diariamente
- Semanalmente
- Mensualmente
- Anualmente

A continuación, puede especificar que las apariciones se detengan en los puntos siguientes:

- A una hora y fecha definidas
- Tras un número definido de apariciones

También puede especificar que las apariciones continúen para siempre.

Si una transferencia planificada se ejecuta a la misma hora cada día, utilice el atributo **adjustScheduleTimeForDaylightSaving** en el archivo de propiedades del agente para ajustar la hora a la que se produce la planificación cuando cambian los relojes. Consulte El archivo MFT `agent.properties` para obtener más información.

Para crear una nueva transferencia de archivos planificada utilizando la línea de mandatos, utilice los parámetros de planificación (**-tb**, **-ss**, **-oi**, **-of**, **-oc** y **-es**) para el mandato fteCreateTransfer.

Para crear una nueva transferencia de archivos planificada utilizando el asistente **Crear nueva transferencia de archivos gestionada** en IBM MQ Explorer, utilice los pasos siguientes:

Procedimiento

1. En la vista Navegador, pulse **Transferencia de archivos gestionada**. En la vista Contenido, aparece **Central de transferencias de archivos gestionadas**.
2. En la vista Navegador, aparecen todos los gestores de colas de coordinación. Expanda el nombre del gestor de colas de coordinación en el que está registrado el agente que desea utilizar para la transferencia. Si está conectado actualmente a un gestor de colas de coordinación que no sea el que desea utilizar para la transferencia, pulse el botón derecho del ratón en dicho nombre de gestor de colas de coordinación en la vista Navegador y pulse **Desconectar**. A continuación, pulse el botón derecho del ratón en el nombre del gestor de colas de coordinación que desea utilizar y pulse **Conectar**.
3. Inicie el asistente **Crear nueva transferencia de archivos gestionada** mediante uno de los métodos siguientes:
   a) Pulse el botón derecho del ratón en el nombre de cualquiera de los nodos siguientes en la vista Navegador: el gestor de colas de coordinación relevante, **Plantillas de transferencia**, **Registro de transferencias** o **Transferencias pendientes**. A continuación, pulse **Nueva transferencia** para iniciar el asistente.
   b) Pulse **Archivo** > **Nuevo** > **Otros** > **Asistentes de transferencia de archivos gestionada** > **Asistente de nueva transferencia**
4. Siga las instrucciones de los paneles del asistente. Asegúrese de que ha seleccionado el recuadro de selección **Habilitar transferencia de planificación** y de que escribe los detalles de planificación en la pestaña **Planificar**. Las transferencias de archivos planificadas empiezan al cabo de un minuto de la hora de inicio de planificación, si no existan problemas que puedan afectar a la transferencia. Por ejemplo, podrían surgir problemas con la red o el agente que impidan que se inicie la transferencia planificada. Para cada panel se proporciona ayuda según contexto. Para acceder a la ayuda según contexto en Windows, pulse F1. En Linux, pulse `Ctrl+F1` o `Shift+F1`.

Resultados

Si desea más información sobre los mensajes implicados en transferencias de archivos planificadas, consulte Formatos de mensaje de registro de transferencia de archivos planificada.

Cómo trabajar con transferencias de archivos pendientes

Puede ver las transferencias de archivos planificadas que están pendientes en IBM MQ Explorer. La ventana **Transferencias pendientes** muestra todas las transferencias pendientes registradas en el gestor de colas de coordinación al que está conectado actualmente.

Acerca de esta tarea

Para ver el estado de una transferencia de archivos planificada que aún no se ha iniciado, siga estos pasos:

Procedimiento

1. Expanda **Transferencia de archivos gestionada** en la vista Navegador. En la vista Contenido, aparece **Central de transferencias de archivos gestionadas**.
2. En la vista Navegador, aparecen todos los gestores de colas de coordinación. Expanda el nombre del gestor de colas de coordinación que ha utilizado para la transferencia planificada. Si desea cambiar el gestor de colas de coordinación al que está conectado, pulse el botón derecho del ratón en el nombre del gestor de colas de coordinación que desea utilizar en la vista Navegador y pulse **Conectar**.
3. Pulse **Transferencias pendientes**. En la vista Contenido, aparece **Transferencias pendientes**.

4. La ventana **Transferencias pendientes** muestra los siguientes detalles sobre las transferencias de archivos gestionadas:
   a) **Nombre** El número de transferencia de archivos planificada. Este número se asigna automáticamente.
   b) **Origen** El nombre del agente de origen.
   c) **Archivo de origen** El nombre del archivo que se transferirá al sistema host.
   d) **Destino** El nombre del agente de destino.
   e) **Archivo de destino** El nombre del archivo de destino después de que se haya transferido al sistema de destino.
   f) **Inicio planificado (huso horario seleccionado)** La fecha y la hora en que se ha planificado que se inicie la transferencia de archivos, expresada en el huso horario seleccionado del administrador. Para cambiar el huso horario visualizado, pulse **Ventana** > **Preferencias** > **IBM MQ Explorer** > **Managed File Transfer** y seleccione un huso horario alternativo en la lista **Huso horario:**. Pulse **Aceptar**.
   g) **Repetir cada** Si ha elegido repetir la transferencia planificada, el intervalo especificado durante el cual desea repetir la transferencia, expresado como número.
   h) **Tipo de repetición** Si ha elegido repetir la transferencia planificada, el tipo de intervalo de repetición que ha especificado para la transferencia de archivos. El tipo puede ser uno de los valores siguientes: `minutes`, `hours`, `days`, `weeks`, `months` o `years`.
   i) **Repetir** Si ha elegido repetir la transferencia planificada, los detalles de cuándo desea que se detenga la transferencia de archivos de repetición. Por ejemplo, una fecha o una hora específicas o después de un número específico de apariciones.

Resultados

Para renovar lo que se visualiza en la ventana **Transferencias pendientes**, pulse el botón Renovar  en la barra de herramientas de la vista Contenido.

Para cancelar una transferencia de archivos pendiente, pulse el botón derecho del ratón en la transferencia específica y pulse **Cancelar**. La cancelación completa de una transferencia descarta la solicitud de transferencia de archivos.

Desencadenamiento de una transferencia de archivos

Puede establecer determinadas condiciones desencadenantes en una transferencia de archivos que se deben cumplir para que se pueda producir la transferencia. Si las condiciones desencadenantes no se cumplen, la transferencia de archivos no se produce y se somete un mensaje de registro opcionalmente para registrar el hecho de que no se produjo la transferencia. A continuación, se descarta la petición de transferencia de archivos. Por ejemplo, puede configurar una transferencia de archivos que se produzca sólo si un archivo nombrado en el sistema donde está ubiicado el agente de origen sobrepasa un tamaño especificado, o si un determinado archivo nombrado existe en el sistema donde está ubicado el agente de origen. Puede configurar una transferencia de archivos desencadenada desde IBM MQ Explorer o desde la línea de mandatos.

Acerca de esta tarea

Puede supervisar un recurso continuamente para que se cumpla una condición desencadenante. Para obtener más información sobre la supervisión de recursos, consulte: “Supervisión de recursos de MFT” en la página 233.

Existen tres diferentes condiciones desencadenantes que se pueden establecer. Las condiciones son las siguientes:

- Si un determinado archivo existe en el mismo sistema que el agente de origen
- Si un determinado archivo no existe en el mismo sistema que el agente de origen

- Si un determinado archivo sobrepasa un determinado tamaño en el sistema donde está ubicado el agente de origen (el tamaño se puede expresar en bytes, KB, MB o GB). Estas unidades de medida utilizan la convención 2^{10}, por ejemplo, 1 KB equivale a 1024 bytes y 1 MB equivale a 1024 KB.

Los tipos de desencadenantes de la lista anterior se pueden combinar de dos maneras:

- Para una sola condición, puede especificar más de un archivo en el sistema donde está ubicado el agente de origen. Esto desencadena la transferencia si cualquiera de los archivos especificados cumple la condición (operador booleano OR).
- Puede especificar varias condiciones. Esto desencadena la transferencia únicamente si se cumplen todas las condiciones (operador booleano AND).

También puede combinar la transferencia desencadenada con una transferencia planificada. Para obtener más información, consulte Creación de una transferencia de archivos planificada. En este caso, las condiciones desencadenantes se evalúan en el momento en el que va a iniciarse la planificación, o bien para una planificación repetitiva cada vez que va a iniciarse la planificación.

Las transferencias desencadenadas no están soportadas en los agentes de puente de protocolo.

Para crear una transferencia de archivos desencadenada utilizando la línea de mandatos, utilice el parámetro **-tr** en el mandato fteCreateTransfer.

Para crear una transferencia de archivos planificada utilizando el asistente **Crear nueva transferencia de archivos gestionada** en IBM MQ Explorer, utilice los pasos siguientes:

Procedimiento

1. En la vista Navegador, pulse **Transferencia de archivos gestionada**. En la vista Contenido, aparece **Central de transferencias de archivos gestionadas**.
2. En la vista Navegador, aparecen todos los gestores de colas de coordinación. Expanda el nombre del gestor de colas de coordinación que ha utilizado para la transferencia planificada. Si desea cambiar el gestor de colas de coordinación al que está conectado, pulse el botón derecho del ratón en el nombre del gestor de colas de coordinación que desea utilizar en la vista Navegador y pulse **Conectar**.
3. Inicie el asistente **Crear nueva transferencia de archivos gestionada** utilizando cualquiera de los dos métodos siguientes:
   a) Pulse el botón derecho del ratón en el nombre de cualquiera de los nodos siguientes en la vista Navegador: el gestor de colas de coordinación relevante, **Plantillas de transferencia**, **Registro de transferencias** o **Transferencias pendientes**. A continuación, pulse **Nueva transferencia** para abrir el asistente.
   b) Pulse **Archivo** > **Nuevo** > **Otros** > **Asistentes de transferencia de archivos gestionada** > **Asistente de nueva transferencia**
4. Siga las instrucciones de los paneles del asistente. Asegúrese de seleccionar el recuadro de selección **Habilitar transferencia desencadenada** en la pestaña **Desencadenantes** y de completar los campos de dicho separador para configurar el desencadenamiento. Para cada panel se proporciona ayuda según contexto. Para acceder a la ayuda según contexto en Windows, pulse F1. En Linux, pulse **Ctrl+F1** o Shift+F1.

Supervisión de transferencias de archivos que están en curso

Puede supervisar una transferencia de archivos que está en curso utilizando la pestaña **Transferencia de archivos gestionada - Progreso de la transferencia actual** en IBM MQ Explorer. Esta transferencia de archivos se puede iniciar desde IBM MQ Explorer o desde la línea de mandatos. la pestaña también muestra el progreso de las transferencias planificadas en el punto de inicio de las transferencias planificadas.

Acerca de esta tarea

Si desea utilizar IBM MQ Explorer para supervisar transferencias asociadas con un gestor de cola de coordinación en un sistema remoto, siga las instrucciones del tema “Configuración de IBM MQ Explorer para supervisar un gestor de colas de coordinación remota” en la página 230.

La información de transferencias de archivos anteriores no se mantiene después de detener y reiniciar IBM MQ Explorer. Al reiniciar el sistema, la información sobre las anteriores transferencias se borrará de la pestaña **Progreso de la transferencia actual**. Puede borrar transferencias completas utilizando **Eliminar transferencias completas** en cualquier momento en el que tenga abierto IBM MQ Explorer.

Procedimiento

Después de que haya iniciado una nueva transferencia de archivos utilizando IBM MQ Explorer o la línea de mandatos, podrá supervisar el progreso de la transferencia en la pestaña **Progreso de la transferencia actual**. Se visualiza la siguiente información para cada transferencia en curso:

a) **Origen**. El nombre del agente que se utiliza para transferir el archivo desde el sistema de origen.

b) **Destino**. El nombre del agente que se utiliza para recibir el archivo en el sistema de destino.

c) **Archivo actual**. El nombre del archivo que se está transfiriendo actualmente. La parte del archivo individual que se ha transferido se visualiza en B, KiB, MiB. GiB, o TiB junto con el tamaño total del archivo entre paréntesis. La unidad de medida visualizada depende del tamaño del archivo.

   B es bytes por segundos. KiB/s es kibibytes por segundo, donde 1 kibibyte equivale a 1024 bytes. MiB/s es mebibytes por segundo, donde 1 mebibyte equivale a 1 048 576 bytes. GiB/s es gibibytes por segundo, donde 1 gibibyte equivale a 1 073 741 824 bytes. TiB/s es tebibytes por segundo, donde 1 tebibyte equivale a 1 099 511 627 776 bytes.

d) **Número de archivos**. Si está transfiriendo más de un archivo, este número representa en qué punto del grupo total de archivos se encuentra la transferencia.

e) **Progreso**. La barra de progreso muestra el porcentaje completado de la transferencia de archivos actual. .

f) **Velocidad**. La velocidad a la que se transfiere el archivo en KiB/s (kibibytes por segundo, donde 1 kibibyte equivale a 1024 bytes).

g) **Iniciada (huso horario seleccionado)**. Hora a la que se ha iniciado la transferencia de archivos, presentada en el huso horario seleccionado del administrador. Para cambiar el huso horario visualizado, pulse **Ventana** > **Preferencias** > **IBM MQ Explorer** > **Managed File Transfer** y seleccione un huso horario alternativo en la lista **Huso horario:**. Pulse **Aceptar**.

   Si la transferencia entra en un estado de recuperación mientras se transfiere el archivo, la hora de inicio se actualiza para reflejar la hora en que se ha reanudado la transferencia de archivo.

Resultados

Este separador renueva automáticamente la información de forma periódica, pero para forzar una vista renovada de lo que se visualiza en la pestaña **Progreso de la transferencia actual**, pulse **Renovar** en la barra de herramientas de la vista Contenido.

Para suprimir transferencias de archivo de la pestaña **Progreso de la transferencia actual**, pulse **Eliminar transferencias completas** en la barra de herramientas de la vista Contenido. Al pulsar este botón, los detalles de la transferencia de archivos sólo se eliminan de la pestaña; no detiene ni cancela una planificación actual o planificada.

Si desea regresar al separador **Progreso de la transferencia actual** después de cerrarlo, puede visualizar la pestaña pulsando en **Ventana** > **Mostrar vista** > **Otros** > **Otros** > **Transferencia de archivos gestionada - Progreso de la transferencia actual**. Pulse **Aceptar**.

Qué hacer a continuación

Además, es posible desarrollar aplicaciones para la supervisión de transferencia de archivos personalizada. Esto puede lograrse mediante la creación de una suscripción al tema administrativo de Managed File Transfer adecuado (ya sea mediante programación o de forma administrativa) y la aplicación de supervisor, puede recibir publicaciones de actividad de transferencia de archivos de Managed File Transfer sobre el tema. Si desea más información sobre el tema de suscripción y el formato del mensaje de publicación, consulte Ejemplos de mensajes de progreso de transferencia de archivos.

Tareas relacionadas

"Configuración de IBM MQ Explorer para supervisar un gestor de colas de coordinación remota" en la página 230
Utilice IBM MQ Explorer para supervisar las transferencias de archivos asociadas con un gestor de colas de coordinación que se ejecuta en un sistema remoto. Necesita un sistema que sea capaz de ejecutar el IBM MQ Explorer. El componente IBM MQ Explorer debe estar instalado para poder conectarse al gestor de colas de coordinación remoto.

"Visualización del estado de transferencias de archivos en el Registro de transferencias" en la página 231
Puede ver los detalles de las transferencias de archivos utilizando el **Registro de transferencias** en IBM MQ Explorer. Puede tratarse de transferencias iniciadas desde la línea de mandatos o desde IBM MQ Explorer. También puede personalizar lo que aparece en el **Registro de transferencias**.

Configuración de IBM MQ Explorer para supervisar un gestor de colas de coordinación remota

Utilice IBM MQ Explorer para supervisar las transferencias de archivos asociadas con un gestor de colas de coordinación que se ejecuta en un sistema remoto. Necesita un sistema que sea capaz de ejecutar el IBM MQ Explorer. El componente IBM MQ Explorer debe estar instalado para poder conectarse al gestor de colas de coordinación remoto.

Acerca de esta tarea

Suposiciones: Autorización para conectarse al gestor de colas de coordinación remoto configurando el gestor de colas para permitir conexiones remotas.

Si desea más información sobre cómo configurar esto, consulte Conexión a un gestor de colas en modalidad cliente con autenticación de canal y Gestión de autoridades para recursos específicos de MFT.

Para supervisar gestores de colas y transferencias de archivos entre agentes en un sistema que no ejecuta Windows o Linux, configure IBM MQ Explorer para conectarse al sistema remoto utilizando los pasos siguientes:

Procedimiento

1. Inicie el IBM MQ Explorer local.
2. Cuando IBM MQ Explorer se haya cargado, pulse el botón derecho del ratón en la carpeta **Transferencia de archivos gestionada** y seleccione **Nueva configuración**.
3. Continúe por el asistente, seleccionado el gestor de colas de coordinación y mandatos y, a continuación, defina un nombre para la configuración.
4. Pulse **Finalizar** para completar la definición.
5. Cuando haya finalizado la definición, pulse el botón derecho del ratón en la definición y seleccione **Connect** (Conectar).

Resultados

Ahora inicie IBM MQ Explorer y utilícelo para supervisar la actividad de transferencia para la red de Managed File Transfer asociada con el gestor de colas de coordinación.

Tareas relacionadas

"Supervisión de transferencias de archivos que están en curso" en la página 228
Puede supervisar una transferencia de archivos que está en curso utilizando la pestaña **Transferencia de archivos gestionada - Progreso de la transferencia actual** en IBM MQ Explorer. Esta transferencia de archivos se puede iniciar desde IBM MQ Explorer o desde la línea de mandatos. la pestaña también muestra el progreso de las transferencias planificadas en el punto de inicio de las transferencias planificadas.

"Visualización del estado de transferencias de archivos en el Registro de transferencias" en la página 231
Puede ver los detalles de las transferencias de archivos utilizando el **Registro de transferencias** en IBM MQ Explorer. Puede tratarse de transferencias iniciadas desde la línea de mandatos o desde IBM MQ Explorer. También puede personalizar lo que aparece en el **Registro de transferencias**.

Visualización del estado de transferencias de archivos en el Registro de transferencias

Puede ver los detalles de las transferencias de archivos utilizando el **Registro de transferencias** en IBM MQ Explorer. Puede tratarse de transferencias iniciadas desde la línea de mandatos o desde IBM MQ Explorer. También puede personalizar lo que aparece en el **Registro de transferencias**.

Procedimiento

1. Expanda **Transferencia de archivos gestionada** en la vista Navegador y, a continuación, expanda el nombre del gestor de colas de coordinación para el que desea ver el registro de transferencias.
2. Pulse **Registro de transferencias** en la vista Navegador. En la vista Contenido aparece el **Registro de transferencias**.
3. La ventana **Registro de transferencias** muestra la información detallada siguiente sobre las transferencias de archivos:
   a) **Origen** El nombre del agente en el sistema en el que se encuentra el archivo de origen.
   b) **Destino** El nombre del agente en el sistema al que desea transferir el archivo.
   c) **Estado de terminación** El estado de la transferencia de archivos. El estado puede ser uno de los valores siguientes: "Iniciada", "En proceso", "Satisfactoria", "Parcialmente satisfactoria", "Cancelada" o "Fallida".
   d) **Propietario** El ID de usuario en el host que sometió la petición de transferencia.
   e) **Iniciada (huso horario seleccionado)** Fecha y hora a la que el agente de Managed File Transfer ha aceptado la solicitud de transferencia de archivo, presentada en el huso horario seleccionado del administrador. Para cambiar el huso horario visualizado, pulse **Ventana** > **Preferencias** > **IBM MQ Explorer** > **Managed File Transfer** y seleccione un huso horario alternativo en la lista **Huso horario:**. Pulse **Aceptar**.
   f) **Estado registrado (huso horario seleccionado)** (Esta columna no aparece de forma predeterminada. Puede elegir visualizar la columna utilizando la ventana **Configurar columnas de registro de transferencias**  .) La fecha y hora en que se registró el estado de terminación, en el huso horario seleccionado por el administrador.
   g) **Nombre de trabajo** Un identificador especificado por el usuario utilizando el parámetro **`-jn`** de **`fteCreateTransfer`** o en un script Ant
   h) **ID de transferencia** El identificador exclusivo de la transferencia de archivo.
   i) **Connect: Direct** Se listan los detalles sobre **Número de proceso**, **Nombre de proceso**, **Nodo primario**, **Nodo secundario**, **Tipo de origen** y **Tipo de destino**.

Resultados

Nota: El formato interno del Registro de transferencias se cambió en IBM MQ 8.0.0 Fix Pack 1 para el APAR IC99545. Como resultado, si se actualiza un IBM MQ Explorer a la versión 8.0.0.1 o posterior y, a continuación, se restaura a la versión 8.0.0.0, no se visualizará ningún XML de auditoría para las

transferencias que han tenido lugar mientras IBM MQ Explorer estaba en la versión 8.0.0.1. El panel XML en la ventana **Propiedades** para estas transferencias contendrá un recuadro de texto vacío.

Para ver más detalles sobre una transferencia completada, expanda la transferencia que le interese pulsando el signo más (+). A continuación, puede ver todos los nombres de archivo de origen y destino incluidos en dicha transferencia. No obstante, si la transferencia está actualmente en proceso y consta de muchos archivos, sólo puede ver los archivos que ya se han transferido hasta el momento.

Para renovar lo que se visualiza en el **Registro de transferencias**, pulse el botón **Renovar**  en la barra de herramientas de la vista Contenido. La información de transferencia de archivos en el Registro de transferencias permanece en el registro después de que detenga y reinicie IBM MQ Explorer. Si desea suprimir todas las transferencias de archivo completas del registro, pulse **Eliminar transferencias completas**  en la barra de herramientas de la vista Contenido.

Para suprimir una transferencia de archivos individual completada del registro, pulse el botón derecho del ratón en la transferencia y pulse **Suprimir**. Si suprime una transferencia, no se detiene o cancela una transferencia que está en proceso o que se ha planificado; se suprimen solamente los datos históricos almacenados.

Para copiar el identificador exclusivo de una transferencia en el portapapeles, pulse el botón derecho del ratón en esa transferencia y pulse **Copiar ID**.

Los metadatos y el XML de la auditoría completa de la transferencia están disponibles en el menú emergente, en la acción **Propiedades**.

Tareas relacionadas

“Supervisión de transferencias de archivos que están en curso” en la página 228
Puede supervisar una transferencia de archivos que está en curso utilizando la pestaña **Transferencia de archivos gestionada - Progreso de la transferencia actual** en IBM MQ Explorer. Esta transferencia de archivos se puede iniciar desde IBM MQ Explorer o desde la línea de mandatos. la pestaña también muestra el progreso de las transferencias planificadas en el punto de inicio de las transferencias planificadas.

“Configuración del registro de transferencias” en la página 232
Puede configurar qué información desea visualizar y cómo se visualiza en el **Registro de transferencias** es IBM MQ Explorer.

“Establecimiento de un tiempo de espera para la recuperación de transferencias estancadas” en la página 337
Puede establecer un tiempo de espera de recuperación de transferencia para las transferencias de archivos estancadas que se aplique a todas las transferencias para un agente de origen. También puede establecer un tiempo de espera de recuperación de transferencia para una transferencia individual. Si establece una cantidad de tiempo específica, en segundos, durante la cual un agente de origen sigue intentando recuperar una transferencia de archivos estancada y la transferencia no se realiza correctamente cuando el agente alcanza el tiempo de espera, la transferencia falla.

Configuración del registro de transferencias

Puede configurar qué información desea visualizar y cómo se visualiza en el **Registro de transferencias** es IBM MQ Explorer.

Acerca de esta tarea

Para reorganizar el orden de las columnas en **Registro de transferencias**, pulse el título de la columna que desea mover y arrastre la columna hasta la nueva posición. El nuevo orden de columnas sólo se mantendrá hasta la próxima vez que detenga y reinicie IBM MQ Explorer.

Para filtrar entradas en el **Registro de transferencias**, especifique una serie en el campo **Filtrar las entradas de registro visualizadas**. Para restaurar todas las entradas en el registro, suprima la serie especificada del campo. Se puede utilizar cualquier expresión regular Java válida en este campo. Si desea más información, consulte Expresiones regulares utilizadas por MFT.

Para personalizar qué columnas se visualizan en el Registro de transferencias, utilice **Configurar columnas del registro de transferencias**  . Utilice los pasos siguientes para iniciar y utilizar la ventana **Configurar columnas del registro de transferencias**.

Procedimiento

1. Asegúrese de que el **Registro de transferencias** está abierto en la vista Contenido. Pulse **Configurar columnas de registro de transferencias**  en la barra de herramientas de la vista Contenido. Se abre la ventana **Configurar columnas del registro de transferencias**.
2. Para personalizar la vista del **Registro de transferencias**, seleccione o deseleccione los recuadros de selección individuales de las columnas que desea mostrar u ocultar. Puede pulsar **Seleccionar todo** y luego **Aceptar** para seleccionar todos los recuadros de selección o **Deseleccionar todo** y luego **Aceptar** para borrar la marca de selección de todos los recuadros.

Tareas relacionadas

"Supervisión de transferencias de archivos que están en curso" en la página 228
Puede supervisar una transferencia de archivos que está en curso utilizando la pestaña **Transferencia de archivos gestionada - Progreso de la transferencia actual** en IBM MQ Explorer. Esta transferencia de archivos se puede iniciar desde IBM MQ Explorer o desde la línea de mandatos. la pestaña también muestra el progreso de las transferencias planificadas en el punto de inicio de las transferencias planificadas.

"Visualización del estado de transferencias de archivos en el Registro de transferencias" en la página 231
Puede ver los detalles de las transferencias de archivos utilizando el **Registro de transferencias** en IBM MQ Explorer. Puede tratarse de transferencias iniciadas desde la línea de mandatos o desde IBM MQ Explorer. También puede personalizar lo que aparece en el **Registro de transferencias**.

Supervisión de recursos de MFT

Puede supervisar recursos de Managed File Transfer; por ejemplo, una cola o un directorio. Cuando se cumple una condición en este recurso, el supervisor de recursos inicia una tarea, por ejemplo una transferencia de archivos. Puede crear un supervisor de recursos utilizando el mandato **`fteCreateMonitor`** o la vista **Supervisores** en el plug-in de Managed File Transfer para IBM MQ Explorer.

Acerca de esta tarea

La supervisión de recursos de Managed File Transfer utiliza la siguiente terminología:

Supervisor de recursos
: Un supervisor de recursos es un proceso que sondea un recurso (por ejemplo, un directorio o una cola) a intervalos regulares predefinidos para ver si el contenido del recurso ha cambiado. En caso de que haya cambiado, el contenido se compara con el conjunto de condiciones de este supervisor. Si se encuentra alguna coincidencia, se iniciará la tarea de este supervisor.

Recurso
: Recurso del sistema que el supervisor de recursos examina en cada intervalo de sondeo para compararlo con las condiciones de desencadenante. Colas, directorios o estructuras de directorios anidados pueden ser el recurso supervisado.

Condición y condición de desencadenante
: Una condición es una expresión que se evalúa (normalmente, respecto al contenido del recurso supervisado). Si la expresión se evalúa como true (verdadera), la condición contribuye a la condición desencadenante global.

  La condición de desencadenante es la condición global, que se cumple cuando se cumplen todas las condiciones. Cuando se cumple la condición desencadenante, la tarea puede proseguir.

Tarea
: Una tarea es la operación que se inicia cuando se cumple la condición o el conjunto de condiciones de desencadenante. Las tareas soportadas son la transferencia de archivos y la llamada de mandatos.

Archivo desencadenante

Un archivo desencadenante es un archivo que se coloca en un directorio supervisado para indicar que se puede iniciar una tarea (normalmente, una transferencia). Por ejemplo, podría indicar que todos los archivos que se van procesar han llegado a una ubicación conocida y se pueden transferir o pueden ser objeto de otras operaciones. El nombre del archivo desencadenante se puede utilizar para especificar los archivos que se van a transferir, mediante la sustitución de variables. Para obtener más información, consulte "Personalización de las tareas del supervisor de recursos de MFT con sustitución de variables" en la página 246.

El archivo desencadenante también se conoce como archivo preparado o archivo en formato go. Sin embargo, en esta documentación se suele denominar archivo desencadenante.

La supervisión de recursos no está soportada en agentes de puente de protocolo o agentes de puente Connect:Direct .

Conceptos relacionados

Guía para configurar un supervisor de recursos de MFT para evitar sobrecargar un agente

Referencia relacionada

`fteCreateMonitor`: crear un supervisor de recursos de MFT

fteListMonitors: listar supervisores de recursos de MFT

FteDeleteMonitor: suprimir un supervisor de recursos de MFT

Formatos de mensajes de solicitud del supervisor de MFT

Conceptos de supervisión de recursos de MFT

Una visión general de los conceptos clave de la característica de supervisión de recursos de Managed File Transfer.

Supervisores de recursos

Puede crear un supervisor de recursos utilizando el mandato **`fteCreateMonitor`** , que crea e inicia un nuevo supervisor de recursos desde la línea de mandatos. El supervisor de recursos está asociado con un agente de Managed File Transfer y solo está activo cuando el agente se está ejecutando. Cuando se detiene el agente de supervisión, también lo hace el supervisor. Si el agente ya se está ejecutando cuando se crea el supervisor de recursos, el supervisor de recursos se inicia inmediatamente. El agente de supervisión también debe ser el agente de origen de la tarea iniciada por el supervisor de recursos.

Los nombres de supervisor de recursos deben ser exclusivos dentro del agente. El nombre del supervisor de recursos debe tener un mínimo de un carácter de longitud y no debe contener caracteres de asterisco (*), porcentaje (%) o signo de interrogación (?). Las mayúsculas y minúsculas que se incluyen en un nombre de supervisor de recursos se omiten y el nombre de supervisor de recursos se convierte a mayúsculas. Si intenta crear un supervisor de recursos con un nombre que ya existe, la solicitud se ignora y el intento se registra en el tema de registro del supervisor de recursos.

Nota: No puede crear un supervisor de recursos con una definición de tarea que contenga transferencias planificadas.

Antes de IBM MQ 9.3.0, la única forma de detener un supervisor de recursos es deteniendo el agente que ejecuta la operación de supervisión. Para reiniciar un supervisor de recursos, debe reiniciar el agente completamente. Desde IBM MQ 9.3.0, puede iniciar y detener supervisores de recursos sin necesidad de detener o reiniciar un agente. Para obtener más información, consulte "Iniciar y detener supervisores de recursos" en la página 237.

No existe ninguna restricción en cuanto al número de supervisores de recursos que se pueden crear en un agente y todos ellos se ejecutan con la misma prioridad. Tenga en cuenta las implicaciones del solapamiento de recursos supervisados, de las condiciones desencadenantes en conflicto y de la frecuencia con la que se sondean los recursos.

Los supervisores de recursos solapados pueden causar:

- Posible contención en la ubicación/elementos de origen.

- Posibles solicitudes de transferencia duplicadas para los mismos elementos de origen.
- Anomalías o errores inesperados para las transferencias debido a conflictos de elementos de origen.

Si varios supervisores exploran la misma ubicación y se pueden desencadenar en los mismos elementos, puede potencialmente terminar con el problema de dos supervisores diferentes que envían solicitudes de transferencia gestionadas para el mismo elemento.

Los supervisores de recursos consultan el contenido de los recursos después de cada periodo de intervalo de sondeo. El contenido del recurso se compara con las condiciones desencadenantes y, si se cumplen estas condiciones, se llama a la tarea asociada al supervisor de recursos.

La tarea se inicia asíncronamente. Si se encuentra una coincidencia de condición y se inicia la tarea, el supervisor de recursos continúa sondeando para realizar más cambios en el contenido del recurso. Así, por ejemplo, si se ha producido una coincidencia porque un archivo denominado `reports.go` ha llegado a un directorio supervisado, la tarea se iniciaría una vez. En el próximo intervalo de sondeo, aunque el archivo siga existiendo, la tarea no se vuelve a iniciar. No obstante, si el archivo se suprime y luego se vuelve a colocar en el directorio, o si el archivo se actualiza (de forma que se cambia el atributo de fecha de última modificación), la siguiente comprobación de condición desencadenante hace que se vuelva a invocar la tarea.

En versiones anteriores a IBM MQ 9.1.5, si un supervisor de recursos realiza un sondeo que lleva más tiempo que el intervalo de sondeo, esto significa que el sondeo siguiente se inicia tan pronto como finaliza el actual sin espacio entre ellos, lo que podría tener afectar a la agilidad con que los supervisores de recursos envían trabajo a un agente. Esto podría provocar problemas de rendimiento si los elementos que se encuentran durante el primer sondeo siguen allí cuando se produce el segundo.

El supervisor de recursos utiliza el servicio ScheduledExecutore inicia el siguiente sondeo sólo después de la finalización del sondeo anterior más el tiempo de intervalo de sondeo configurado. Esto significa que siempre habrá una brecha entre los intervalos de sondeo, en lugar de tener otro sondeo que empiece inmediatamente después del sondeo anterior, si el tiempo de sondeo era más largo que el intervalo de sondeo.

Si un archivo no se ha podido transferir, puede borrar el historial del supervisor de recursos, lo que permite que se envíe otra solicitud de transferencia sin necesidad de suprimir el archivo y colocarlo de nuevo en el directorio, o actualizar el archivo para cambiar su atributo de fecha de última modificación. El borrado del historial es útil, por ejemplo, en situaciones en las que es necesario transferir el archivo, pero no es posible modificarlo. Para obtener más información, consulte "Borrado del historial del supervisor de recursos" en la página 264.

Recursos

Los supervisores de recursos de Managed File Transfer pueden sondear el contenido de los dos tipos de recursos siguientes:

Directorios o estructuras de directorios anidados

Un escenario habitual es supervisar un directorio para ver si hay un archivo desencadenante. Es posible que una aplicación externa procese varios archivos y los coloque en un directorio de origen conocido. Cuando la aplicación ha terminado de procesarse, indica que los archivos están listos para su transferencia o pueden ser objeto de otras operaciones, colocando un archivo desencadenante en una ubicación supervisada. El archivo desencadenante puede ser detectado por un supervisor de recursos de Managed File Transfer y se inicia la transferencia de esos archivos del directorio de origen a otro Managed File Transfer Agent.

De forma predeterminada, se supervisa el directorio especificado. Para examinar también los subdirectorios, establezca el nivel de recurrencia en el mandato **`fteCreateTransfer`**.

Los siguientes son dos ejemplos de supervisión de un directorio:

- Supervise un archivo desencadenante (por ejemplo, `trigger.file`) y luego transfiera un comodín (por ejemplo, `*.zip`).
- Supervise `*.zip` y luego transfiera `${FilePath}` (por ejemplo, el archivo que ha desencadenado la transferencia). Para obtener más información sobre la sustitución de variables, consulte

"Personalización de las tareas del supervisor de recursos de MFT con sustitución de variables" en la página 246.

Nota: No cree un supervisor que supervise `*.zip` y, a continuación, transfiera `*.zip`. El supervisor intenta iniciar una transferencia de `*.zip` para cada archivo .zip del sistema. Es decir, el supervisor genera * número de transferencias para `*.zip`.

Para obtener un ejemplo de la creación de un supervisor de recursos para supervisar un directorio, consulte "Supervisar un directorio y utilizar la sustitución de variables" en la página 243.

Colas de IBM MQ

Un ejemplo de supervisión de un cola es que una aplicación externa puede estar generando mensajes y colocándolos en una cola conocida con el mismo ID de grupo. Cuando la aplicación ha terminado de poner mensajes en la cola, esto indica que el grupo está completo. El grupo completo de mensajes puede ser detectado por un supervisor de recursos de Managed File Transfer y se inicia la transferencia de un grupo de mensajes de la cola de origen a un archivo. Para obtener un ejemplo de la creación de un supervisor de recursos para supervisar una cola, consulte "Ejemplo: configuración de un recurso MFT" en la página 245.

Nota: Puede especificar un solo supervisor por cola. Si especifica más de un supervisor para sondear una cola de IBM MQ, se produce un comportamiento impredecible.

La supervisión de conjuntos de datos no está soportada.

Condiciones y condiciones de desencadenante

La condición se cumple cuando el recurso contiene un valor que coincide con alguna serie o algún patrón. Las condiciones pueden ser una de las siguientes:

- Coincidencia de nombres de archivo (patrón)
- Ninguna coincidencia en el nombre de archivo (patrón)
- Tamaño de archivo
- Coincidencia si el tamaño de archivo sigue siendo el mismo para un número de sondeos

La coincidencia de nombres de archivos se puede expresar como:

- Coincidencia exacta de series
- Coincidencia de comodines simples como se describe en Utilización de caracteres comodín con MFT
- Coincidencia de expresiones regulares

Los nombres de archivo también se pueden excluir de la comparación de nombres de archivo utilizando un comodín o una expresión regular Java que identifique los nombres de archivo que nunca coinciden.

Cuando se detecta un archivo coincidente, se conserva la indicación de fecha y hora de última modificación. Si en sondeos posteriores se detecta que el archivo ha cambiado, la condición desencadenante se cumple de nuevo y la tarea se inicia. Si la condición consiste en detectar cuándo un archivo no existe, si ningún archivo del directorio supervisado coincide con el patrón de nombres de archivos, la tarea se inicia. Si luego se añade un archivo al directorio que sí coincide con el patrón de nombre de archivo, la tarea sólo se inicia si luego el archivo se suprime.

Tareas

Managed File Transfer da soporte a los dos tipos de tareas siguientes que se pueden configurar para que las inicien los supervisores de recursos:

Tareas de transferencia de archivos

Las tareas de transferencia de archivos se definen de la misma forma que cualquier otra transferencia de archivos. Una forma útil de generar el XML de tarea que necesita un supervisor es ejecutar el mandato fteCreateTransfer con el parámetro **`-gt`**. Este mandato genera una definición de tarea como un documento XML, incluyendo la especificación de la transferencia. A continuación, pasa el nombre del documento XML de tarea como valor del parámetro **`-mt`** en el mandato fteCreateMonitor. Cuando el mandato **`fteCreateMonitor`** se ejecuta, lee el documento XML de tarea. Una vez ejecutado el

mandato **fteCreateMonitor**, los cambios realizados en el archivo XML de tarea no son utilizados por el supervisor.

Cuando se utiliza una tarea de transferencia de archivos, puede seleccionar cuántas condiciones desencadenantes se procesan por lotes en una tarea. El valor predeterminado es que una condición desencadenante inicie una tarea. Puede ejecutar el mandato fteCreateMonitor con la opción **-bs** para seleccionar el número de condiciones de desencadenante que se agrupan en una tarea.

Tareas de mandato

Las tareas de mandato pueden ejecutar scripts Ant, llamar a programas ejecutables o ejecutar trabajos JCL. Para obtener más información, consulte "Configuración de tareas de supervisión de MFT para iniciar mandatos y scripts" en la página 239.

Archivos desencadenantes

Puede utilizar el contenido de un archivo desencadenante en un supervisor de recursos para definir un conjunto de archivos que deben transferirse en una única solicitud de transferencia. Cada vez que se detecte un archivo desencadenante coincidente, su contenido se analizará en busca de vías de acceso de archivo de origen y, opcionalmente, vías de acceso de archivo de destino. Estas vías de acceso de archivo se utilizarán para definir elementos del archivo XML de transferencia de tarea que especifique, que se someterá como una única solicitud de transferencia al agente. La definición del supervisor de recursos determina si el contenido del desencadenante está habilitado.

El formato de cada archivo desencadenante es una única vía de acceso de archivo que debe transferirse en cada línea de texto. El formato predeterminado de la línea es una única vía de acceso de archivo de origen, o una vía de acceso de origen y destino separada por una coma.

Para obtener más información y ejemplos, consulte "Utilización de un archivo desencadenante" en la página 255.

Iniciar y detener supervisores de recursos

Antes de IBM MQ 9.3.0, la única forma de detener un supervisor de recursos es deteniendo el agente que ejecuta la operación de supervisión. Para reiniciar un supervisor de recursos, debe reiniciar el agente completamente. Para obtener más información, consulte "Iniciar un agente de MFT" en la página 214 y "Detener un agente de MFT" en la página 220.

Desde IBM MQ 9.3.0, puede iniciar y detener los supervisores de recursos sin tener que detener o reiniciar un agente, utilizando los mandatos **fteStartMonitor** y **fteStopMonitor**. Esto es útil, por ejemplo, en las siguientes situaciones:

- Si un agente tiene varios supervisores de recursos y unos pocos tienen errores pero el resto de supervisores de recursos están funcionando correctamente, de manera que solo desea reiniciar los supervisores de recursos fallidos.
- Si desea detener un supervisor de recursos para llevar a cabo tareas de mantenimiento, o si el supervisor de recursos no es necesario durante un tiempo determinado y no quiere que se ejecute innecesariamente, ya que consume valiosos recursos del sistema.

Para obtener más información, consulte "Iniciar un supervisor de recursos de MFT" en la página 260 y "Detener un supervisor de recursos de MFT" en la página 261.

Tabla 9. Comportamiento de un supervisor de recursos en función del mandato que se ejecuta

Mandato	Comportamiento del supervisor de recursos
fteStartMonitor	Si el agente está en ejecución, el supervisor de recursos se inicia si está detenido en este momento.
fteStopMonitor	Si el agente está en ejecución, el supervisor de recursos se detiene si está iniciado en este momento.

Tabla 9. Comportamiento de un supervisor de recursos en función del mandato que se ejecuta (continuación)	
Mandato	**Comportamiento del supervisor de recursos**
fteStartAgent	El supervisor de recursos se inicia como parte del inicio del agente, independientemente de las llamadas anteriores a **fteStopMonitor**.
fteStopAgent	Se detiene cualquier supervisor de recursos que esté en ejecución.

Copia de seguridad y restauración de supervisores de recursos

Puede hacer copia de seguridad de los supervisores de recursos ya definidos para poder volver a utilizarlos en el futuro. Hay varias opciones que puede utilizar como se indica a continuación:

- Utilice el mandato **fteCreateMonitor** con el parámetro **-ox** para exportar una configuración de supervisor de recursos a un archivo XML y con el parámetro **-ix** para restaurar un supervisor de recursos importando la configuración de supervisor de recursos desde un archivo XML.
- Utilice el mandato **fteListMonitors** con **-ox** para exportar la definición de un único supervisor de recursos a un archivo XML.
- Utilice el mandato **fteListMonitors** con **-od** para exportar varias definiciones de supervisor de recursos a un directorio especificado. Cada definición de supervisor de recursos se guarda en el archivo XML aparte. También puede utilizar la opción **-od** para exportar una única definición de supervisor de recursos a un directorio especificado.

Para obtener más información, consulte "Copia de seguridad y restauración de supervisores de recursos de MFT" en la página 263.

Registro del supervisor de recursos

Managed File Transfer incluye el registro del supervisor de recursos. Para obtener más información, consulte "Registro de supervisores de recursos de MFT" en la página 258.

Conceptos relacionados

"Personalización de las tareas del supervisor de recursos de MFT con sustitución de variables" en la página 246
Cuando se cumplen las condiciones desencadenantes de un supervisor de recursos activo, se llama a la tarea definida. Además de llamar a la tarea de transferencia o mandato con el mismo agente de destino o el mismo nombre de archivo de destino cada vez, también puede modificar la definición de tarea durante la ejecución. Para ello, inserte los nombres de variables en el XML de definición de tarea. Cuando el supervisor determina que se cumplen las condiciones de desencadenante y que la definición de tarea contiene nombres de variable, sustituye los nombres de variable por los valores de variable y, a continuación, llama a la tarea.

Tareas relacionadas

"Configuración de tareas de supervisión de MFT para iniciar mandatos y scripts" en la página 239
Los supervisores de recursos no se limitan a realizar transferencias de archivos y la tarea asociada. También puede configurar el supervisor para invocar otros mandatos desde el agente de supervisión, incluidos los programas ejecutables, los scripts Ant o los trabajos JCL. Para llamar a los mandatos, edite el formato XML de definición de tarea del supervisor para incluir uno o varios elementos de mandatos con los parámetros de llamada de mandatos correspondientes, como por ejemplo, argumentos y propiedades.

"Ejemplo: configuración de un recurso MFT" en la página 245
Puede especificar una cola de IBM MQ como el recurso que va a supervisar un supervisor de recursos utilizando el parámetro **-mq** con el mandato **fteCreateMonitor**.

"Supervisión de una cola y utilización de sustitución de variables" en la página 251

Puede supervisar una cola y transferir mensajes de la cola supervisada a un archivo utilizando el mandato **fteCreateMonitor**. El valor de cualquier propiedad de mensaje de IBM MQ en el primer mensaje que se va a leer de la cola supervisada se puede sustituir en la definición XML de la tarea y se utiliza para definir el comportamiento de la transferencia.

Referencia relacionada

fteCreateMonitor: crear un supervisor de recursos de MFT

fteListMonitors: listar supervisores de recursos de MFT

FteDeleteMonitor: suprimir un supervisor de recursos de MFT

Configuración de tareas de supervisión de MFT para iniciar mandatos y scripts

Los supervisores de recursos no se limitan a realizar transferencias de archivos y la tarea asociada. También puede configurar el supervisor para invocar otros mandatos desde el agente de supervisión, incluidos los programas ejecutables, los scripts Ant o los trabajos JCL. Para llamar a los mandatos, edite el formato XML de definición de tarea del supervisor para incluir uno o varios elementos de mandatos con los parámetros de llamada de mandatos correspondientes, como por ejemplo, argumentos y propiedades.

Acerca de esta tarea

La vía de acceso de archivos al programa ejecutable, script Ant o trabajo JCL al que desea que el agente de supervisión llame, debe estar incluido en `commandPath` del agente de supervisión. Si desea más información sobre la propiedad de vía de acceso de mandato, consulte Propiedad commandPath de MFT.

Puede crear el documento XML de definición de tarea en una de las formas siguientes:

- Cree el documento XML de definición de tarea manualmente de acuerdo con el esquema de `FileTransfer.xsd`.
- Utilice un documento XML generado como base para la definición de tarea.

Si desea una tarea de transferencia o una tarea de mandato, la definición de tarea debe empezar por un elemento raíz `<request>`. El elemento hijo de `<request>` debe ser `<managedTransfer>` o `<managedCall>`. Normalmente, se selecciona `<managedCall>` cuando hay un solo mandato o script para ejecutar y `<managedTransfer>` si desea que la tarea incluya una transferencia de archivos y opcionalmente, hasta cuatro llamadas de mandatos.

Procedimiento

- Para crear el documento XML de definición de tarea manualmente de acuerdo con el esquema `FileTransfer.xsd`, consulte “Creación manual de un XML de definición de tarea según el esquema” en la página 239.
- Para crear una definición de tarea modificando un documento generado, edite el documento XML generado mediante el parámetro **fteCreateTransfer -gt**. Para obtener más información, consulte “Creación de un documento de definición de tarea modificando un documento generado” en la página 242.

Creación manual de un XML de definición de tarea según el esquema

Puede crear manualmente un archivo XML de definición de tarea de acuerdo con el esquema `FileTransfer.xsd`.

Acerca de esta tarea

El esquema `FileTransfer.xsd` se puede encontrar en *MQ_INSTALLATION_PATH*`/mqft/samples/schema`. Si desea más información sobre este esquema, consulte Formato de mensaje de solicitud de transferencia de archivos.

Ejemplo

El ejemplo siguiente muestra un documento XML de definición de tarea de ejemplo guardado como `cleanuptask.xml`, que utiliza el elemento <managedCall> para llamar a un script Ant denominado `RunCleanup.xml`. El script `RunCleanup.xml` Ant debe estar ubicado en commandPath del agente de supervisión.

```
<?xml version="1.0" encoding="UTF-8"?>
<request version="4.00" xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="FileTransfer.xsd">
   <managedCall>
     <originator>
       <hostName>hostName</hostName>
       <userID>userID</userID>
       <mqmdUserID>mqmdUserID</mqmdUserID>
     </originator>
     <agent QMgr="QM1" agent="AGENT1"/>
     <reply QMGR="QM1">reply</reply>
     <transferSet priority="1">
       <metaDataSet>
         <metaData key="name1">value1</metaData>
       </metaDataSet>
       <call>
           <command name="RunCleanup.xml" type="antscript" retryCount="2"
          retryWait="30" successRC="0">
           <target>check_exists</target>
           <target>copy_to_archive</target>
           <target>rename_temps</target>
             <target>delete_files</target>
           <property name="trigger.filename" value="${FileName}"/>
           <property name="trigger.path" value="${FilePath}"/>
           </command>
       </call>
     </transferSet>
     <job>
       <name>JOBCLEAN1</name>
     </job>
   </managedCall>
</request>
```

El elemento <agent> especifica el Managed File Transfer Agent que se configura con el script Ant con nombre en su commandPath.

La estructura <call><command>... define el ejecutable o script que desea ejecutar. El mandato toma un atributo `type` opcional que puede tener uno de los valores siguientes:

antscript
Ejecute un sript Ant en una JVM por separado.

executable
Invoque un programa ejecutable.

jcl
Invoque un trabajo JCL.

Si omite el atributo `type`, se utiliza el valor predeterminado executable.

El atributo `name` especifica el nombre del script Ant, del ejecutable o del trabajo JCL que desea ejecutar, sin ninguna información de vía de acceso. El agente busca el script o programa en las ubicaciones especificadas por la propiedad commandPath en el archivo `agent.properties` del agente.

El atributo `retrycount` especifica el número de veces que se intenta volver a llamar al programa si el programa no devuelve un código de retorno correcto. El valor asignado a este atributo no debe ser negativo. Si no especifica el atributo `retrycount`, se utiliza un valor predeterminado de cero.

El atributo `retrywait` especifica el tiempo de espera, en segundos, antes de volver a intentar la invocación del programa. El valor asignado a este atributo no debe ser negativo. Si no especifica el atributo `retrywait`, se utiliza un valor predeterminado de cero.

El atributo `successrc` es una expresión utilizada para determinar cuándo se ejecuta correctamente la invocación del programa. El código de retorno del proceso para el mandato se evalúa mediante esta expresión. El valor puede constar de una o varias expresiones combinadas con un carácter de barra

vertical (|) que indica el booleano OR o un carácter ampersand (&) que indica el booleano AND. Cada expresión puede ser uno de los tipos siguientes de expresión:

- Un número que indica una prueba de igualdad entre el código de retorno del proceso y el número.
- Un número que tiene como prefijo un carácter mayor que (>) para indicar una prueba mayor que entre el número y el código de retorno del proceso.
- Un número que tiene como prefijo un carácter menor que (<) para indicar una prueba menor que entre el número y el código de retorno del proceso.
- Un número precedido de un carácter de signo de exclamación (!) para indicar una prueba de no igualdad entre el número y el código de retorno del proceso. Por ejemplo: >2&<7&!5|0|14 significa que los códigos de retorno siguientes son satisfactorios: 0, 3, 4, 6, 14. Los demás códigos de retorno se interpretan como no satisfactorios.

Si no especifica el atributo `successrc`, se utiliza un valor predeterminado de cero. Esto significa que se interpreta que el mandato se ha ejecutado satisfactoriamente si, y solamente si, dicho mandato devuelve un código de cero.

Para un script Ant, lo normal es especificar los elementos `<target>` y `<property>`. Los valores del elemento `<target>` deben coincidir con los nombres de destino en el script Ant.

Para programas ejecutables, puede especificar elementos `<argument>`. Los elementos de argumentos anidados especifican argumentos para transferir al programa que se invoca como parte de la invocación de programa. Los argumentos del programa se crean a partir de los valores especificados por los elementos de argumento en el orden en que se encuentran los elementos de argumento. Puede especificar cero o varios elementos de argumento de una invocación de programa.

El administrador define e inicia el supervisor como normal utilizando el documento XML de definición de tarea que incluye el elemento `<managedCall>`. Por ejemplo:

```
fteCreateMonitor -ma AGENT1 -mm QM1 -md /monitored -mn MONITOR01 -mt
   /tasks/cleanuptask.xml -pi 30 -pu seconds -tr match,*.go
```

La vía de acceso al documento XML de definición de transferencia debe estar en el sistema de archivos local desde el que se ejecuta el mandato **`fteCreateMonitor`** (en este ejemplo, `/tasks/cleanuptask.xml`). El documento `cleanuptask.xml` se utiliza sólo para crear el supervisor de recursos. Las tareas que las referencias de documento `cleanuptask.xml` (scripts Ant o trabajos JCL) deben estar en la vía de acceso del mandato del agente de supervisión. Cuando se cumple la condición desencadenante del supervisor, las variables del documento XML de definición de tarea se sustituyen por los valores reales del supervisor. Así, por ejemplo, ${FilePath} se sustituye en el mensaje de solicitud enviado al agente por `/monitored/cleanup.go`. El mensaje de solicitud se pone en la cola de mandatos de agente. El procesador de mandatos detecta que la solicitud es para una llamada de programa e inicia el programa especificado. Si se llama a un mandato de tipo `antscript`, se inicia una nueva JVM y la tarea Ant se ejecuta bajo la nueva JVM. Para obtener más información sobre la utilización de sustitución de variables, consulte Personalización de tareas con la sustitución de variables.

Conceptos relacionados

“Personalización de las tareas del supervisor de recursos de MFT con sustitución de variables” en la página 246

Cuando se cumplen las condiciones desencadenantes de un supervisor de recursos activo, se llama a la tarea definida. Además de llamar a la tarea de transferencia o mandato con el mismo agente de destino o el mismo nombre de archivo de destino cada vez, también puede modificar la definición de tarea durante la ejecución. Para ello, inserte los nombres de variables en el XML de definición de tarea. Cuando el supervisor determina que se cumplen las condiciones de desencadenante y que la definición de tarea contiene nombres de variable, sustituye los nombres de variable por los valores de variable y, a continuación, llama a la tarea.

Referencia relacionada

Formato de mensaje de solicitud de transferencia de archivos

Propiedad commandPath de MFT

Creación de un documento de definición de tarea modificando un documento generado

Puede crear el documento de definición de tarea del supervisor modificando el documento XML generado por la opción **-gt** de **fteCreateTransfer**.

Acerca de esta tarea

El documento generado tiene una <request> seguida del elemento <managedTransfer>. Para convertir esta definición de tarea en una estructura <managedCall> válida, siga estos pasos:

Procedimiento

1. Sustituya las etiquetas de inicio y finalización de <managedTransfer> por etiquetas <managedCall>.
2. Elimine los nodos hijo y elemento <schedule>.
3. Sustituya las etiquetas de inicio y finalización de <sourceAgent> por <agent> para que coincidan con los detalles de configuración del agente de supervisión.
4. Elimine los elementos <destinationAgent> y <trigger>.
5. Elimine los elementos <item>.
6. Elimine los elementos preSourceCall, postSourceCall, preDestinationCallo postDestinationCall .
7. Inserte una nueva estructura <call>...</call> dentro del elemento <transferSet>. Esta estructura contiene la definición de mandato tal como se muestra en el ejemplo siguiente:

```
<call>
           <command name="RunCleanup.xml" type="antscript" retryCount="2"
          retryWait="30" successRC="0">
           <target>check_exists</target>
           <target>copy_to_archive</target>
           <target>rename_temps</target>
             <target>delete_files</target>
           <property name="trigger.filename" value="${FileName}"/>
           <property name="trigger.path" value="${FilePath}"/>
           </command>
       </call>
```

Ejemplo

También puede retener el elemento <managedTransfer> incluidos todos los detalles de transferencia de archivos e insertar hasta cuatro llamadas de mandatos. En este caso, inserta cualquier selección de los siguientes elementos de llamada entre los elementos <metaDataSet> y <item>:

preSourceCall
Llame a un programa en el agente de origen antes de iniciar la transferencia.

postSourceCall
Llame a un programa en el agente de origen después de completar la transferencia.

preDestinationCall
Llame a un programa en el agente de destino antes de iniciar la transferencia.

postDestinationCall
Llame a un programa en el agente de destino después de completar la transferencia.

Cada uno de estos elementos toma la estructura del elemento <command> tal como se describe en el ejemplo anterior. El esquema FileTransfer.xsd define los tipos utilizados por los diversos elementos de llamada.

En el ejemplo siguiente se muestra preSourceCall, postSourceCall, preDestinationCall y postDestinationCall en un documento de definición de tarea:

```
⋮
    <transferSet priority="1">
       <metaDataSet>
         <metaData key="key1">value1</metaData>
```

```
        </metaDataSet>
        <preSourceCall>
          <command name="send.exe" retryCount="0" retryWait="0" successRC="0"
            type="executable">
            <argument>report1.pdf</argument>
            <argument>true</argument>
          </command>
        </preSourceCall>
        <postSourceCall>
          <command name="//DO_IT.JCL" retryCount="0" retryWait="0" successRC="0"
            type="jcl">
            <argument>argument</argument>
          </command>
        </postSourceCall>
        <preDestinationCall>
          <command name="ant_script.xml" retryCount="0" retryWait="0" successRC="0"
            type="antscript">
            <target>step1</target>
            <property name="name" value="value"/>
          </command>
        </preDestinationCall>
        <postDestinationCall>
          <command name="runit.cmd" retryCount="0" retryWait="0" successRC="0" />
        </postDestinationCall>
        <item checksumMethod="none" mode="binary">
⋮
```

Puede combinar diferentes tipos de mandato en la transferencia. Los elementos argument, target y property son opcionales.

Supervisar un directorio y utilizar la sustitución de variables

Puede supervisar un directorio utilizando el mandato **fteCreateMonitor**. El valor de una variable de sustitución se puede sustituir en la definición del XML de tarea y se puede utilizar para definir el comportamiento de la transferencia.

Acerca de esta tarea

En este ejemplo, el agente de origen se denomina AGENT_HOP. El directorio que AGENT_HOP supervisa se llama `/test/monitored`. El agente sondea el directorio cada 5 minutos.

Después de escribir un archivo `.zip` en el directorio, la aplicación que escribe el archivo en el directorio escribe un archivo desencadenante en el mismo directorio. El nombre del archivo desencadenante es el mismo que el nombre del archivo `.zip`, pero tiene una extensión de archivo diferente. Por ejemplo, después de escribir el archivo `file1.zip` en el directorio, el archivo `file1.go` se escribe en el directorio. El supervisor de recursos supervisa el directorio de los archivos que coinciden con el patrón `*.go` y, a continuación, utiliza la sustitución de variable para solicitar una transferencia del archivo `.zip` asociado.

Procedimiento

1. Cree el XML de tarea que define la tarea que el supervisor lleva a cabo cuando se desencadena.

```
<?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>blue.example.com</hostName>
       <userID>USER1</userID>
     </originator>
     <sourceAgent agent="AGENT_HOP" QMgr="QM_HOP" />
     <destinationAgent agent="AGENT_SKIP" QMgr="QM_SKIP" />
     <transferSet>
       <item mode="binary" checksumMethod="none">
         <source>
           <file>/test/monitored/${fileName{token=1}{separator=.}}.zip</file>
         </source>
         <destination type="file" exist="overwrite">
           <file>/out/${fileName{token=1}{separator=.}}.zip</file>
         </destination>
```

```
      </item>
    </transferSet>
  </managedTransfer>
</request>
```

Las variables que se sustituyen por los valores asociados al archivo desencadenante están resaltadas en **negrita**. Este XML de tarea se guarda en el archivo `/home/USER1/task.xml`

2. Cree un supervisor de recursos para supervisar el directorio `/test/monitored`.

   Someta el siguiente mandato:

```
fteCreateMonitor -ma AGENT_HOP -mm QM_HOP -md /test/monitored
                 -mn myMonitor -mt /home/USER1/task.xml
                 -tr match,*.go -pi 5 -pu minutes
```

3. Un usuario o programa escribe el archivo `jump.zip` en el directorio `/test/monitored` y, a continuación, escribe el archivo `jump.go` en el directorio.
4. El supervisor se desencadena por la existencia del archivo `jump.go`. El agente sustituye la información sobre el archivo desencadenante en el XML de tarea.

   El resultado es la transformación del XML de tarea en:

```
<?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>blue.example.com</hostName>
       <userID>USER1</userID>
     </originator>
     <sourceAgent agent="AGENT_HOP" QMgr="QM_HOP" />
     <destinationAgent agent="AGENT_SKIP" QMgr="QM_SKIP" />
     <transferSet>
       <item mode="binary" checksumMethod="none">
         <source>
           <file>/test/monitored/jump.zip</file>
         </source>
         <destination type="file" exist="overwrite">
           <file>/out/jump.zip</file>
         </destination>
       </item>
     </transferSet>
   </managedTransfer>
</request>
```

Resultados

La transferencia definida por el script XML de tarea se lleva a cabo. AGENT_HOP lee el archivo `jump.zip` del directorio `/test/monitored` y se transfiere a un archivo denominado `/out/jump.zip` ubicado en el sistema donde se ejecuta AGENT_SKIP.

Conceptos relacionados

“Personalización de las tareas del supervisor de recursos de MFT con sustitución de variables” en la página 246

Cuando se cumplen las condiciones desencadenantes de un supervisor de recursos activo, se llama a la tarea definida. Además de llamar a la tarea de transferencia o mandato con el mismo agente de destino o el mismo nombre de archivo de destino cada vez, también puede modificar la definición de tarea durante la ejecución. Para ello, inserte los nombres de variables en el XML de definición de tarea. Cuando el supervisor determina que se cumplen las condiciones de desencadenante y que la definición de tarea contiene nombres de variable, sustituye los nombres de variable por los valores de variable y, a continuación, llama a la tarea.

Tareas relacionadas

“Configuración de tareas de supervisión de MFT para iniciar mandatos y scripts” en la página 239

Los supervisores de recursos no se limitan a realizar transferencias de archivos y la tarea asociada. También puede configurar el supervisor para invocar otros mandatos desde el agente de supervisión, incluidos los programas ejecutables, los scripts Ant o los trabajos JCL. Para llamar a los mandatos,

edite el formato XML de definición de tarea del supervisor para incluir uno o varios elementos de mandatos con los parámetros de llamada de mandatos correspondientes, como por ejemplo, argumentos y propiedades.

Referencia relacionada

fteCreateMonitor: crear un supervisor de recursos de MFT

Ejemplo: configuración de un recurso MFT

Puede especificar una cola de IBM MQ como el recurso que va a supervisar un supervisor de recursos utilizando el parámetro **-mq** con el mandato **fteCreateMonitor**.

Acerca de esta tarea

En este ejemplo, el recurso que se va a supervisar es la cola *MONITORED_QUEUE*. Esta cola debe estar en el gestor de colas del agente de supervisión, *QM_NEPTUNE*. La condición por la que la cola se supervisa es la presencia de un grupo completo de mensajes. La tarea que se realizará si se cumple la condición se define en el archivo `task.xml`.

Nota: No cree más de un supervisor de recursos para supervisar una cola individual. Si lo hace, se puede generar un comportamiento impredecible.

Procedimiento

Escriba el siguiente mandato:

```
fteCreateMonitor -ma AGENT_NEPTUNE -mn myMonitor -mm QM_NEPTUNE -mq MONITORED_QUEUE
  -mt task.xml -tr completeGroups -pi 5 -pu minutes
```

El supervisor comprueba la cola cada cinco minutos para ver si la condición `completeGroups` es verdadera. Si hay uno o más grupos completos en la cola, el supervisor ejecuta la tarea definida en el archivo `task.xml` una vez para cada grupo completo.

Conceptos relacionados

“Personalización de las tareas del supervisor de recursos de MFT con sustitución de variables” en la página 246

Cuando se cumplen las condiciones desencadenantes de un supervisor de recursos activo, se llama a la tarea definida. Además de llamar a la tarea de transferencia o mandato con el mismo agente de destino o el mismo nombre de archivo de destino cada vez, también puede modificar la definición de tarea durante la ejecución. Para ello, inserte los nombres de variables en el XML de definición de tarea. Cuando el supervisor determina que se cumplen las condiciones de desencadenante y que la definición de tarea contiene nombres de variable, sustituye los nombres de variable por los valores de variable y, a continuación, llama a la tarea.

Tareas relacionadas

“Configuración de tareas de supervisión de MFT para iniciar mandatos y scripts” en la página 239

Los supervisores de recursos no se limitan a realizar transferencias de archivos y la tarea asociada. También puede configurar el supervisor para invocar otros mandatos desde el agente de supervisión, incluidos los programas ejecutables, los scripts Ant o los trabajos JCL. Para llamar a los mandatos, edite el formato XML de definición de tarea del supervisor para incluir uno o varios elementos de mandatos con los parámetros de llamada de mandatos correspondientes, como por ejemplo, argumentos y propiedades.

“Supervisión de una cola y utilización de sustitución de variables” en la página 251

Puede supervisar una cola y transferir mensajes de la cola supervisada a un archivo utilizando el mandato **fteCreateMonitor**. El valor de cualquier propiedad de mensaje de IBM MQ en el primer mensaje que se va a leer de la cola supervisada se puede sustituir en la definición XML de la tarea y se utiliza para definir el comportamiento de la transferencia.

Referencia relacionada

fteCreateMonitor: crear un supervisor de recursos de MFT

Personalización de las tareas del supervisor de recursos de MFT con sustitución de variables

Cuando se cumplen las condiciones desencadenantes de un supervisor de recursos activo, se llama a la tarea definida. Además de llamar a la tarea de transferencia o mandato con el mismo agente de destino o el mismo nombre de archivo de destino cada vez, también puede modificar la definición de tarea durante la ejecución. Para ello, inserte los nombres de variables en el XML de definición de tarea. Cuando el supervisor determina que se cumplen las condiciones de desencadenante y que la definición de tarea contiene nombres de variable, sustituye los nombres de variable por los valores de variable y, a continuación, llama a la tarea.

Atención: Los nombres de variables no son sensibles a mayúsculas y minúsculas.

Las variables que se utilizan para la sustitución sólo están disponibles para condiciones desencadenantes positivas. Sólo las condiciones desencadenantes `match` y `fileSize` hacen que se sustituyan variables. Si se utiliza una condición `noMatch` y hay nombres de variable de sustitución en la definición de tarea, la tarea no se llama y el supervisor genera un código de retorno de 110 y el mensaje de error `BFGDM0060E`.

Si el recurso supervisado es una cola

El valor de cualquier propiedad de mensaje de IBM MQ del primer mensaje que se va a leer de la cola supervisada se puede sustituir en la definición XML de la tarea.

Las propiedades de mensaje definidas por el usuario tienen el prefijo `usr.`, pero no incluya este prefijo en el nombre de variable. Los nombres de variable deben ir precedidos de un carácter de signo de dólar ($) y escribirse entre llaves, {}.

Por ejemplo, `${destFileName}` se sustituye por el valor de la propiedad de mensaje `usr.destFileName` del primer mensaje que se va a leer de la cola de origen. Si desea más información, consulte Propiedades de mensaje MQ leídas por MFT de mensajes en colas de origen y “Supervisión de una cola y utilización de sustitución de variables” en la página 251.

Si una variable no está definida como una propiedad de mensaje, el supervisor notifica un error BFGDM0060E y devuelve el código de retorno 110 (la sustitución de la variable de tarea de supervisor ha fallado). Además, el agente escribe el siguiente mensaje de error en su registro de sucesos (`outputN.log`):

```
BFGDM0113W: Trigger failure for <monitor name> for reason BFGDM0060E: A monitor task could
not complete as a variable substitution <variable name> was not present.
```

Si el registro de supervisor de recursos moderado o detallado está habilitado para el supervisor, el supervisor escribe el siguiente mensaje en el registro de sucesos del supervisor de recursos del agente (`resmoneventN.log`):

```
BFGDM0060E: A monitor task could not complete as a variable substitution
<variable name> was not present.
```

Consulte “Registro de supervisores de recursos de MFT” en la página 258 para obtener más información sobre el registro del supervisor de recursos.

En la tabla siguiente se muestran las variables de sustitución que se proporcionan de forma predeterminada. Por ejemplo, `${AGENTNAME}` se sustituye por el nombre del agente de supervisor de recursos.

Tabla 10. Variables de sustitución proporcionadas de forma predeterminada

Variable	Descripción
AGENTNAME	Nombre del agente de supervisor de recursos.
NOMBRECOLA	El nombre de la cola que se está supervisando.

Tabla 10. Variables de sustitución proporcionadas de forma predeterminada (continuación)	
Variable	**Descripción**
ENCODING	La codificación de caracteres del primer mensaje en la cola o el primer mensaje de un grupo.
MESSAGEID	El ID de mensaje de IBM MQ del primer mensaje de la cola o el primer mensaje del grupo.
GROUPID	El ID de grupo de IBM MQ del grupo o el ID de mensaje si sólo se encuentra un único mensaje. Esta variable sólo se establece si se supervisan grupos completos.
CurrentTimeStamp	Una indicación de fecha y hora basada en la hora local a la que se desencadenó el supervisor. La indicación de fecha y hora es exclusiva para el agente.
CurrentTimeStamp UTC	Una indicación de fecha y hora basada en la hora, en el huso horario UTC, a la que se desencadenó el supervisor. La indicación de fecha y hora es exclusiva para el agente.

Si el recurso supervisado es un directorio

En la tabla siguiente se muestra el conjunto de nombres de variable que se pueden sustituir en la definición XML de tarea.

Tabla 11. Variables que se pueden sustituir	
Variable	**Descripción**
FilePath	Nombre completo de la vía de acceso del archivo desencadenante.
FileName	Parte de nombre de archivo del desencadenante.
LastModifiedTime	La hora a la que se modificó por última vez el archivo desencadenante. Esta hora se expresa como la hora local del huso horario en el que se ejecuta el agente y tiene un formato de hora ISO 8601.
LastModifiedDate	La fecha a la que se modificó por última vez el archivo desencadenante. Esta fecha se expresa como fecha local del huso horario en el que se ejecuta el agente y tiene el formato de fecha ISO 8601.
LastModifiedTimeUTC	La hora a la que se modificó por última vez el archivo desencadenante. Esta hora se expresa como la hora local convertida al huso horario UTC y tiene un formato de hora ISO 8601
LastModifiedDateUTC	La fecha a la que se modificó por última vez el archivo desencadenante. Esta fecha se expresa como fecha local convertida al huso horario UTC y se formatea como fecha ISO 8601.
AgentName	Nombre del agente de supervisor de recursos.
CurrentTimeStamp	Una indicación de fecha y hora basada en la hora local a la que se desencadenó el supervisor. La indicación de fecha y hora es exclusiva para el agente.
CurrentTimeStampUTC	Una indicación de fecha y hora basada en la hora, en el huso horario UTC, a la que se desencadenó el supervisor. La indicación de fecha y hora es exclusiva para el agente.

Si el recurso supervisado es un archivo desencadenante

La tabla siguiente muestra el conjunto de nombres de variable que se pueden sustituir cuando un supervisor de recursos utiliza el contenido de un archivo desencadenante para determinar los archivos que deben transferirse.

Tabla 12. Variables que se pueden sustituir al utilizar un archivo desencadenante	
Variable	**Descripción**
contentSource	El nombre completo de la vía de acceso del archivo de origen.
contentDestination	El nombre completo de la vía de acceso del archivo de destino.

Los nombres de variable deben ir precedidos por un carácter de signo de dólar ($) e ir entre llaves, {}. Por ejemplo, `${FilePath}` se sustituye por la vía de acceso de archivo completa del archivo desencadenante coincidente.

Existen dos palabras clave especiales que se pueden aplicar a los nombres de las variables para proporcionar un mayor perfeccionamiento. Son las siguientes:

token
El índice de señales que hay que sustituir (empezando por 1 a la izquierda y empezando por -1 a la derecha)

separator
Un carácter único para señalizar el valor de la variable. El valor predeterminado es el carácter de barra inclinada (/) en plataformas AIX and Linux o el carácter de barra inclinada invertida (\) en plataformas Windows , pero el separador puede ser cualquier carácter válido que pueda aparecer en el valor de la variable.

Si la palabra clave separator se especifica en un nombre de variable, el valor de la variable se divide en señales según el carácter de separador.

El valor que se asigna a la palabra clave token se utiliza como índice para seleccionar qué señal se debe utilizar para sustituir el nombre de variable. El índice de señal es relativo al primer carácter de la variable y se inicia en 1. Si no se especifica la palabra clave de señal, se inserta toda la variable.

Todos los valores que se sustituyen en un nombre de agente en el XML del mensaje se tratan sin distinguir entre mayúsculas y minúsculas. Todos los nombres de Managed File Transfer Agent están en mayúsculas. Si el valor `Paris` se sustituye en un atributo de agente en el XML del mensaje, este valor se interpreta como una referencia al agente `PARIS`.

Conceptos relacionados
“Ejemplos: Sustitución de variables para definiciones de supervisor de recursos” en la página 248
Ejemplos de sustitución de variables para definiciones de supervisor de recursos utilizando XML y IBM MQ Explorer.

Tareas relacionadas
Qué hacer si la sustitución de variable hace que varios archivos vayan a un solo nombre de archivo

Ejemplos: Sustitución de variables para definiciones de supervisor de recursos

Ejemplos de sustitución de variables para definiciones de supervisor de recursos utilizando XML y IBM MQ Explorer.

Ejemplos que muestran cómo funciona la sustitución de variable

Suponiendo que la vía de acceso de archivo al archivo desencadenante coincidente sea `c:\MONITOR\REPORTS\Paris\Report2009.doc` en Windows y `/MONITOR/REPORTS/Paris/Report2009.doc` en plataformas AIX and Linux , las variables se sustituyen tal como se muestra en la tabla siguiente.

Tabla 13. Cómo se sustituyen las variables	
Especificación de variable	**Después de la sustitución de variables**
${FilePath}	Windows :`c:\MONITOR\REPORTS\Paris\Report2009.doc` AIX and Linux : `/MONITOR/REPORTS/Paris/Report2009.doc`
${FilePath{token=1}{separator=.}}	Windows :`c:\MONITOR\REPORTS\Paris\Report2009` AIX and Linux : `/MONITOR/REPORTS/Paris/Report2009`
${FilePath{token=2}{separator=.}}	Windows: doc AIX and Linux : doc
${FilePath{token=3}}	Windows : INFORMES AIX and Linux : París

También puede especificar un índice de señales (tokens) negativo para seleccionar señales relativas al último carácter de la variable, como se muestra en la tabla siguiente. Los ejemplos de la tabla utilizan el mismo valor de variable, `c:\MONITOR\REPORTS\Paris\Report2009.doc` en Windows y `/MONITOR/REPORTS/Paris/Report2009.doc` en AIX and Linux.

Tabla 14. Ejemplos de uso de un índice de señales negativo	
Especificación de variable	**Después de la sustitución de variables**
${FilePath}	Windows :`c:\MONITOR\REPORTS\Paris\Report2009.doc` AIX and Linux : `/MONITOR/REPORTS/Paris/Report2009.doc`
${FilePath{token=-2}{separator=.}}	Windows :`c:\MONITOR\REPORTS\Paris\Report2009` AIX and Linux : `/MONITOR/REPORTS/Paris/Report2009`
${FilePath{token=-2}{separator=\}}	Windows : París AIX and Linux : París
${FilePath{token=-4}}	Windows : SUPERVISOR AIX and Linux : SUPERVISOR

Las variables que se utilizan para la sustitución sólo están disponibles para las siguientes condiciones de desencadenante positivas y la opción `noSizeChange`, que es una excepción a la regla de condición de desencadenante positiva:

- `match`
- `fileSize`
- `noSizeChange`

Si se utiliza una condición `noMatch` y hay nombres de variable de sustitución en la definición de tarea, no se llama a la tarea y el supervisor genera un código de retorno de 110 y el mensaje de error `BFGDM0060E`.

Ejemplo de uso de XML

El XML de la definición de tarea de ejemplo siguiente utiliza el nombre de agente supervisor como agente de origen para la transferencia (`Paris`), utiliza el penúltimo nombre de directorio en la vía de acceso de archivo como nombre de agente de destino para la transferencia (`Report2009`) y renombra el archivo transferido para que sea la raíz del nombre de archivo desencadenante con una extensión de `.rpt`.

```
<?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="${AgentName}" QMgr="QM1" />
     <destinationAgent agent="${FilePath{token=-2}}" QMgr="QMD" />
     <transferSet>
       <item mode="binary" checksumMethod="MD5">
         <source recursive="false" disposition="leave">
           <file>c:/incoming/reports/summary/report.doc</file>
         </source>
         <destination type="file" exist="overwrite">
           <file>/reports/${FileName{token=1}{separator=.}}.rpt</file>
         </destination>
       </item>
     </transferSet>
   </managedTransfer>
</request>
```

El resultado es la transformación del XML de tarea en:

```
<?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="Paris" QMgr="QMD" />
     <transferSet>
       <item mode="binary" checksumMethod="MD5">
         <source recursive="false" disposition="leave">
           <file>c:/incoming/reports/summary/report.doc</file>
         </source>
         <destination type="file" exist="overwrite">
           <file>/reports/Report2009.rpt</file>
         </destination>
       </item>
     </transferSet>
   </managedTransfer>
</request>
```

La variable `${FilePath{token=-2}}` en el atributo `agent` del elemento `<destinationAgent>` se sustituye por el valor `Paris`. Este valor se trata sin distinguir entre mayúsculas y minúsculas y se interpreta como una referencia al agente PARIS.

Ejemplos de uso de IBM MQ Explorer

Al crear un supervisor de recursos mediante IBM MQ Explorer,y una vez que se han especificado las propiedades de supervisor y las condiciones de desencadenante, se proporciona la opción para añadir elementos de transferencia al supervisor. En los ejemplos siguientes se muestra cómo se pueden utilizar las variables ${FilePath} y ${FileName} en el "**panel Añadir un elemento de transferencia**" para personalizar las transferencias que se producen a raíz de una coincidencia de supervisor de recursos.

Ejemplo 1

Para transferir simplemente el archivo de origen cuando se cumple una condición de desencadenante, se puede utilizar la variable ${FilePath}:

- Establezca el origen **Nombre de archivo** en `${FilePath}`.
- En el menú desplegable de **Tipo** para el destino, seleccione **Directorio**.
- Establezca el destino **Nombre de archivo** para que sea la ubicación a la que desea que se transfiera el archivo de origen, por ejemplo, podría ser `C:\MFT\out\`.

Ejemplo 2

Para transferir el archivo de origen a otra ubicación y cambiar la extensión del archivo, se puede utilizar la variable `${FileName}` junto con la variable `${FilePath}`:

En el ejemplo siguiente se presupone que la vía de acceso de archivo del archivo de origen es igual a `C:\MONITOR\REPORTS\Paris\Report2009.doc`:

- Establezca el origen **Nombre de archivo** en `${FilePath}`.
- Establezca el destino **Nombre de archivo** como la ubicación a la que desea que se transfiera el archivo de origen, seguido de `${FileName{token=1}{separator=.}}`, seguido de la nueva extensión del archivo. Por ejemplo, esto podría ser `C:\MFT\out\${FileName{token=1}{separator=.}}.rpt`, lo que equivaldría a `C:\MFT\out\Report2009.rpt` con el nombre de archivo de origen.

Ejemplo 3

Para utilizar parte de la vía de acceso del archivo de origen para determinar el destino de la transferencia, se puede utilizar la variable `${FilePath}` junto con las especificaciones de señal y separador.

En el ejemplo siguiente se presupone que la vía de acceso del archivo de origen es igual a `C:\MONITOR\REPORTS\Paris\Report2009.doc`.

Es posible utilizar parte de la vía de acceso del archivo de origen para determinar el destino del archivo. Utilizando el ejemplo de vía de acceso de archivo de `C:\MONITOR\REPORTS\Paris\Report2009.doc`, si el archivo se va a transferir a una carpeta en función de la ubicación del archivo de origen, es decir, `Paris` en este ejemplo, se podría realizar lo siguiente:

- Establezca el origen **Nombre de archivo** en `${FilePath}`.
- Establezca el **Nombre de archivo** de destino en el destino donde se sitúan las carpetas para cada ubicación, y luego añada la parte de destino de la vía de acceso y el nombre de archivo. Por ejemplo, esto podría ser `C:\MFT\out\${FilePath{token=-2}{separator=\}}\${FileName}`, lo que equivaldría a `C:\MFT\out\Paris\Report2009.doc` con el nombre de archivo de origen.

Conceptos relacionados

"Personalización de las tareas del supervisor de recursos de MFT con sustitución de variables" en la página 246

Cuando se cumplen las condiciones desencadenantes de un supervisor de recursos activo, se llama a la tarea definida. Además de llamar a la tarea de transferencia o mandato con el mismo agente de destino o el mismo nombre de archivo de destino cada vez, también puede modificar la definición de tarea durante la ejecución. Para ello, inserte los nombres de variables en el XML de definición de tarea. Cuando el supervisor determina que se cumplen las condiciones de desencadenante y que la definición de tarea contiene nombres de variable, sustituye los nombres de variable por los valores de variable y, a continuación, llama a la tarea.

Tareas relacionadas

Qué hacer si la sustitución de variable hace que varios archivos vayan a un solo nombre de archivo

Supervisión de una cola y utilización de sustitución de variables

Puede supervisar una cola y transferir mensajes de la cola supervisada a un archivo utilizando el mandato **`fteCreateMonitor`**. El valor de cualquier propiedad de mensaje de IBM MQ en el primer mensaje que se va a leer de la cola supervisada se puede sustituir en la definición XML de la tarea y se utiliza para definir el comportamiento de la transferencia.

Acerca de esta tarea

En este ejemplo, el agente de origen se denomina AGENT_VENUS, que se conecta a QM_VENUS. La cola que AGENT_VENUS supervisa se denomina START_QUEUE y se encuentra en QM_VENUS. El agente sondea la cola cada 30 minutos.

Cuando un grupo completo de mensajes se graba en la cola, la tarea de supervisor envía el grupo de mensajes a un archivo en uno de los muchos agentes de destino, que se conectan todos ellos al gestor de colas QM_MARS. El nombre del archivo al que se transfiere el grupo de mensajes se define mediante la propiedad de mensaje de IBM MQ `usr.fileName` en el primer mensaje del grupo. El nombre del agente al que se envía el grupo de mensajes se define mediante la propiedad de mensaje de IBM MQ `usr.toAgent` en el primer mensaje del grupo. Si la cabecera `usr.toAgent` no está establecida, el valor predeterminado que se va a utilizar para el agente de destino es AGENT_MAGENTA.

Cuando especifica `useGroups="true"`, si no especifica también `groupId="${GROUPID}"`, la transferencia sólo toma el primer mensaje de la cola. Por ejemplo, si está utilizando la sustitución de variables para generar el nombre de archivo (fileName), es posible que el contenido de a `.txt` no sea correcto. Esto se debe a que el nombre de archivo (fileName) lo genera el supervisor, pero la transferencia obtiene realmente un mensaje que no es el que debe generar el archivo denominado fileName.

Procedimiento

1. Cree el XML de tarea que define la tarea que el supervisor lleva a cabo cuando se desencadena.

```
<?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="AGENT_VENUS" QMgr="QM_VENUS" />
     <destinationAgent agent="${toAgent}" QMgr="QM_MARS" />
     <transferSet>
       <item mode="binary" checksumMethod="none">
         <source>
           <queue useGroups="true" groupId="${GROUPID}">START_QUEUE</queue>
         </source>
         <destination type="file" exist="overwrite">
           <file>/reports/${fileName}.rpt</file>
         </destination>
       </item>
     </transferSet>
   </managedTransfer>
</request>
```

   Las variables que se sustituyen por los valores de las cabeceras de mensajes de IBM MQ se resaltan en **negrita**. Este XML de tarea se guarda en el archivo `/home/USER1/task.xml`

2. Cree un supervisor de recursos para supervisar la cola START_QUEUE.

   Someta el siguiente mandato:

```
fteCreateMonitor -ma AGENT_VENUS -mm QM_VENUS -mq START_QUEUE
                 -mn myMonitor -mt /home/USER1/task.xml
                 -tr completeGroups -pi 30 -pu minutes -dv toAgent=AGENT_MAGENTA
```

3. Un usuario o un programa graba un grupo de mensajes en la cola START_QUEUE.

   El primer mensaje de este grupo cuenta con las siguientes propiedades de mensajes de IBM MQ establecidas:

```
usr.fileName=larmer
usr.toAgent=AGENT_VIOLET
```

4. El supervisor se desencadena cuando se ha grabado el grupo completo. El agente sustituye las propiedades de mensajes de IBM MQ en el XML de tarea.

   El resultado es la transformación del XML de tarea en:

```
<?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="AGENT_VENUS" QMgr="QM_VENUS" />
     <destinationAgent agent="AGENT_VIOLET" QMgr="QM_MARS" />
     <transferSet>
       <item mode="binary" checksumMethod="none">
         <source>
           <queue useGroups="true" groupId="${GROUPID}">START_QUEUE</queue>
         </source>
         <destination type="file" exist="overwrite">
           <file>/reports/larmer.rpt</file>
         </destination>
       </item>
     </transferSet>
   </managedTransfer>
</request>
```

Resultados

Se realiza la transferencia definida por el XML de tarea. El grupo completo de mensajes leídos desde START_QUEUE por AGENT_VENUS se escribe en un archivo denominado `/reports/larmer.rpt` en el sistema donde AGENT_VIOLET está en ejecución.

Qué hacer a continuación

Transferir cada mensaje a un archivo distinto

Si desea supervisar una cola y hacer que cada mensaje se transfiera a un archivo independiente, puede utilizar una técnica similar a la que se ha descrito anteriormente en este tema.

1. Cree el supervisor tal como se ha descrito anteriormente, especificando el parámetro **-tr completeGroups** en el mandato **fteCreateMonitor** .
2. En el XML de la tarea, especifique lo siguiente:

```
<queue useGroups="true" groupId="${GROUPID}">START_QUEUE</queue>
```

Sin embargo, cuando coloque los mensajes en la cola de origen, no los coloque en un grupo de IBM MQ. Añada propiedades de mensaje de IBM MQ a cada mensaje. Por ejemplo, especifique la propiedad `usr.filename` con un valor de nombre de archivo exclusivo para cada mensaje. Esto hace que el Managed File Transfer Agent trate cada mensaje en la cola de origen como un grupo aparte.

Conceptos relacionados

“Transferir datos de mensajes a archivos” en la página 285
La función de mensaje a archivo de Managed File Transfer le permite transferir datos de uno o más mensajes en una cola de IBM MQ a un archivo , un conjunto de datos (en z/OS) o un espacio de archivos de usuario. Si tiene una aplicación que crea o procesa mensajes de IBM MQ, puede utilizar la función de mensaje a archivo de Managed File Transfer para transferir estos mensajes a un archivo en cualquier sistema de la red de Managed File Transfer.

“Personalización de las tareas del supervisor de recursos de MFT con sustitución de variables” en la página 246
Cuando se cumplen las condiciones desencadenantes de un supervisor de recursos activo, se llama a la tarea definida. Además de llamar a la tarea de transferencia o mandato con el mismo agente de destino o el mismo nombre de archivo de destino cada vez, también puede modificar la definición de

tarea durante la ejecución. Para ello, inserte los nombres de variables en el XML de definición de tarea. Cuando el supervisor determina que se cumplen las condiciones de desencadenante y que la definición de tarea contiene nombres de variable, sustituye los nombres de variable por los valores de variable y, a continuación, llama a la tarea.

Qué hacer si los archivos de destino creados por una transferencia iniciada por un supervisor de recursos de cola contienen datos incorrectos

Tareas relacionadas

"Configuración de tareas de supervisión de MFT para iniciar mandatos y scripts" en la página 239
Los supervisores de recursos no se limitan a realizar transferencias de archivos y la tarea asociada. También puede configurar el supervisor para invocar otros mandatos desde el agente de supervisión, incluidos los programas ejecutables, los scripts Ant o los trabajos JCL. Para llamar a los mandatos, edite el formato XML de definición de tarea del supervisor para incluir uno o varios elementos de mandatos con los parámetros de llamada de mandatos correspondientes, como por ejemplo, argumentos y propiedades.

"Ejemplo: configuración de un recurso MFT" en la página 245
Puede especificar una cola de IBM MQ como el recurso que va a supervisar un supervisor de recursos utilizando el parámetro **-mq** con el mandato **fteCreateMonitor**.

Referencia relacionada

fteCreateMonitor: crear un supervisor de recursos de MFT

Propiedades de mensajes de MQ leídas por MFT de mensajes en las colas de origen

Configuración del comportamiento de reintentos del supervisor para transferencias de mensaje a archivo

Si una transferencia de mensaje a archivo que ha desencadenado un supervisor de recursos falla y deja el grupo de mensajes que desencadenaron el supervisor en la cola, dicha transferencia se vuelve a enviar en intervalos de sondeos posteriores. El número de veces que la transferencia se vuelve a enviar está limitada por la propiedad **monitorGroupRetryLimit** del agente de supervisión.

Acerca de esta tarea

Cada vez que se desencadena una transferencia de mensaje a archivo, se genera un nuevo ID de transferencia para la tarea de transferencia.

Si se reinicia el agente, el supervisor vuelve a desencadenar una transferencia aunque el número de veces que se ha desencadenado la transferencia haya superado el valor de **monitorGroupRetryLimit** en el archivo `agent.properties`. El valor de la propiedad **monitorGroupRetryLimit** es el número máximo de veces que un supervisor vuelve a desencadenar una transferencia de mensaje a archivo si el grupo de mensajes sigue existiendo en la cola. El valor predeterminado de esta propiedad es 10. El valor de esta propiedad se puede establecer en cualquier valor entero positivo o -1. Si se especifica el valor -1 para esta propiedad, el supervisor vuelve a desencadenar la transferencia un número ilimitado de veces, hasta que no se cumpla la condición desencadenante.

Si un intento de transferencia hace que el número de veces que se ha desencadenado la transferencia supere el valor de **monitorGroupRetryLimit**, el agente escribe un error en el registro de sucesos.

Un solo mensaje se trata como si fuera un solo grupo y la transferencia se vuelve a desencadenar en cada intervalo de sondeo mientras el mensaje permanece en la cola y el número de veces que la transferencia se ha desencadenado sea inferior al valor de **monitorGroupRetryLimit**.

Para establecer la propiedad **monitorGroupRetryLimit** en el agente de supervisión, realice los pasos siguientes:

Procedimiento

1. Detenga el agente de supervisión utilizando el mandato **fteStopAgent**.
2. Edite el archivo `agent.properties` para que el agente de supervisión incluya la línea siguiente:

```
monitorGroupRetryLimit=number_of_retries
```

El archivo `agent.properties` se encuentra en el directorio *MQ_DATA_PATH*/mqft/config/*coordination_qmgr_name*/agents/*monitoring_agent_name*.

3. Inicie el agente de supervisión utilizando el mandato **fteStartAgent**.

Tareas relacionadas

“Ejemplo: configuración de un recurso MFT” en la página 245

Puede especificar una cola de IBM MQ como el recurso que va a supervisar un supervisor de recursos utilizando el parámetro **-mq** con el mandato **fteCreateMonitor**.

Utilización de un archivo desencadenante

Puede utilizar el contenido de un archivo desencadenante en un supervisor de recursos para definir un conjunto de archivos que deben transferirse en una única solicitud de transferencia. Cada vez que se detecte un archivo desencadenante coincidente, su contenido se analizará en busca de vías de acceso de archivo de origen y, opcionalmente, vías de acceso de archivo de destino. Estas vías de acceso de archivo se utilizarán para definir elementos del archivo XML de transferencia de tarea que especifique, que se someterá como una única solicitud de transferencia al agente. La definición del supervisor de recursos determina si el contenido del desencadenante está habilitado.

Puede habilitar el desencadenamiento del contenido de archivo cuando cree un supervisor especificando el parámetro **-tc** (contenido de desencadenante). Este parámetro **-tc** se aplica solamente a las opciones de desencadenante de archivo `match` y `noSizeChange`. Si desea más información sobre cómo crear un supervisor, consulte **fteCreateMonitor**: crear un supervisor de recursos MFT.

Cuando se utiliza un archivo de contenido desencadenante, el formato predeterminado de cada línea es:

- Una única vía de acceso de archivo de origen o
- Una vía de acceso de archivo de origen y una vía de acceso de archivo de destino, separadas por una coma

donde los caracteres de espacio en blanco se manejan como parte de las vías de acceso de los archivos. Es posible cambiar el formato de línea predeterminado especificando los parámetros **-tcr** y **-tcc** en el mandato **fteCreateMonitor**. Para obtener más información, consulte “Opciones avanzadas” en la página 256.

Tras analizar un archivo desencadenante, se genera y aplica una lista de vías de acceso de archivo al XML de tarea de transferencia que haya especificado. Tal como ocurre con todos los supervisores, el formato del XML de tarea de transferencia es un XML de tarea de transferencia completo, que genera el mandato **fteCreateTransfer** con un único elemento o archivo definido. El elemento único debe utilizar las variables de sustitución ${contentSource} y, opcionalmente, ${contentDestination}, como reemplazo de las vías de acceso de archivo de origen y destino. El supervisor expande el XML de tarea de transferencia para que incluya un elemento de archivo para cada línea (vía de acceso de archivo) del archivo desencadenante.

No puede utilizar el desencadenamiento del contenido con el parámetro **-bs** porque el parámetro **-tc** implica una solicitud de transferencia para cada archivo desencadenante.

Ejemplo

El ejemplo siguiente define un supervisor para desencadenar en un archivo que finaliza en `trig` y lee las vías de acceso de archivo en dicho archivo.

```
fteCreateTransfer -gt task.xml -sa SrcAgent -da DestAgent -dd /file/destdir ${contentSource}
fteCreateMonitor -mn TrigMonitor -md /home/trigdir -mt task.xml -ma SrcAgent -tr "match,*.trig"
-tc
```

El mandato **fteCreateTransfer** crea un archivo que se denomina `task.xml` para un único archivo con una vía de acceso de archivo de origen de ${contentSource}. Por ejemplo:

```
<item checksumMethod="MD5" mode="binary">
    <source disposition="leave" recursive="false">
          <file>${contentSource}</file>
    </source>
</item>
```

El mandato **fteCreateMonitor** explora en busca de archivos que terminen en `trig`, en el directorio `/home/trigdir`, y utiliza el contenido para crear una única solicitud de transferencia basada en el archivo `task.xml` para todas las vías de acceso de dicho archivo desencadenante. El formato del archivo desencadenante debe ser una vía de acceso de archivo (sólo de origen) en cada línea, sin separador de coma. Por ejemplo:

```
/home/file/first.txt
/home/file/second.txt
/home/different/third.txt
⋮
```

Todos los archivos se entregan en el directorio `/file/destdir` con su nombre de archivo y no su vía de acceso de archivo, es decir, `/home/file/first.txt` se entrega en `/file/destdir/first.txt`.

De forma alternativa, si cambia el parámetro **-dd /file/destdir** en el mandato **fteCreateTransfer** a `-df ${contentDestination}` y el formato del contenido de un archivo desencadenante a *vía de acceso de archivo de origen*,*vía de acceso de archivo de destino*, puede definir distintas vías de acceso de destino para el mismo agente de destino. Por ejemplo:

```
/home/file/first.txt,/home/other/sixth.txt
```

La ubicación de destino se convierte en `/home/other/sixth.txt`.

Las variable de sustitución pueden señalizarse. Por ejemplo, puede separar la parte del nombre de archivo de la vía de acceso proporcionada utilizando ${contentDestination{token=-1}}. Por lo tanto, si el destino **fteCreateTransfer** se define como `-df /file/destdir/${contentDestination{token=-1}}`, el nuevo destino para `/home/file/first.txt` es `/file/destdir/sixth.txt`.

Opciones avanzadas

Puede cambiar el formato de línea predeterminado para el contenido del archivo desencadenante, mediante el parámetro **-tcr** *regex*. Proporcione una expresión regular que coincida con el formato de línea pertinente y que proporciona uno o dos grupos de captura. El primer grupo de captura es el origen, y el segundo grupo de captura, opcional, es el destino. Por ejemplo:

- Las vías de acceso de origen y de destino se separan mediante un guión:

  ```
  ((?:[^-])+)-((?:[^-])+)
  ```

  En este ejemplo, el separador se define en tres ubicaciones, y las tres instancias del guión, -, pueden cambiarse por cualquier otro carácter. Asegúrese de añadir un carácter de escape a los caracteres especiales que aparezcan.
- Las vías de acceso de origen y de destino se separan mediante una coma con espacios al final. Los comentarios que se indiquen mediante un signo de almohadilla (#) se ignoran.

  ```
  ((?:[^,])+),((?:[^,])+) *(?:#.*)+
  ```

  Las vías de acceso de archivo no pueden contener el signo de almohadilla (#). Normalmente, una entrada es la siguiente: `/home/source/from.txt,/home/destination/to.txt # some comment`.

Si utiliza el parámetro **-tcr**, asegúrese de que la expresión regular se haya diseñado y probado bien, de forma que la expresión pueda detectar errores y analizar correctamente los archivos desencadenantes.

Puede invertir el orden de la captura, mediante el parámetro **-tcc destSrc**. Si especifica este parámetro, el primer grupo de captura es la vía de acceso del archivo de destino, y el segundo grupo es la vía de acceso del archivo de origen.

Cómo se manejan los errores

Archivo desencadenante vacío
Si el archivo desencadenante está vacío, el resultado es que no se realiza ninguna transferencia de archivos. Es decir, el supervisor crea una solicitud de transferencia pero no se ha especificado ningún elemento de archivo.

Archivo desencadenante con errores
Si alguna entrada de un archivo desencadenante no puede analizarse en relación al formato esperado, no se generará ninguna solicitud de transferencia. Se publicará un registro de errores del supervisor, y el error también se registrará en el registro de errores. El archivo desencadenante se marca como procesado, y el supervisor no trata de volver a procesar el archivo hasta que se haya actualizado el archivo.

XML de tarea de transferencia no coincidente
El XML de la tarea de transferencia debe coincidir con el archivo desencadenante, es decir, si el XML de la tarea de transferencia tiene ${contentSource} y ${contentDestination}, todos los archivos desencadenantes para dicho supervisor deben tener las vías de acceso de los archivos de origen y destino y lo mismo ocurre a la inversa. En el primer caso, el supervisor informa de un error de sustitución de ${contentDestination} si el archivo desencadenante sólo proporciona la vía de acceso del archivo de origen.

Ejemplos

El ejemplo siguiente es un desencadenante de contenido básico en que el contenido de un archivo desencadenante tiene sólo una vía de acceso de archivo de origen.

```
fteCreateTransfer -gt task.xml -sa SrcAgent -da DestAgent -dd /file/destdir ${contentSource}
fteCreateMonitor -mn TrigMonitor -md /home/trigdir -mt task.xml -ma SrcAgent -tr "match,*.trig"
-tc
```

El parámetro **-tcr** define dos grupos de captura de una secuencia de cualquier número de caracteres que estén separados por un carácter de espacio. El parámetro y la opción **-tcc destSrc** indican que los grupos de captura deben procesarse primero como destino y, después, como origen.

```
fteCreateTransfer -gt task.xml -sa SrcAgent -da DestAgent -df ${contentDestination} $
{contentSource}
fteCreateMonitor -mn TrigMonitor -md /home/trigdir -mt task.xml -ma SrcAgent -tr "match,*.trig"
-tc
    -tcr "((?:[^ ])+) ((?:[^ ])+)" -tcc destSrc
```

Conceptos relacionados

"Personalización de las tareas del supervisor de recursos de MFT con sustitución de variables" en la página 246
Cuando se cumplen las condiciones desencadenantes de un supervisor de recursos activo, se llama a la tarea definida. Además de llamar a la tarea de transferencia o mandato con el mismo agente de destino o el mismo nombre de archivo de destino cada vez, también puede modificar la definición de tarea durante la ejecución. Para ello, inserte los nombres de variables en el XML de definición de tarea. Cuando el supervisor determina que se cumplen las condiciones de desencadenante y que la definición de tarea contiene nombres de variable, sustituye los nombres de variable por los valores de variable y, a continuación, llama a la tarea.

Tareas relacionadas

"Supervisión de una cola y utilización de sustitución de variables" en la página 251

Puede supervisar una cola y transferir mensajes de la cola supervisada a un archivo utilizando el mandato **fteCreateMonitor**. El valor de cualquier propiedad de mensaje de IBM MQ en el primer mensaje que se va a leer de la cola supervisada se puede sustituir en la definición XML de la tarea y se utiliza para definir el comportamiento de la transferencia.

Referencia relacionada

fteCreateMonitor: crear un supervisor de recursos de MFT

fteCreateTransfer: iniciar una nueva transferencia de archivos

Registro de supervisores de recursos de MFT

Puede obtener información de diagnóstico sobre supervisores de recursos utilizando el registro.

Acerca de esta tarea

Puede utilizar el registro para supervisores de recursos utilizando el mandato **fteSetAgentLoglevel** o el archivo `agent.properties` para controlar el registro del supervisor de recursos.

Tenga en cuenta que los puntos de rastreo existentes todavía están en uso para capturar la información.

Los registros del supervisor de recursos se escriben en un archivo denominado `resmonevent`*N*`.log`, donde *N* significa un número; por ejemplo, `resmonevent0.log`. Los archivos de registro de sucesos registran varias acciones que tienen lugar cuando un supervisor sondea un recurso, por ejemplo, un directorio o una cola.

Atención: Todos los supervisores de recursos de un agente graban en el mismo archivo de registro.

Para obtener alguna salida de ejemplo de un archivo `resmonevent`*N*`.log`, consulte Qué hacer si el supervisor de recursos de directorio MFT no está desencadenando archivos.

En la tabla siguiente se enumeran los tipos de sucesos que el supervisor de recursos graba en el archivo de registro. La tercera columna describe el nivel de registro necesario para capturar cada suceso donde el nivel inferior es INFO y el nivel superior es VERBOSE.

Tenga en cuenta que establecer un nivel de registro elevado, graba también sucesos de nivel inferior. Por ejemplo, al establecer el nivel de registro en MODERATE se graban también los sucesos de nivel INFO, pero no los sucesos de nivel VERBOSE.

Número	Suceso	Nivel de registro	Descripción
1	Supervisor creado	INFO	Se ha creado un supervisor de recursos.
2	Supervisor suprimido	INFO	Se ha suprimido un supervisor de recursos.
3	Supervisor detenido	INFO	Se ha detenido un supervisor de recursos.
4	Supervisor iniciado	INFO	Se ha iniciado un supervisor de recursos.
5	Iniciar sondeo	INFO	Un supervisor de recursos ha iniciado un nuevo ciclo de sondeo.
6	Finalizar sondeo	INFO	Un ciclo de sondeo del supervisor de recursos ha finalizado.
7	Coincidencia de patrón	VERBOSE	Se ha encontrado un archivo en el directorio del supervisor de desencadenante o un mensaje en una cola que coincide con el patrón especificado.
8	Discrepancia de patrón	VERBOSE	Se ha encontrado un archivo en el directorio del supervisor de desencadenante o un

Número	Suceso	Nivel de registro	Descripción
			mensaje en una cola que no coincide con el patrón especificado.
9	Solicitud de transferencia	INFO	El supervisor de recursos ha iniciado una transferencia.
10	Directorio demasiado profundo	VERBOSE	El directorio supervisado por el supervisor de recursos contiene más subdirectorios que sondear que el número especificado en la configuración del supervisor de recursos.
11	Archivo bloqueado	MODERATE	Otro proceso ha bloqueado el archivo desencadenante supervisado por el supervisor de recursos.
12	Tamaño de archivo pequeño	MODERATE	El archivo desencadenante tiene un tamaño menor que el especificado en la configuración del supervisor de recursos.
13	Tamaño de archivo inestable	MODERATE	El archivo desencadenante cambia con más frecuencia de lo que esperaba la configuración del supervisor de recursos.
14	Demasiados sondeos	MODERATE	Un supervisor de recursos ha sondeado un archivo desencadenante inestable demasiadas veces.
15	Elementos coincidentes	INFO	Número total de archivos desencadenantes encontrados en el directorio sondeado por un supervisor de recursos.
16	Transferir elementos	INFO	Número total de elementos en la solicitud de transferencia.
17	FDC generado	MODERATE	Un supervisor de recursos ha generado una excepción.
18	Solicitud de transferencia	INFO	El supervisor de recursos ha enviado una solicitud de transferencia.
19	Error de inicio de supervisor	MODERATE	No se ha podido iniciar un supervisor de recursos.
20	Historial borrado	INFO	Se ha borrado la información del historial del supervisor.
21	Error al borrar historial del supervisor	INFO	Ha fallado el intento de borrar la información del historial del supervisor.
22	ID de transferencia	INFO	El supervisor ha enviado el ID de la solicitud de transferencia.
23	Proceso por lotes	INFO	Número total de solicitudes de transferencia para elementos coincidentes: *N*, donde *N* es un Número.
V 9.4.0 24	Conectado	VERBOSE	Un supervisor de recursos se ha conectado al gestor de colas del agente.
V 9.4.0 25	Desconectado	VERBOSE	Un supervisor de recursos se ha desconectado del gestor de colas del agente.

Número	Suceso	Nivel de registro	Descripción
V 9.4.0 26	Error durante la desconexión	VERBOSE	Un supervisor de recursos ha encontrado un problema al desconectarse del gestor de colas del agente.

Procedimiento

- Para utilizar el mandato **fteSetAgentLoglevel** para activar y desactivar el registro del supervisor de recursos, consulte Nivel defteSetAgentLog para obtener una descripción del parámetro **logMonitor** y ejemplos de cómo utilizar las distintas opciones.
- Para utilizar el archivo `agent.properties` para controlar el registro del supervisor de recursos, consulte El archivo MFT `agent.properties` para obtener una descripción de las propiedades adicionales que le permiten realizar las siguientes actividades de registro:
  - Activar o desactivar el registro
  - Limitar el tamaño de cada archivo de registro
  - Limitar el número de registros que los supervisores de recursos pueden generar

Ejemplo

El siguiente mensaje de ejemplo establece el registro de nivel `verbose` para el agente HA2, en el gestor de colas MFTDEMO:

```
<?xml version="1.0"?>
<log:log version="6.00"
         xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
         xmlns:log="https://www.ibm.com/log">
    <log:originator>
      <log:request>
        <log:hostName>192.168.7.1</log:hostName>
        <log:userID>johndoe</log:userID>
      </log:request>
    </log:originator>
    <log:endpoint agent="HA2" QMgr="MFTDEMO"/>
    <log:logMonitor>MON1="verbose"</log:logMonitor>
</log:log>
```

Referencia relacionada

Mandato fteSetAgentLogLevel

El archivo MFT `agent.properties`

Iniciar un supervisor de recursos de MFT

Desde IBM MQ 9.3.0, puede iniciar los supervisores de recursos sin tener que detener o reiniciar un agente, utilizando el mandato **fteStartMonitor**.

Antes de empezar

Si la gestión de autorizaciones de usuario se ha habilitado estableciendo el atributo **authorityChecking** en `true` en el archivo `agent.properties`, debe tener la autorización `Supervisor` u `Operaciones de supervisión` para iniciar un supervisor de recursos. Para obtener más información sobre la gestión de autorizaciones de usuario, consulte Restricción de las autorizaciones de usuario en las acciones del agente de MFT.

Acerca de esta tarea

Puede ejecutar el mandato **fteStartMonitor** desde cualquier sistema donde esté instalado el componente de mandatos de Managed File Transfer, lo cual significa que puede iniciar un recurso desde

cualquier lugar y no está restringido al sistema en el que el agente es el propietario del recurso en ejecución. Para obtener información sobre los parámetros obligatorios y opcionales de este mandato, consulte fteStartMonitor (iniciar un supervisor de recursos de MFT).

Procedimiento

- Para averiguar el estado de un agente antes de ejecutar el mandato **fteStartMonitor**, utilice el mandato **fteListMonitors** con el parámetro **-v**, tal como se muestra en el ejemplo siguiente:

  ```
  fteListMonitors -ma monitoring_agent_name -v
  ```

- Para iniciar un supervisor de recursos de un agente que se ejecuta en la misma máquina, especifique el mandato **fteStartMonitor** de la siguiente manera:

  ```
  fteStartMonitor -mn monitor_name -ma agent_name
  ```

- Para iniciar un supervisor de recursos de un agente que se ejecuta en otra máquina, especifique el mandato **fteStartMonitor** de la siguiente manera:

  ```
  fteStartMonitor -mn monitor_name -ma agent_name -mm AgentQueueManager
  ```

  Si el gestor de colas de mandatos es también el gestor de colas de agente del agente de supervisión, entonces el parámetro **-mm** es opcional, de lo contrario, debe especificar el gestor de colas de agente con el parámetro **-mm**.

Resultados

Si el agente está en ejecución, el supervisor de recursos se inicia si está detenido en este momento. El mandato genera los siguientes mensajes y registra un suceso en el `output0.log`del agente.

```
BFGCL0816I: Se ha emitido una solicitud para iniciar el supervisor de recursos
'nombre_supervisor' del agente 'nombre_agente'.
BFGCL0251I: La solicitud se ha completado satisfactoriamente.
```

Para obtener información sobre los mensajes que el mandato genera si no puede iniciar el supervisor de recursos, consulte fteStartMonitor (iniciar un supervisor de recursos de MFT).

Conceptos relacionados

“Conceptos de supervisión de recursos de MFT” en la página 234
Una visión general de los conceptos clave de la característica de supervisión de recursos de Managed File Transfer.

Tareas relacionadas

“Detener un supervisor de recursos de MFT” en la página 261
Desde IBM MQ 9.3.0, puede detener los supervisores de recursos sin tener que detener o reiniciar un agente, utilizando el mandato **fteStopMonitor**.

Referencia relacionada

fteStartMonitor (iniciar un supervisor de recursos de MFT)

Detener un supervisor de recursos de MFT

Desde IBM MQ 9.3.0, puede detener los supervisores de recursos sin tener que detener o reiniciar un agente, utilizando el mandato **fteStopMonitor**.

Antes de empezar

Si la gestión de autorizaciones de usuario se ha habilitado estableciendo el atributo **authorityChecking** en `true` en el archivo `agent.properties`, debe tener la autorización `Supervisor` u `Operaciones de supervisión` para detener un supervisor de recursos. Para obtener más información sobre la gestión de autorizaciones de usuario, consulte Restricción de las autorizaciones de usuario en las acciones del agente de MFT.

Acerca de esta tarea

Puede ejecutar el mandato **fteStopMonitor** desde cualquier sistema donde esté instalado el componente de mandatos de Managed File Transfer, lo cual significa que puede detener un recurso desde cualquier lugar y no está restringido al sistema en el que el agente es el propietario del recurso en ejecución. Para obtener información sobre los parámetros obligatorios y opcionales de este mandato, consulte fteStopMonitor (detener un supervisor de recursos de MFT).

Cuando se detiene un supervisor de recursos, escribe un mensaje en el registro de sucesos del supervisor de recursos del agente, `resmoneventnumber.log`. Si el supervisor de recursos se ha detenido con el mandato **fteStopMonitor**, el mensaje incluye el nombre del usuario que ha emitido la solicitud de detención:

```
Supervisor de recursos detenido por el usuario '<mquser_id>'
```

Si el agente se reinicia, se inicia automáticamente un supervisor de recursos, incluso si el supervisor de recursos se había detenido anteriormente utilizando el mandato **fteStopMonitor**.

Los agente procesan las solicitudes de detención del supervisor en serie en lugar de en paralelo, de manera que, por ejemplo, si un agente recibe una solicitud para detener el supervisor M1 y rápidamente recibe otra solicitud para detener el supervisor M2, detendrá primero M1 antes de intentar detener M2.

Procedimiento

- Para averiguar el estado de un agente antes de ejecutar el mandato **fteStopMonitor**, utilice el mandato **fteListMonitors** con el parámetro **-v**, tal como se muestra en el ejemplo siguiente:

  ```
  fteListMonitors -ma monitoring_agent_name -v
  ```

- Para detener un supervisor de recursos de un agente que se ejecuta en la misma máquina, especifique el mandato **fteStopMonitor** de la siguiente manera:

  ```
  fteStopMonitor -mn monitor_name -ma agent_name
  ```

- Para detener un supervisor de recursos de un agente que se ejecuta en otra máquina, especifique el mandato **fteStopMonitor** de la siguiente manera:

  ```
  fteStopMonitor -mn monitor_name -ma agent_name -mm AgentQueueManager
  ```

  Si el gestor de colas de mandatos es también el gestor de colas de agente del agente de supervisión, entonces el parámetro **-mm** es opcional, de lo contrario, debe especificar el gestor de colas de agente con el parámetro **-mm**.

Resultados

Si el agente está en ejecución, el supervisor de recursos se detiene si está iniciado en este momento. El mandato genera los siguientes mensajes y registra un suceso en el `output0.log`del agente.

```
BFGCL0813I: Se ha intentado detener el supervisor de recursos 'MNTR' del agente 'SOURCE'.
BFGCL0251I: La solicitud se ha completado satisfactoriamente.
```

Para obtener información sobre los mensajes que el mandato genera si no puede detener el supervisor de recursos, consulte fteStopMonitor (detener un supervisor de recursos de MFT).

Conceptos relacionados

“Conceptos de supervisión de recursos de MFT” en la página 234
Una visión general de los conceptos clave de la característica de supervisión de recursos de Managed File Transfer.

Tareas relacionadas

“Iniciar un supervisor de recursos de MFT” en la página 260
Desde IBM MQ 9.3.0, puede iniciar los supervisores de recursos sin tener que detener o reiniciar un agente, utilizando el mandato **fteStartMonitor**.

Referencia relacionada

fteStopMonitor (detener un supervisor de recursos de MFT)

Copia de seguridad y restauración de supervisores de recursos de MFT

Puede hacer copia de seguridad de los supervisores de recursos que desea tener disponibles para su uso futuro exportando sus definiciones a un archivo XML que puede importar después para crear un nuevo supervisor de recursos desde la copia de seguridad.

Acerca de esta tarea

Puede que necesite hacer copia de seguridad de los supervisores de recursos definidos anteriormente para poder volver a utilizar sus definiciones en el futuro, por ejemplo, para volver a crear los supervisores de recursos en otra infraestructura o si se debe crear un supervisor de recursos debido a problemas del gestor de colas.

Puede hacer copia de seguridad de una definición única del gestor de recursos utilizando el mandato **fteCreateMonitor** o el mandato **fteListMonitors** con el parámetro **-ox**. En ambos casos, se hace copia de seguridad de la definición del gestor de recursos exportándola a un archivo XML. Luego se puede utilizar el parámetro **-ix** del mandato **fteCreateMonitor** para crear un nuevo gestor de recursos importando la definición del archivo XML.

Con el parámetro **-ox**, sólo puede realizar una copia de seguridad de una definición de supervisor de recursos a la vez.

Se ha añadido el parámetro **-od** al mandato **fteListMonitors**. Al especificar este parámetro, puede hacer copia de seguridad de más de un supervisor de recursos a la vez exportando sus definiciones de forma masiva a un directorio especificado. Cada definición de supervisor de recursos se guarda en un archivo XML independiente con un nombre en el formato *agent name*.*monitor name*.xml.

El parámetro **-od** es particularmente útil si tiene un gran número de supervisores de recursos de los que desea hacer copia de seguridad porque debe ejecutar el mandato **fteListMonitors -od** solo una vez, en lugar de tener que ejecutar el mandato **fteListMonitors -ox** por separado para cada definición de recurso o utilizar un script independiente para ejecutar el mandato **fteListMonitors -ox** para cada supervisor de recursos.

Procedimiento

- Para hacer copia de seguridad de la definición de un supervisor de recursos exportándolo a un archivo XML, utilice cualquiera de los mandatos siguientes:
  - El mandato **fteCreateMonitor** con el parámetro **-ox**.
  - El mandato **fteListMonitors** con el parámetro **-ox**.

  Cuando utilice el parámetro **-ox**, también debe especificar los parámetros **-ma** y **-mn**, tal como se muestra en el ejemplo siguiente:

  ```
  fteListMonitors -ma AGENT1 -mn MONITOR1 -ox filename1.xml
  ```

- Para realizar una copia de seguridad de varias definiciones de supervisor de recursos exportándolas a archivos XML en un directorio especificado, utilice el mandato **fteListMonitors** con el parámetro **-od** tal como se muestra en el ejemplo siguiente:

  ```
  fteListMonitors -od /usr/mft/resmonbackup
  ```

  Debe especificar un directorio de destino válido cuando hace copia de seguridad de los supervisores de recursos de forma masiva. Si no se especifica una vía de acceso de destino se genera un mensaje de error como se muestra en el ejemplo siguiente:

  ```
  BFGCL0762E: No se ha especificado el directorio de salida. Vuelva a ejecutar el mandato
  especificando una vía de acceso válida.
  ```

El parámetro **-od** no se debe combinar con el parámetro **-ox**; de lo contrario, se visualiza el siguiente mensaje de error:

```
BFGCL0761E: No es válido especificar los dos parámetros '-od' y '-ox' juntos.
```

Puede definir un determinado conjunto de supervisores de recursos para incluir en la copia de seguridad. Por ejemplo, utilizando el parámetro **-ma** para especificar el nombre de un agente, puede realizar una copia de seguridad de todos los supervisores de recursos para dicho agente, tal como se muestra en el ejemplo siguiente:

```
fteListMonitors -ma AGENT1 -od /usr/mft/resmonbackup
```

Puede utilizar también la coincidencia de comodín incluyendo un carácter de asterisco (*) al definir un patrón que se va a utilizar para la coincidencia de nombres de agente o nombres de patrón, o ambos. En el ejemplo siguiente se hace copia de seguridad de todos los supervisores de recursos que tienen nombres que coinciden con un patrón especificado y que se encuentran en un agente con un nombre que coincide con un patrón especificado:

```
fteListMonitors -ma AGENT* -mn MON* -od /usr/mft/resmonbackup
```

Cuando el mandato se ejecuta, se muestran los mensajes de informe de progreso siguientes:

```
Se ha encontrado un total de número definiciones de supervisor de recursos coincidentes.
Se han guardado índice de número definiciones de supervisor de recursos en el sistema de
archivos.
```

Si utiliza la opción de información detallada, el total acumulado se sigue mostrando, pero en lugar de mostrar

```
Se han guardado índice de número definiciones de supervisor de recursos en el sistema de
archivos
```

el mandato muestra el nombre de la definición de supervisor guardada, por ejemplo:

```
BFGCL0762I: Definición del supervisor 'FILEMON' del agente 'XFERAGENT' guardada como
FILEMON.XFERAGENT.XML al sistema de archivos.
```

- Para realizar una copia de seguridad de un supervisor de recursos para un agente determinado exportándolo a un archivo XML en un directorio especificado, utilice el mandato **fteListMonitors** con el parámetro **-od** :

  ```
  fteListMonitors -ma AGENT1 -mn MONITOR1 -od /usr/mft/resmonbackup
  ```

  La utilización del parámetro **-od** para realizar una copia de seguridad de un único supervisor de recursos es similar a la utilización del parámetro **-ox**, excepto que el nombre del archivo de salida está en el formato *agent name*.*monitor name*.xml.
- Para restaurar definiciones de supervisor de recursos de una copia de seguridad, utilice el mandato **fteCreateMonitor** con el parámetro **-ix** como se muestra en el ejemplo siguiente:

  ```
  fteCreateMonitor -ix file name
  ```

  Para obtener más ejemplos de cómo utilizar el parámetro **-od**, consulte fteListMonitors: listar supervisores de recursos de MFT.

Referencia relacionada

fteCreateMonitor: crear un supervisor de recursos de MFT

fteListMonitors: listar supervisores de recursos de MFT

Borrado del historial del supervisor de recursos

Puede borrar el historial de un supervisor de recursos para que se pueda enviar otra solicitud de transferencia de archivo para un archivo que no se transfirió anteriormente debido a una anomalía. Para borrar el historial del supervisor de recursos, puede utilizar el mandato **fteClearMonitorHistory** o IBM MQ Explorer.

Antes de empezar

Si la gestión de autorizaciones de usuario se ha habilitado estableciendo el atributo **authorityChecking** en `true` en el archivo `agent.properties`, el usuario que borra el historial del supervisor debe tener la autorización adecuada, tal como se muestra en la tabla siguiente.

*Tabla 15. Autorización de usuario necesaria para ejecutar el mandato **fteClearMonitorHistory***

Usuario que borra el historial de supervisión	Autorización de acceso de MFT	Autorización necesaria
El mismo usuario que el que ha creado el supervisor de recursos.	Supervisor	BROWSE on SYSTEM.FTE.AUTHMON1.<*monitor_agent_name*> Se trata de la misma autorización que la autorización necesaria para crear o suprimir el supervisor de recursos.
Cualquier usuario que no sea el usuario que ha creado el supervisor de recursos.	Operaciones de supervisión	SET on SYSTEM.FTE.AUTHPS1.<*agent_name*> Se trata de la misma autorización que la autorización necesaria para suprimir el supervisor de recursos.

Para obtener más información sobre la gestión de autorizaciones de usuario, consulte Restricción de las autorizaciones de usuario en las acciones del agente de MFT.

Si un usuario sin la autorización necesaria intenta borrar el historial del supervisor de recursos, el mandato **fteClearMonitorHistory** genera un mensaje de error y registra la anomalía en el archivo `output0.log` del agente. Para obtener más información, consulte fteClearMonitorHistory: borrar el historial de supervisor de recursos.

Acerca de esta tarea

Si se ha iniciado una transferencia de archivos y no se puede transferir un archivo por alguna razón, el supervisor de recursos no selecciona este archivo para transferirlo de nuevo en su siguiente sondeo porque el historial del supervisor indica que el archivo se ha visto en un sondeo anterior y no se ha modificado desde entonces (consulte "Conceptos de supervisión de recursos de MFT" en la página 234).

En versiones anteriores a IBM MQ 9.1.3, si un archivo no se puede transferir, la transferencia de archivos solo se puede iniciar de nuevo si el archivo se suprime y luego se vuelve a colocar en el directorio, o si el archivo se actualiza de modo que se modifique el atributo de última fecha de modificación, o si se vuelve a crear el propio supervisor de recursos.

Sin embargo, puede borrar el historial del supervisor de recursos utilizando el mandato **fteClearMonitorHistory** o utilizando IBM MQ Explorer. El borrado del historial permite que otra solicitud de transferencia de un archivo que no se ha podido transferir se envíe sin la necesidad de suprimir el archivo y luego volverlo a colocar en el directorio, o actualizar el archivo para cambiar su atributo de última fecha de modificación, que es útil, por ejemplo, en situaciones en las que es necesario transferir el archivo, pero no es posible modificarlo. Poder borrar el historial de un supervisor de recursos significa también que no se tiene que volver a crear el supervisor de recursos para enviar otra solicitud de transferencia de un archivo que no se ha podido transferir.

z/OS El miembro SCSQFCMD de ejemplo que se suministra con Managed File Transfer en z/OS incluye un script JCL para borrar el historial de un supervisor.

Procedimiento

- Para utilizar el mandato **fteClearMonitorHistory** para borrar el historial del supervisor de recursos, especifique el mandato en el formato siguiente:

  ```
  fteClearMonitorHistory -p <configuration> -ma <agent name> -mn <monitor name> -w 1000
  ```

  Solo se necesitan los parámetros **-ma** y **-mn**. Los demás parámetros son opcionales. Para obtener más información sobre cómo utilizar el mandato **fteClearMonitorHistory**, incluidos los ejemplos, consulte fteClearMonitorHistory: borrar el historial de supervisor de recursos.

  Si el historial se borra satisfactoriamente, el mandato genera el siguiente mensaje:

  ```
  BFGCL0780I: Se ha emitido una solicitud para borrar el historial del supervisor de recursos
  'nombre de supervisor' del agente 'nombre de agente'.
  BFGCL0251I: La solicitud se ha completado satisfactoriamente.
  ```

  y registra el éxito en el archivo `output0.log` del agente.

  Si el intento de borrar el historial del supervisor de recursos falla, **fteClearMonitorHistory** genera un mensaje de error y registra la anomalía en el archivo `output0.log` del agente.

- Para utilizar la vista de supervisor de recursos en el plug-in de IBM MQ Explorer MFT para borrar el historial del supervisor de recursos, pulse el botón derecho (del ratón) en el supervisor de recursos y seleccione **Borrar historial** en el menú desplegable.

  Si el historial se borra satisfactoriamente, se visualiza el siguiente mensaje:

  ```
  BFGUI00171: El historial del supervisor de recursos se ha borrado correctamente.
  ```

  Si falla el intento de borrar el historial, se visualiza un mensaje de error. Por ejemplo:

  ```
  BFGUI0016E No se ha podido borrar el historial del supervisor de recursos especificado -
  motivo 2059.
  ```

Cómo trabajar con plantillas de transferencia de archivos

Las plantillas de transferencia de archivos se pueden utilizar para almacenar valores de transferencia de archivos comunes para transferencias repetitivas y complejas. Cree una plantilla de transferencia desde la línea de mandatos utilizando el mandato **fteCreateTemplate** o utilice IBM MQ Explorer para crear una plantilla de transferencia mediante el asistente **Crear nueva plantilla para transferencia de archivos** o guarde una plantilla mientras crea una transferencia de archivos marcando el recuadro de selección **Guardar valores de transferencia como plantilla**. La ventana **Plantillas de transferencia** muestra todas las plantillas de transferencia que ha creado en la red de Managed File Transfer.

Acerca de esta tarea

Para crear una plantilla de transferencia desde la línea de mandatos, utilice el mandato fteCreateTemplate. A continuación, cuando desee someter una plantilla de transferencia que haya creado en la línea de mandatos, pulse **Someter** en IBM MQ Explorer.

Para ver las plantillas de transferencia en IBM MQ Explorer, siga estos pasos:

Procedimiento

1. Expanda **Transferencia de archivos gestionada** en la vista Navegador. En la vista Contenido, aparece **Central de transferencias de archivos gestionadas**.
2. En la vista Navegador, aparecen listados todos los gestores de colas de coordinación. Expanda el nombre del gestor de colas de coordinación que ha utilizado para la transferencia planificada. Si desea cambiar el gestor de colas de coordinación al que está conectado, pulse el botón derecho del ratón en el nombre del gestor de colas de coordinación que desea utilizar en la vista Navegador y pulse **Conectar**.
3. Pulse **Plantillas de transferencia**. En la vista Contenido, aparece la ventana **Plantillas de transferencia**.

4. La ventana **Plantillas de transferencia** muestra los siguientes detalles sobre las transferencias de archivos:
   a) **Nombre** El nombre de la plantilla de transferencia de archivos.
   b) **Origen** El nombre del agente utilizado para transferir el archivo del sistema de origen.
   c) **Archivo de origen** El nombre del archivo que se transferirá al sistema host.
      Expanda la información de plantilla de transferencia para ver este campo.
   d) **Destino** El nombre del agente utilizado para recibir el archivo en el sistema de destino.
   e) **Archivo de destino** El nombre del archivo de destino después de que se haya transferido al sistema de destino.
      Expanda la información de plantilla de transferencia para ver esta carpeta.
   f) **Inicio planificado (huso horario seleccionado)** La fecha y hora en que se ha planificado que se inicie la transferencia de archivos, en el huso horario utilizado por el administrador. Para cambiar el huso horario visualizado, pulse **Ventana** > **Preferencias** > **IBM MQ Explorer** > **Managed File Transfer** y seleccione un huso horario alternativo en la lista **Huso horario:**. Pulse **Aceptar**.
   g) **Sucesos desencadenantes** El tipo de suceso que desencadena el inicio de la transferencia de archivos. El tipo puede ser uno de los valores siguientes: `existe`, `no existe` o `sobrepasa`.

Resultados

Para renovar lo que se visualiza en la ventana **Plantillas de transferencia**, pulse el botón Renovar en la barra de herramientas de la vista Contenido.

Para someter una plantilla de transferencia e iniciar la transferencia definida en la plantilla, pulse el botón derecho del ratón en el nombre de plantilla y pulse **Someter**.

Para modificar una plantilla de transferencia, pulse el botón derecho del ratón en el nombre de plantilla y pulse **Editar**. Todos los archivos incluidos en la plantilla original aparecen listados como parte de un grupo de transferencias, incluso si no se incluyeron como parte de un grupo en la plantilla original. Si desea eliminar un archivo de la plantilla, debe seleccionar la especificación de archivo en el grupo y pulsar **Eliminar seleccionado**. Si desea añadir nuevas especificaciones de archivo a la plantilla, utilice los campos del panel de plantilla y pulse el botón **Añadir a grupo**. Cuando haya realizado las modificaciones, se le solicitará que asigne un nuevo nombre a la plantilla editada.

Para crear una transferencia de archivos a partir de una plantilla de transferencia, pulse el botón derecho del ratón en el nombre de plantilla y pulse **Editar como nueva transferencia**.

Para crear una copia duplicada de una plantilla de transferencia, pulse el botón derecho del ratón en el nombre de plantilla y pulse **Duplicar**. La plantilla de transferencia duplicada se guarda automáticamente con el mismo nombre que la plantilla original, con la palabra "(copia)" añadida al final.

Para suprimir una plantilla de transferencia, pulse el botón derecho del ratón en el nombre de plantilla y pulse **Suprimir**.

Tareas relacionadas

“Creación de una plantilla de transferencia de archivos mediante IBM MQ Explorer” en la página 268
Puede crear una plantilla de transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos. A continuación, puede utilizar la plantilla para crear nuevas transferencias de archivos utilizando los detalles de plantilla o someter la plantilla para empezar la transferencia de archivos.

Referencia relacionada

`fteCreateTemplate`: crear nueva plantilla de transferencia de archivos
fteListTemplates
fteDeleteTemplates

Creación de una plantilla de transferencia de archivos mediante IBM MQ Explorer

Puede crear una plantilla de transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos. A continuación, puede utilizar la plantilla para crear nuevas transferencias de archivos utilizando los detalles de plantilla o someter la plantilla para empezar la transferencia de archivos.

Acerca de esta tarea

Para crear una transferencia de archivos desde la línea de mandatos, utilice el mandato fteCreateTemplate.

Para crear una plantilla de transferencia de archivos mediante el asistente **Crear nueva plantilla para transferencia de archivos gestionada** en IBM MQ Explorer, ejecute los pasos siguientes:

Procedimiento

1. En la vista Navegador, pulse **Transferencia de archivos gestionada**. En la vista Contenido, aparece **Central de transferencias de archivos gestionadas**.
2. En la vista Navegador, aparecen todos los gestores de colas de coordinación. Expanda el nombre del gestor de colas de coordinación que ha utilizado para la transferencia planificada. Si desea cambiar el gestor de colas de coordinación al que está conectado, pulse el botón derecho del ratón en el nombre del gestor de colas de coordinación que desea utilizar en la vista Navegador y pulse **Conectar**.
3. Inicie el asistente **Crear nueva plantilla para transferencia de archivos gestionada** pulsando con el botón derecho del ratón en **Plantillas de transferencia** y a continuación, pulse **Nueva plantilla** .
4. Siga las instrucciones de los paneles del asistente. Para cada panel se proporciona ayuda según contexto. Para acceder a la ayuda según contexto en Windows, pulse F1. En Linux, pulse `Ctrl`+F1 o `Shift`+F1.

   Si ha creado una plantilla que contiene todos los detalles de la transferencia necesarios, asegúrese de que ha seleccionado el recuadro de selección **Guardar valores de transferencia como plantilla** en la página **Resumen de la transferencia** si este recuadro de selección aún no está seleccionado. Asimismo, especifique un nombre para la plantilla en el campo Nombre. Si crea una plantilla que aún no contiene todos los detalles de la transferencia necesarios, el recuadro de selección **Guardar valores de transferencia como plantilla** se marcará automáticamente.

Tareas relacionadas

“Cómo trabajar con plantillas de transferencia de archivos” en la página 266
Las plantillas de transferencia de archivos se pueden utilizar para almacenar valores de transferencia de archivos comunes para transferencias repetitivas y complejas. Cree una plantilla de transferencia desde la línea de mandatos utilizando el mandato **fteCreateTemplate** o utilice IBM MQ Explorer para crear una plantilla de transferencia mediante el asistente **Crear nueva plantilla para transferencia de archivos** o guarde una plantilla mientras crea una transferencia de archivos marcando el recuadro de selección **Guardar valores de transferencia como plantilla**. La ventana **Plantillas de transferencia** muestra todas las plantillas de transferencia que ha creado en la red de Managed File Transfer.

Referencia relacionada

fteCreateTemplate: crear nueva plantilla de transferencia de archivos
fteListTemplates
fteDeleteTemplates

Copia de seguridad de una definición de plantilla de transferencia de archivos

Las plantillas de transferencia de archivos contienen un documento XML que define las especificaciones de archivo de origen y destino para la transferencia. Puede utilizar este archivo XML como entrada para el mandato **fteCreateTemplate** para volver a crear una plantilla de transferencia de archivos.

Acerca de esta tarea

Para realizar copia de seguridad del documento XML que contiene las especificaciones de archivo de origen y destino de una plantilla de transferencia, utilice el mandato fteCreateTransfer o IBM MQ Explorer. Para crear un archivo de copia de seguridad con formato XML de plantilla de transferencia, siga estos pasos:

Procedimiento

- Método uno: utilice el parámetro **-gt** en un mandato fteCreateTransfer para generar un mensaje XML de plantilla de transferencia en un archivo nuevo.
- Método dos: cree la plantilla utilizando IBM MQ Explorer.

  Cuando llegue a la página *Resumen de la plantilla de la transferencia*:

  a) Copie *Vista previa XML de mensaje de solicitud*.

  b) Guarde este mensaje XML de plantilla de transferencia en un archivo nuevo.
- Método tres: utilice IBM MQ Explorer para hacer copia de seguridad de las plantillas existentes.

  a) Vaya a **Managed File Transfer** > **Nombre del gestor de colas** > **Plantillas de transferencia**.

  b) En el panel de transferencia, resalte la plantilla de la que se debe hacer una copia de seguridad, pulse el botón derecho (del ratón) y seleccione **Editar** en el menú emergente.

  c) Pulse **Siguiente** hasta que llegue a la página *Resumen de la plantilla de la transferencia*.

  d) Copie *Vista previa XML de mensaje de solicitud*.

  e) Guarde este mensaje XML de plantilla de transferencia en un archivo nuevo.

Resultados

Puede utilizar el archivo de mensaje XML de plantilla de transferencia, creado mediante uno de los métodos anteriores, como entrada para el mandato fteCreateTemplate. Consulte el mandato **fteCreateTemplate** para obtener detalles de cómo utilizar este mandato.

Referencia relacionada

Mandato fteCreateTemplate

Mandato fteListtTmplates

Transferencia de datos de archivos a mensajes

Puede utilizar la característica de archivo a mensaje de Managed File Transfer para transferir datos desde un archivo a un único mensaje o varios mensajes, en una cola de IBM MQ.

Para obtener información sobre transferencias de mensaje a archivo, consulte “Transferir datos de mensajes a archivos” en la página 285.

El agente de destino para la transferencia de archivo a mensaje no puede ser un agente de puente de protocolo o un agente de puente Connect:Direct.

Puede transferir datos de archivo a datos de mensaje de IBM MQ. Los mensajes de IBM MQ pueden ser leídos y utilizados por las aplicaciones. Los siguientes tipos de transferencias de archivo a mensaje está soportados:

- De un único archivo a un único mensaje. El mensaje no tiene establecido un ID de grupo de IBM MQ.
- De un único archivo a varios mensajes, dividiendo el archivo en mensajes de una longitud determinada. Los mensajes tienen todos el mismo ID de grupo de IBM MQ.
- De un único archivo a varios mensajes, dividiendo un archivo de texto en un delimitador de expresión regular Java. Los mensajes tienen todos el mismo ID de grupo de IBM MQ.
- De un único archivo hasta varios mensajes, dividiendo un archivo binario en un delimitador hexadecimal. Los mensajes tienen todos el mismo ID de grupo de IBM MQ.

Si desea dividir un archivo binario utilizando una secuencia de bytes como delimitador, utilice el parámetro **-sqdb** del mandato **fteCreateTransfer**. Para obtener más información, consulte el parámetro -sqdb.

De forma predeterminada los mensajes creados por una transferencia de archivo a mensaje son persistentes. Los mensajes pueden configurarse para ser no persistente o para tener el valor de persistencia definido por la cola de destino.

Si especifica que un archivo se divida en varios mensajes, todos los mensajes creados desde el archivo tendrán el mismo ID de grupo de IBM MQ. Si no especifica que un archivo se divida en varios mensajes, sólo se creará un mensaje desde el archivo y este mensaje no tendrá establecido el ID de grupo de IBM MQ.

Si va a transferir archivos a mensajes grandes, o a muchos mensajes pequeños, es posible que tenga que cambiar algunas propiedades de IBM MQ o Managed File Transfer. Si desea más información al respecto, consulte Instrucciones para establecer atributos MQ y propiedades MFT asociadas al tamaño del mensaje.

Nota: Si la cola de destino es una cola en clúster o un alias para una cola en clúster, obtendrá un mensaje de error al transferir un archivo en una cola si la propiedad de agente enableClusterQueueInputOutput no se ha establecido en true. Si desea más información, consulte Qué hacer si la cola de destino es una cola agrupada en clúster, o un alias a una cola agrupada en clúster

Tareas relacionadas

“Configuración de un agente para realizar transferencias de archivo a mensaje” en la página 271
De forma predeterminada, los agentes pueden realizar transferencias de archivo a mensaje o de mensaje a archivo. Para habilitar esta función debe establecer la propiedad de agente enableQueueInputOutput en true (verdadera). Para habilitar la escritura en las colas en clúster de IBM MQ, también debe establecer la propiedad de agente enableClusterQueueInputOutput en true.

“Ejemplo: Transferencia de un único archivo en un mensaje” en la página 272
Puede especificar una cola como destino de una transferencia de archivo utilizando el parámetro **-dq** con el mandato **fteCreateTransfer**. El archivo de origen debe ser más pequeño que la longitud máxima de mensaje establecida en la cola de destino. La cola de destino no tiene que estar en el mismo gestor de colas que el gestor de colas al que se conecta el agente de destino, pero estos dos gestores de colas deben poder comunicarse.

“Ejemplo: División de un único archivo en varios mensajes por longitud” en la página 274
Puede dividir un archivo en varios mensajes de IBM MQ utilizando el parámetro **-qs** del mandato **fteCreateTransfer**. El archivo se divide en secciones de longitud fija, cada una de las cuales se escribe en un mensaje individual.

“Ejemplo: División de un archivo de texto con un delimitador de expresión regular e incluyendo el delimitador en los mensajes” en la página 277
Transferencia de un único archivo de texto en varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada e inclusión de la coincidencia de la expresión regular en los mensajes resultantes. Para hacerlo, utilice los parámetros **-dqdt** y **-qi** del mandato **fteCreateTransfer**.

“Ejemplo: división un archivo de texto en varios mensajes utilizando un delimitador de expresión regular” en la página 275
Transferencia de un único archivo de texto con varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada. Para hacerlo, utilizará el parámetro **-dqdt** del mandato **fteCreateTransfer**.

“Ejemplo: establecer propiedades de mensaje de IBM MQ en una transferencia de archivo a mensaje” en la página 279
Puede utilizar el parámetro **-qmp** en el mandato **fteCreateTransfer** para especificar si las propiedades de mensaje de IBM MQ han sido establecidas por la transferencia en el primer mensaje escrito en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

“Ejemplo: Configuración de las propiedades definidas por el usuario en una transferencia de archivo a mensaje” en la página 281
Los metadatos definidos por el usuario se establecen como una propiedad de mensaje de IBM MQ en el primer mensaje que la transferencia ha grabado en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a las cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

“Inicio de una nueva transferencia de archivos” en la página 221
Puede iniciar una nueva de transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos y puede elegir transferir un único archivo o varios archivos de un grupo.

Referencia relacionada

“Fallo en una transferencia de archivo a mensaje” en la página 284
Si una transferencia de archivo a mensaje falla después de que el agente haya empezado a escribir los datos de archivo en la cola de destino, el agente escribe un mensaje en la cola para indicar a la aplicación que consume los mensajes que se ha producido un fallo.

Propiedades de mensaje de MQ establecidas por MFT en mensajes escritos en las colas de destino

Guía para establecer atributos MQ y propiedades de MFT asociadas al tamaño de mensaje

Configuración de un agente para realizar transferencias de archivo a mensaje

De forma predeterminada, los agentes pueden realizar transferencias de archivo a mensaje o de mensaje a archivo. Para habilitar esta función debe establecer la propiedad de agente enableQueueInputOutput en true (verdadera). Para habilitar la escritura en las colas en clúster de IBM MQ, también debe establecer la propiedad de agente enableClusterQueueInputOutput en true.

Acerca de esta tarea

Si intenta realizar una transferencia de archivo a mensaje a un agente de destino que no tenga la propiedad `enableQueueInputOutput` establecida en true, la transferencia fallará. El mensaje de registro de transferencia publicado en el gestor de cola de coordinación contiene el siguiente mensaje:

```
BFGIO0197E: An attempt to write to a queue was rejected by the destination agent. The
agent must have enableQueueInputOutput=true set in the agent.properties file to
support transferring to a queue.
```

Para que el agente pueda escribir y leer en las colas, siga los siguientes pasos:

Procedimiento

1. Detenga el agente de destino utilizando el mandato **fteStopAgent**.
2. Edite el archivo `agent.properties` para incluir la línea `enableQueueInputOutput=true`.
   El archivo `agent.properties` se encuentra en el directorio *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*destination_agent_name*.
3. Opcional: Edite el archivo `agent.properties` para incluir la línea `enableClusterQueueInputOutput=true`. El archivo `agent.properties` se encuentra en el directorio *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*destination_agent_name*.
4. Inicie el agente de destino utilizando el mandato **fteStartAgent**.

Conceptos relacionados

“Transferencia de datos de archivos a mensajes” en la página 269
Puede utilizar la característica de archivo a mensaje de Managed File Transfer para transferir datos desde un archivo a un único mensaje o varios mensajes, en una cola de IBM MQ.

Tareas relacionadas

“Ejemplo: Transferencia de un único archivo en un mensaje” en la página 272

Puede especificar una cola como destino de una transferencia de archivo utilizando el parámetro **-dq** con el mandato **fteCreateTransfer**. El archivo de origen debe ser más pequeño que la longitud máxima de mensaje establecida en la cola de destino. La cola de destino no tiene que estar en el mismo gestor de colas que el gestor de colas al que se conecta el agente de destino, pero estos dos gestores de colas deben poder comunicarse.

"Ejemplo: División de un único archivo en varios mensajes por longitud" en la página 274
Puede dividir un archivo en varios mensajes de IBM MQ utilizando el parámetro **-qs** del mandato **fteCreateTransfer**. El archivo se divide en secciones de longitud fija, cada una de las cuales se escribe en un mensaje individual.

"Ejemplo: División de un archivo de texto con un delimitador de expresión regular e incluyendo el delimitador en los mensajes" en la página 277
Transferencia de un único archivo de texto en varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada e inclusión de la coincidencia de la expresión regular en los mensajes resultantes. Para hacerlo, utilice los parámetros **-dqdt** y **-qi** del mandato **fteCreateTransfer**.

"Ejemplo: división un archivo de texto en varios mensajes utilizando un delimitador de expresión regular" en la página 275
Transferencia de un único archivo de texto con varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada. Para hacerlo, utilizará el parámetro **-dqdt** del mandato **fteCreateTransfer**.

"Ejemplo: establecer propiedades de mensaje de IBM MQ en una transferencia de archivo a mensaje" en la página 279
Puede utilizar el parámetro **-qmp** en el mandato **fteCreateTransfer** para especificar si las propiedades de mensaje de IBM MQ han sido establecidas por la transferencia en el primer mensaje escrito en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

"Ejemplo: Configuración de las propiedades definidas por el usuario en una transferencia de archivo a mensaje" en la página 281
Los metadatos definidos por el usuario se establecen como una propiedad de mensaje de IBM MQ en el primer mensaje que la transferencia ha grabado en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a las cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

Referencia relacionada

fteStopAgent

fteStartAgent

El archivo MFT `agent.properties`

"Fallo en una transferencia de archivo a mensaje" en la página 284
Si una transferencia de archivo a mensaje falla después de que el agente haya empezado a escribir los datos de archivo en la cola de destino, el agente escribe un mensaje en la cola para indicar a la aplicación que consume los mensajes que se ha producido un fallo.

Ejemplo: Transferencia de un único archivo en un mensaje

Puede especificar una cola como destino de una transferencia de archivo utilizando el parámetro **-dq** con el mandato **fteCreateTransfer**. El archivo de origen debe ser más pequeño que la longitud máxima de mensaje establecida en la cola de destino. La cola de destino no tiene que estar en el mismo gestor de colas que el gestor de colas al que se conecta el agente de destino, pero estos dos gestores de colas deben poder comunicarse.

Acerca de esta tarea

El archivo de origen se llama `/tmp/single_record.txt` y se encuentra en el mismo sistema que el agente de origen, AGENT_NEPPTUNE. El agente de origen, AGENT_NEPTUNE, utiliza el gestor de colas QM_NEPTUNE. El agente de destino es AGENT_VENUS y este agente se conecta con el gestor de colas

QM_VENUS. La cola de destino RECEIVING_QUEUE, se encuentra en el gestor de colas QM_MERCURY. QM_MERCURY se encuentra en la misma red de IBM MQ que el gestor de colas QM_VENUS al que puede acceder.

Procedimiento

Escriba el siguiente mandato:

```
fteCreateTransfer -sa AGENT_NEPTUNE -sm QM_NEPTUNE -da AGENT_VENUS -dm QM_VENUS
                  -dq RECEIVING_QUEUE@QM_MERCURY /tmp/single_record.txt
```

Si la cola de destino se encuentra en un gestor de colas diferente al gestor de colas utilizado por el agente de destino, debe especificar el valor del parámetro **-dq** en el formato siguiente *nombre_cola*@*nombre_gestor_colas*. Si no especifica @*nombre_gestor_colas* en el valor, el agente de destino da por supuesto que la cola de destino se encuentra en el gestor de colas del agente de destino. Se produce la excepción cuando la propiedad de agente enableClusterQueueInputOutput se ha establecido en true. En este caso, el agente de destino utilizará procedimientos de resolución de IBM MQ estándar para determinar dónde se encuentra la cola.

El agente de origen, AGENT_NEPPTUNE, lee los datos del archivo `/tmp/single_record.txt` y transfiere estos datos al agente de destino, AGENT_VENUS. El agente de destino, AGENT_VENUS, envía los datos a un mensaje persistente en la cola RECEIVING_QUEUE@QM_MERCURY. El mensaje no tiene establecido un ID de grupo de IBM MQ.

Conceptos relacionados

“Transferencia de datos de archivos a mensajes” en la página 269
Puede utilizar la característica de archivo a mensaje de Managed File Transfer para transferir datos desde un archivo a un único mensaje o varios mensajes, en una cola de IBM MQ.

Tareas relacionadas

“Configuración de un agente para realizar transferencias de archivo a mensaje” en la página 271
De forma predeterminada, los agentes pueden realizar transferencias de archivo a mensaje o de mensaje a archivo. Para habilitar esta función debe establecer la propiedad de agente enableQueueInputOutput en true (verdadera). Para habilitar la escritura en las colas en clúster de IBM MQ, también debe establecer la propiedad de agente enableClusterQueueInputOutput en true.

“Ejemplo: División de un único archivo en varios mensajes por longitud” en la página 274
Puede dividir un archivo en varios mensajes de IBM MQ utilizando el parámetro **-qs** del mandato **fteCreateTransfer**. El archivo se divide en secciones de longitud fija, cada una de las cuales se escribe en un mensaje individual.

“Ejemplo: División de un archivo de texto con un delimitador de expresión regular e incluyendo el delimitador en los mensajes” en la página 277
Transferencia de un único archivo de texto en varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada e inclusión de la coincidencia de la expresión regular en los mensajes resultantes. Para hacerlo, utilice los parámetros **-dqdt** y **-qi** del mandato **fteCreateTransfer**.

“Ejemplo: división un archivo de texto en varios mensajes utilizando un delimitador de expresión regular” en la página 275
Transferencia de un único archivo de texto con varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada. Para hacerlo, utilizará el parámetro **-dqdt** del mandato **fteCreateTransfer**.

“Ejemplo: establecer propiedades de mensaje de IBM MQ en una transferencia de archivo a mensaje” en la página 279
Puede utilizar el parámetro **-qmp** en el mandato **fteCreateTransfer** para especificar si las propiedades de mensaje de IBM MQ han sido establecidas por la transferencia en el primer mensaje escrito en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

"Ejemplo: Configuración de las propiedades definidas por el usuario en una transferencia de archivo a mensaje" en la página 281
Los metadatos definidos por el usuario se establecen como una propiedad de mensaje de IBM MQ en el primer mensaje que la transferencia ha grabado en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a las cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

"Inicio de una nueva transferencia de archivos" en la página 221
Puede iniciar una nueva de transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos y puede elegir transferir un único archivo o varios archivos de un grupo.

Referencia relacionada

"Fallo en una transferencia de archivo a mensaje" en la página 284
Si una transferencia de archivo a mensaje falla después de que el agente haya empezado a escribir los datos de archivo en la cola de destino, el agente escribe un mensaje en la cola para indicar a la aplicación que consume los mensajes que se ha producido un fallo.

Ejemplo: División de un único archivo en varios mensajes por longitud

Puede dividir un archivo en varios mensajes de IBM MQ utilizando el parámetro **-qs** del mandato **fteCreateTransfer**. El archivo se divide en secciones de longitud fija, cada una de las cuales se escribe en un mensaje individual.

Acerca de esta tarea

El archivo de origen se llama `/tmp/source.file` y tiene un tamaño de 36 KB. El archivo de origen se encuentra en el mismo sistema que el archivo de origen AGENT_NEPTUNE. El agente de origen, AGENT_NEPTUNE, se conecta con el gestor de colas QM_NEPTUNE. El agente de destino es AGENT_MERCURY, que se conecta con el gestor de colas QM_MERCURY. La cola de destino RECEIVING_QUEUE, también se encuentra en el gestor de colas QM_MERCURY. La transferencia divide el archivo de origen en secciones de 1 KB y escribe cada una de estas secciones en un mensaje en RECEIVING_QUEUE.

Procedimiento

Escriba el siguiente mandato:

```
fteCreateTransfer -sa AGENT_NEPTUNE -sm QM_NEPTUNE -da AGENT_MERCURY -dm QM_MERCURY
                  -dq RECEIVING_QUEUE -qs 1K /tmp/source.file
```

El agente de origen, AGENT_NEPPTUNE, lee los datos del archivo `/tmp/source.file` y transfiere estos datos al agente de destino, AGENT_MERCURY. El agente de destino, AGENT_MERCURY, escribe los datos en tres y seis mensajes persistentes de 1KB en la cola RECEIVING_QUEUE@QM_MERCURY. Estos mensajes tienen todos el mismo ID de grupo de IBM MQ; el último mensaje del grupo tiene el distintivo de IBM MQ LAST_MSG_IN_GROUP definido.

Conceptos relacionados

"Transferencia de datos de archivos a mensajes" en la página 269
Puede utilizar la característica de archivo a mensaje de Managed File Transfer para transferir datos desde un archivo a un único mensaje o varios mensajes, en una cola de IBM MQ.

Tareas relacionadas

"Configuración de un agente para realizar transferencias de archivo a mensaje" en la página 271
De forma predeterminada, los agentes pueden realizar transferencias de archivo a mensaje o de mensaje a archivo. Para habilitar esta función debe establecer la propiedad de agente enableQueueInputOutput en true (verdadera). Para habilitar la escritura en las colas en clúster de IBM MQ, también debe establecer la propiedad de agente enableClusterQueueInputOutput en true.

"Ejemplo: Transferencia de un único archivo en un mensaje" en la página 272
Puede especificar una cola como destino de una transferencia de archivo utilizando el parámetro **-dq** con el mandato **fteCreateTransfer**. El archivo de origen debe ser más pequeño que la longitud máxima

de mensaje establecida en la cola de destino. La cola de destino no tiene que estar en el mismo gestor de colas que el gestor de colas al que se conecta el agente de destino, pero estos dos gestores de colas deben poder comunicarse.

“Ejemplo: División de un archivo de texto con un delimitador de expresión regular e incluyendo el delimitador en los mensajes” en la página 277
Transferencia de un único archivo de texto en varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada e inclusión de la coincidencia de la expresión regular en los mensajes resultantes. Para hacerlo, utilice los parámetros **-dqdt** y **-qi** del mandato **fteCreateTransfer**.

“Ejemplo: división un archivo de texto en varios mensajes utilizando un delimitador de expresión regular” en la página 275
Transferencia de un único archivo de texto con varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada. Para hacerlo, utilizará el parámetro **-dqdt** del mandato **fteCreateTransfer**.

“Ejemplo: establecer propiedades de mensaje de IBM MQ en una transferencia de archivo a mensaje” en la página 279
Puede utilizar el parámetro **-qmp** en el mandato **fteCreateTransfer** para especificar si las propiedades de mensaje de IBM MQ han sido establecidas por la transferencia en el primer mensaje escrito en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

“Ejemplo: Configuración de las propiedades definidas por el usuario en una transferencia de archivo a mensaje” en la página 281
Los metadatos definidos por el usuario se establecen como una propiedad de mensaje de IBM MQ en el primer mensaje que la transferencia ha grabado en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a las cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

“Inicio de una nueva transferencia de archivos” en la página 221
Puede iniciar una nueva de transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos y puede elegir transferir un único archivo o varios archivos de un grupo.

Referencia relacionada

“Fallo en una transferencia de archivo a mensaje” en la página 284
Si una transferencia de archivo a mensaje falla después de que el agente haya empezado a escribir los datos de archivo en la cola de destino, el agente escribe un mensaje en la cola para indicar a la aplicación que consume los mensajes que se ha producido un fallo.

Ejemplo: división un archivo de texto en varios mensajes utilizando un delimitador de expresión regular

Transferencia de un único archivo de texto con varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada. Para hacerlo, utilizará el parámetro **-dqdt** del mandato **fteCreateTransfer**.

Acerca de esta tarea

El archivo se divide en secciones de longitud variable, cada una de las cuales se escribe en un mensaje individual. El archivo de texto se divide en cada punto en que el texto en el archivo coincide con una expresión regular determinada. El archivo de origen se llama `/tmp/names.text` y tiene el contenido siguiente:

```
Jenny Jones,John Smith,Jane Brown
```

La expresión regular que especifica dónde dividir el archivo es un carácter de coma (`,`).

El archivo de origen se encuentra en el mismo sistema del agente de origen *AGENT_NEPTUNE*, que se conecta con el gestor de colas QM_NEPTUNE. La cola de destino *RECEIVING_QUEUE*, se encuentra en el gestor de colas *QM_MERCURY*. *QM_MERCURY* es también el gestor de colas utilizado por el agente de destino *AGENT_MERCURY*. La transferencia divide el archivo de origen en secciones y escribe cada una de estas secciones en un mensaje en *RECEIVING_QUEUE*.

Procedimiento

Escriba el siguiente mandato:

```
fteCreateTransfer -sa AGENT_NEPTUNE -sm QM_NEPTUNE -da AGENT_MERCURY -dm QM_MERCURY
                  -dq RECEIVING_QUEUE -t text -dqdp postfix -dqdt "," /tmp/names.text
```

El agente de origen, AGENT_NEPPTUNE, lee los datos del archivo `/tmp/names.text` y transfiere estos datos al agente de destino, AGENT_MERCURY. El agente de destino, AGENT_MERCURY, escribe los datos en tres mensajes persistentes en la cola *RECEIVING_QUEUE*. Estos mensajes tienen todos el mismo ID de grupo de IBM MQ; el último mensaje del grupo tiene el distintivo de IBM MQ LAST_MSG_IN_GROUP definido.

Los datos en los mensajes son los siguientes.

- Primer mensaje:

  ```
  Jenny Jones
  ```

- Segundo mensaje:

  ```
  John Smith
  ```

- Tercer mensaje:

  ```
  Jane Brown
  ```

Conceptos relacionados

“Transferencia de datos de archivos a mensajes” en la página 269
Puede utilizar la característica de archivo a mensaje de Managed File Transfer para transferir datos desde un archivo a un único mensaje o varios mensajes, en una cola de IBM MQ.

Tareas relacionadas

“Configuración de un agente para realizar transferencias de archivo a mensaje” en la página 271
De forma predeterminada, los agentes pueden realizar transferencias de archivo a mensaje o de mensaje a archivo. Para habilitar esta función debe establecer la propiedad de agente enableQueueInputOutput en true (verdadera). Para habilitar la escritura en las colas en clúster de IBM MQ, también debe establecer la propiedad de agente enableClusterQueueInputOutput en true.

“Ejemplo: Transferencia de un único archivo en un mensaje” en la página 272
Puede especificar una cola como destino de una transferencia de archivo utilizando el parámetro **-dq** con el mandato **fteCreateTransfer**. El archivo de origen debe ser más pequeño que la longitud máxima de mensaje establecida en la cola de destino. La cola de destino no tiene que estar en el mismo gestor de colas que el gestor de colas al que se conecta el agente de destino, pero estos dos gestores de colas deben poder comunicarse.

“Ejemplo: División de un único archivo en varios mensajes por longitud” en la página 274
Puede dividir un archivo en varios mensajes de IBM MQ utilizando el parámetro **-qs** del mandato **fteCreateTransfer**. El archivo se divide en secciones de longitud fija, cada una de las cuales se escribe en un mensaje individual.

“Ejemplo: División de un archivo de texto con un delimitador de expresión regular e incluyendo el delimitador en los mensajes” en la página 277

Transferencia de un único archivo de texto en varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada e inclusión de la coincidencia de la expresión regular en los mensajes resultantes. Para hacerlo, utilice los parámetros **-dqdt** y **-qi** del mandato **fteCreateTransfer**.

"Ejemplo: establecer propiedades de mensaje de IBM MQ en una transferencia de archivo a mensaje" en la página 279
Puede utilizar el parámetro **-qmp** en el mandato **fteCreateTransfer** para especificar si las propiedades de mensaje de IBM MQ han sido establecidas por la transferencia en el primer mensaje escrito en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

"Ejemplo: Configuración de las propiedades definidas por el usuario en una transferencia de archivo a mensaje" en la página 281
Los metadatos definidos por el usuario se establecen como una propiedad de mensaje de IBM MQ en el primer mensaje que la transferencia ha grabado en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a las cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

"Inicio de una nueva transferencia de archivos" en la página 221
Puede iniciar una nueva de transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos y puede elegir transferir un único archivo o varios archivos de un grupo.

Referencia relacionada

"Fallo en una transferencia de archivo a mensaje" en la página 284
Si una transferencia de archivo a mensaje falla después de que el agente haya empezado a escribir los datos de archivo en la cola de destino, el agente escribe un mensaje en la cola para indicar a la aplicación que consume los mensajes que se ha producido un fallo.

Expresiones regulares utilizadas por MFT

Ejemplo: División de un archivo de texto con un delimitador de expresión regular e incluyendo el delimitador en los mensajes

Transferencia de un único archivo de texto en varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada e inclusión de la coincidencia de la expresión regular en los mensajes resultantes. Para hacerlo, utilice los parámetros **-dqdt** y **-qi** del mandato **fteCreateTransfer**.

Acerca de esta tarea

Transfiera un único archivo de texto en varios mensajes en una cola. El archivo se divide en secciones de longitud variable, cada una de las cuales se escribe en un mensaje individual. El archivo de texto se divide en cada punto en que el texto en el archivo coincide con una expresión regular determinada. El archivo de origen se llama `/tmp/customers.text` y tiene el contenido siguiente:

```
Customer name: John Smith
Customer contact details: john@example.net
Customer number: 314

Customer name: Jane Brown
Customer contact details: jane@example.com
Customer number: 42

Customer name: James Jones
Customer contact details: jjones@example.net
Customer number: 26
```

La expresión regular que especifica dónde dividir el archivo es `Customer\snumber:\s\d+`, que coincide con el texto "Customer number: " seguido de cualquier número de dígitos. Las expresiones regulares especificadas en la línea de mandatos deben rodearse de comillas dobles para impedir que el shell de

mandatos evalúe la expresión regular. La expresión regular se evalúa como expresión regular Java. Si desea más información, consulte Expresiones regulares utilizadas por MFT.

De forma predeterminada, el número de caracteres con el que puede coincidir una expresión regular está establecido en cinco. La expresión regular utilizada en este ejemplo coincide con las series que sean mayores de cinco caracteres. Para permitir coincidencias mayores de cinco caracteres, edite el archivo de propiedades del agente para incluir la propiedad **maxDelimiterMatchLength**.

De forma predeterminada, el texto que coincide con la expresión regular no se incluye en los mensajes. Para incluir el texto que coincide con la expresión regular en los mensajes, como en este ejemplo, utilice el parámetro **-qi**. El archivo de origen se encuentra en el mismo sistema del agente de origen AGENT_NEPTUNE, que se conecta con el gestor de colas QM_NEPTUNE. La cola de destino RECEIVING_QUEUE, se encuentra en el gestor de colas QM_MERCURY. QM_MERCURY es también el gestor de colas utilizado por el agente de destino AGENT_MERCURY. La transferencia divide el archivo de origen en secciones y escribe cada una de estas secciones en un mensaje en RECEIVING_QUEUE.

Procedimiento

1. Detenga el agente de destino utilizando el siguiente mandato:

```
fteStopAgent AGENT_MERCURY
```

2. Añada la siguiente línea al archivo de propiedades del agente para AGENT_MERCURY:

```
maxDelimiterMatchLength=25
```

   Nota: Aumentar el valor de **maxDelimiterMatchLength** puede reducir el rendimiento.

3. Inicie el agente de destino utilizando el siguiente mandato:

```
fteStartAgent AGENT_MERCURY
```

4. Escriba el siguiente mandato:

```
fteCreateTransfer -sa AGENT_NEPTUNE -sm QM_NEPTUNE -da AGENT_MERCURY -dm QM_MERCURY
 -dq RECEIVING_QUEUE
     text -dqdt "Customer\snumber:\s\d+" -qi -dqdp postfix /tmp/customers.text
```

   El agente de origen, AGENT_NEPPTUNE, lee los datos del archivo `/tmp/customers.text` y transfiere estos datos al agente de destino, AGENT_MERCURY. El agente de destino, AGENT_MERCURY, escribe los datos en tres mensajes persistentes en la cola RECEIVING_QUEUE. Estos mensajes tienen todos el mismo ID de grupo de IBM MQ; el último mensaje del grupo tiene el distintivo de IBM MQ LAST_MSG_IN_GROUP definido.

   Los datos en los mensajes son los siguientes.

   - Primer mensaje:

```
Customer name: John Smith
Customer contact details: john@example.net
Customer number: 314
```

   - Segundo mensaje:

```
Customer name: Jane Brown
Customer contact details: jane@example.com
Customer number: 42
```

   - Tercer mensaje:

```
Customer name: James Jones
```

```
Customer contact details: jjones@example.net
Customer number: 26
```

Conceptos relacionados

“Transferencia de datos de archivos a mensajes” en la página 269
Puede utilizar la característica de archivo a mensaje de Managed File Transfer para transferir datos desde un archivo a un único mensaje o varios mensajes, en una cola de IBM MQ.

Tareas relacionadas

“Configuración de un agente para realizar transferencias de archivo a mensaje” en la página 271
De forma predeterminada, los agentes pueden realizar transferencias de archivo a mensaje o de mensaje a archivo. Para habilitar esta función debe establecer la propiedad de agente enableQueueInputOutput en true (verdadera). Para habilitar la escritura en las colas en clúster de IBM MQ, también debe establecer la propiedad de agente enableClusterQueueInputOutput en true.

“Ejemplo: Transferencia de un único archivo en un mensaje” en la página 272
Puede especificar una cola como destino de una transferencia de archivo utilizando el parámetro **-dq** con el mandato **fteCreateTransfer**. El archivo de origen debe ser más pequeño que la longitud máxima de mensaje establecida en la cola de destino. La cola de destino no tiene que estar en el mismo gestor de colas que el gestor de colas al que se conecta el agente de destino, pero estos dos gestores de colas deben poder comunicarse.

“Ejemplo: División de un único archivo en varios mensajes por longitud” en la página 274
Puede dividir un archivo en varios mensajes de IBM MQ utilizando el parámetro **-qs** del mandato **fteCreateTransfer**. El archivo se divide en secciones de longitud fija, cada una de las cuales se escribe en un mensaje individual.

“Ejemplo: división un archivo de texto en varios mensajes utilizando un delimitador de expresión regular” en la página 275
Transferencia de un único archivo de texto con varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada. Para hacerlo, utilizará el parámetro **-dqdt** del mandato **fteCreateTransfer**.

“Ejemplo: establecer propiedades de mensaje de IBM MQ en una transferencia de archivo a mensaje” en la página 279
Puede utilizar el parámetro **-qmp** en el mandato **fteCreateTransfer** para especificar si las propiedades de mensaje de IBM MQ han sido establecidas por la transferencia en el primer mensaje escrito en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

“Ejemplo: Configuración de las propiedades definidas por el usuario en una transferencia de archivo a mensaje” en la página 281
Los metadatos definidos por el usuario se establecen como una propiedad de mensaje de IBM MQ en el primer mensaje que la transferencia ha grabado en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a las cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

“Inicio de una nueva transferencia de archivos” en la página 221
Puede iniciar una nueva de transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos y puede elegir transferir un único archivo o varios archivos de un grupo.

Referencia relacionada

El archivo MFT `agent.properties`

Expresiones regulares utilizadas por MFT

Ejemplo: establecer propiedades de mensaje de IBM MQ en una transferencia de archivo a mensaje

Puede utilizar el parámetro **-qmp** en el mandato **fteCreateTransfer** para especificar si las propiedades de mensaje de IBM MQ han sido establecidas por la transferencia en el primer mensaje escrito en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación

seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

Acerca de esta tarea

Incluya el parámetro `-qmp true` en el mandato **fteCreateTransfer**. En este ejemplo, el ID de usuario MQMD del usuario que envía el mandato es `larmer`.

Procedimiento

Escriba el siguiente mandato:

```
fteCreateTransfer -sa AGENT_JUPITER -da AGENT_SATURN -dq MY_QUEUE@MyQM -qmp true
      -t text /tmp/source_file.txt
```

Las propiedades de mensaje de IBM MQ del primer mensaje escruto por el agente de destino, `AGENT_SATURN`, en la cola, `MY_QUEUE`, en el gestor de colas, `MyQM`, están definidas con estos valores:

```
usr.WMQFTETransferId=414cbaedefa234889d999a8ed09782395ea213ebbc9377cd
usr.WMQFTETransferMode=text
usr.WMQFTESourceAgent=AGENT_JUPITER
usr.WMQFTEDestinationAgent=AGENT_SATURN
usr.WMQFTEFileName=source_file.txt
usr.WMQFTEFileSize=1024
usr.WMQFTEFileLastModified=1273740879040
usr.WMQFTEFileIndex=0
usr.WMQFTEMqmdUser=larmer
```

Conceptos relacionados

"Transferencia de datos de archivos a mensajes" en la página 269
Puede utilizar la característica de archivo a mensaje de Managed File Transfer para transferir datos desde un archivo a un único mensaje o varios mensajes, en una cola de IBM MQ.

Tareas relacionadas

"Configuración de un agente para realizar transferencias de archivo a mensaje" en la página 271
De forma predeterminada, los agentes pueden realizar transferencias de archivo a mensaje o de mensaje a archivo. Para habilitar esta función debe establecer la propiedad de agente enableQueueInputOutput en true (verdadera). Para habilitar la escritura en las colas en clúster de IBM MQ, también debe establecer la propiedad de agente enableClusterQueueInputOutput en true.

"Ejemplo: Transferencia de un único archivo en un mensaje" en la página 272
Puede especificar una cola como destino de una transferencia de archivo utilizando el parámetro **-dq** con el mandato **fteCreateTransfer**. El archivo de origen debe ser más pequeño que la longitud máxima de mensaje establecida en la cola de destino. La cola de destino no tiene que estar en el mismo gestor de colas que el gestor de colas al que se conecta el agente de destino, pero estos dos gestores de colas deben poder comunicarse.

"Ejemplo: División de un único archivo en varios mensajes por longitud" en la página 274
Puede dividir un archivo en varios mensajes de IBM MQ utilizando el parámetro **-qs** del mandato **fteCreateTransfer**. El archivo se divide en secciones de longitud fija, cada una de las cuales se escribe en un mensaje individual.

"Ejemplo: División de un archivo de texto con un delimitador de expresión regular e incluyendo el delimitador en los mensajes" en la página 277
Transferencia de un único archivo de texto en varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada e inclusión de la coincidencia de la expresión regular en los mensajes resultantes. Para hacerlo, utilice los parámetros **-dqdt** y **-qi** del mandato **fteCreateTransfer**.

"Ejemplo: división un archivo de texto en varios mensajes utilizando un delimitador de expresión regular" en la página 275

Transferencia de un único archivo de texto con varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada. Para hacerlo, utilizará el parámetro **-dqdt** del mandato **fteCreateTransfer**.

"Ejemplo: Configuración de las propiedades definidas por el usuario en una transferencia de archivo a mensaje" en la página 281
Los metadatos definidos por el usuario se establecen como una propiedad de mensaje de IBM MQ en el primer mensaje que la transferencia ha grabado en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a las cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

"Inicio de una nueva transferencia de archivos" en la página 221
Puede iniciar una nueva de transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos y puede elegir transferir un único archivo o varios archivos de un grupo.

Referencia relacionada

"Fallo en una transferencia de archivo a mensaje" en la página 284
Si una transferencia de archivo a mensaje falla después de que el agente haya empezado a escribir los datos de archivo en la cola de destino, el agente escribe un mensaje en la cola para indicar a la aplicación que consume los mensajes que se ha producido un fallo.

Propiedades de mensaje de MQ establecidas por MFT en mensajes escritos en las colas de destino

Ejemplo: Configuración de las propiedades definidas por el usuario en una transferencia de archivo a mensaje

Los metadatos definidos por el usuario se establecen como una propiedad de mensaje de IBM MQ en el primer mensaje que la transferencia ha grabado en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a las cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

Acerca de esta tarea

Incluya los parámetros `-qmp true` y `-md account=123456` en el mandato **fteCreateTransfer** , para establecer la propiedad `usr.account` en 123456 en la cabecera RFH2 .

Procedimiento

Escriba el siguiente mandato:

```
fteCreateTransfer -sa AGENT_JUPITER -da AGENT_SATURN -dq MY_QUEUE@MyQM
                  -qmp true -md account=123456 /tmp/source_file.txt
```

Además del conjunto estándar de propiedades de mensajes de IBM MQ, la propiedad definida por el usuario está establecida en la cabecera de mensaje del primer mensaje escrito por el agente de destino, `AGENT_SATURN`, en la cola, `MY_QUEUE`, en el gestor de colas, `MyQM`. La cabecera se establece en el siguiente valor:

```
usr.account=123456
```

El prefijo `usr` se añade al principio del nombre de los metadatos definidos por el usuario.

Conceptos relacionados

"Transferencia de datos de archivos a mensajes" en la página 269
Puede utilizar la característica de archivo a mensaje de Managed File Transfer para transferir datos desde un archivo a un único mensaje o varios mensajes, en una cola de IBM MQ.

Tareas relacionadas

"Configuración de un agente para realizar transferencias de archivo a mensaje" en la página 271

De forma predeterminada, los agentes pueden realizar transferencias de archivo a mensaje o de mensaje a archivo. Para habilitar esta función debe establecer la propiedad de agente enableQueueInputOutput en true (verdadera). Para habilitar la escritura en las colas en clúster de IBM MQ, también debe establecer la propiedad de agente enableClusterQueueInputOutput en true.

"Ejemplo: Transferencia de un único archivo en un mensaje" en la página 272
Puede especificar una cola como destino de una transferencia de archivo utilizando el parámetro **-dq** con el mandato **fteCreateTransfer**. El archivo de origen debe ser más pequeño que la longitud máxima de mensaje establecida en la cola de destino. La cola de destino no tiene que estar en el mismo gestor de colas que el gestor de colas al que se conecta el agente de destino, pero estos dos gestores de colas deben poder comunicarse.

"Ejemplo: División de un único archivo en varios mensajes por longitud" en la página 274
Puede dividir un archivo en varios mensajes de IBM MQ utilizando el parámetro **-qs** del mandato **fteCreateTransfer**. El archivo se divide en secciones de longitud fija, cada una de las cuales se escribe en un mensaje individual.

"Ejemplo: División de un archivo de texto con un delimitador de expresión regular e incluyendo el delimitador en los mensajes" en la página 277
Transferencia de un único archivo de texto en varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada e inclusión de la coincidencia de la expresión regular en los mensajes resultantes. Para hacerlo, utilice los parámetros **-dqdt** y **-qi** del mandato **fteCreateTransfer**.

"Ejemplo: división un archivo de texto en varios mensajes utilizando un delimitador de expresión regular" en la página 275
Transferencia de un único archivo de texto con varios mensajes dividiendo el archivo en cada coincidencia de una expresión regular Java determinada. Para hacerlo, utilizará el parámetro **-dqdt** del mandato **fteCreateTransfer**.

"Ejemplo: establecer propiedades de mensaje de IBM MQ en una transferencia de archivo a mensaje" en la página 279
Puede utilizar el parámetro **-qmp** en el mandato **fteCreateTransfer** para especificar si las propiedades de mensaje de IBM MQ han sido establecidas por la transferencia en el primer mensaje escrito en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

"Inicio de una nueva transferencia de archivos" en la página 221
Puede iniciar una nueva de transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos y puede elegir transferir un único archivo o varios archivos de un grupo.

Referencia relacionada

Propiedades de mensaje de MQ establecidas por MFT en mensajes escritos en las colas de destino

Ejemplo: añadir una propiedad de mensaje definida por el usuario para una transferencia de archivo a mensaje

Si utiliza Managed File Transfer para las transferencias gestionadas de mensaje a archivo, puede incluir una propiedad de mensaje definida por el usuario para el mensaje resultante.

Acerca de esta tarea

Puede utilizar cualquiera de los métodos siguientes para definir una propiedad de mensaje personalizada:

- Especifique el parámetro **-md** en la solicitud de transferencia. Para obtener más información, consulte "Ejemplo: Configuración de las propiedades definidas por el usuario en una transferencia de archivo a mensaje" en la página 281.
- Utilice una tarea Ant; puede utilizar fte:filecopy o fte:filemove. En el ejemplo siguiente se muestra una tarea fte:filecopy:

```
<project xmlns:fte="antlib:com.ibm.wmqfte.ant.taskdefs" default="complete">
```

```
<!-- Initialise the properties used in this script.-->

<target name="init" description="initialise task properties">
               <property name="src.file" value="/home/user/file1.bin"/>
               <property name="dst.queue" value="TEST.QUEUE@qm2"/>
               <fte:uuid property="job.name" length="8"
prefix="copyjob#"/>
</target>
<target name="step1" depends="init" description="transfer file">

<fte:filecopy cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
             src="agent1@qm1" dst="agent2@qm2"
            rcproperty="copy.result">

<fte:metadata>
<fte:entry name="fileName" value="${FileName}"/>
</fte:metadata>

<fte:filespec srcfilespec="${src.file}" dstqueue="${dst.queue}"
dstmsgprops="true"/>

</fte:filecopy>

</target>
</project>
```

- Utilizar un supervisor de recursos y una sustitución de variable. En el ejemplo siguiente se muestra parte de un XML de tarea de transferencia:

```
<?xml version="1.0" encoding="UTF-8"?>
<monitor:monitor
xmlns:monitor="https://www.ibm.com/xmlns/wmqfte/7.0.1/MonitorDefinition"
xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance" version="5.00"
xsi:schemaLocation="https://www.ibm.com/xmlns/wmqfte/7.0.1/MonitorDefinit ion ./Monitor.xsd">
 <name>METADATA</name>
 <pollInterval units="minutes">5</pollInterval>
 <batch maxSize="5"/>
 <agent>AGENT1</agent>
 <resources>
   <directory recursionLevel="0">e:\temp</directory>
 </resources>
 <triggerMatch>
   <conditions>
     <allOf>
       <condition>
         <fileMatch>
           <pattern>*.txt</pattern>
         </fileMatch>
       </condition>
     </allOf>
   </conditions>
 </triggerMatch>
 <tasks>
   <task>
     <name/>
     <transfer>
       <request version="5.00"
xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="FileTransfer.xsd">
         <managedTransfer>
           <originator>
             <hostName>mqjason.raleigh.ibm.com.</hostName>
             <userID>administrator</userID>
           </originator>
           <sourceAgent QMgr="AGENTQM" agent="AGENT1"/>
           <destinationAgent QMgr="AGENTQM" agent="AGENT2"/>
           <transferSet priority="0">
             <metaDataSet>
               <metaData key="FileName">${FileName}</metaData>
             </metaDataSet>
             <item checksumMethod="MD5" mode="text">
               <source disposition="delete" recursive="false">
                 <file>${FilePath}</file>
               </source>
               <destination type="queue">
                 <queue persistent="true"
setMqProps="true">TEST.QUEUE@AGENTQM</queue>
```

```
              </destination>
            </item>
          </transferSet>
          <job>
            <name>Metadata_example</name>
          </job>
        </managedTransfer>
      </request>
    </transfer>
  </task>
 </tasks>
 <originator>
   <hostName>mqjason.raleigh.ibm.com.</hostName>
   <userID>administrator</userID>
 </originator>
</monitor:monitor>
```

Tareas relacionadas

“Ejemplo: establecer propiedades de mensaje de IBM MQ en una transferencia de archivo a mensaje” en la página 279

Puede utilizar el parámetro **-qmp** en el mandato **fteCreateTransfer** para especificar si las propiedades de mensaje de IBM MQ han sido establecidas por la transferencia en el primer mensaje escrito en la cola de destino. Las propiedades de mensaje de IBM MQ permiten que una aplicación seleccione mensajes que procesar o que recupere información sobre un mensaje sin acceder a cabeceras de IBM MQ Message Descriptor (MQMD) o MQRFH2.

Referencia relacionada

Tarea Ant fte:filecopy

Tarea Ant fte:filemove

Fallo en una transferencia de archivo a mensaje

Si una transferencia de archivo a mensaje falla después de que el agente haya empezado a escribir los datos de archivo en la cola de destino, el agente escribe un mensaje en la cola para indicar a la aplicación que consume los mensajes que se ha producido un fallo.

El mensaje escrito en la cola de destino si se produce un fallo:

- está vacío
- tiene el mismo ID de grupo de IBM MQ que el mensaje anterior escrito en la cola de destino por el agente
- tiene establecido el distintivo de IBM MQ LAST_MSG_IN_GROUP
- contiene propiedades de mensaje adicionales de IBM MQ, si se habilitan las propiedades del mensaje. Si desea más información, consulte el tema Propiedades de mensaje MQ establecidas por MFT en mensajes escritos en colas de destino.

Ejemplo

Se solicita una transferencia ejecutando el siguiente mandato:

```
fteCreateTransfer -sa AGENT_JUPITER -da AGENT_SATURN -dq RECEIVING_QUEUE
                  -qmp true -qs 1K /tmp/source1.txt
```

El archivo `source1.txt` es de 48 KB. La transferencia divide este archivo en mensajes de 1 KB y escribe dichos mensajes en la cola de destino RECEIVING_QUEUE.

Mientras la transferencia está en curso, después de que el agente ha escrito 16 mensajes en RECEIVING_QUEUE, se produce un fallo en el agente de origen.

El agente escribe un mensaje en blanco en RECEIVING_QUEUE. Además del conjunto estándar de propiedades de mensaje, el mensaje en blanco tiene las siguientes propiedades definidas:

```
usr.WMQFTEResultCode = 40
usr.WMQFTESupplement = BFGTR0036I: The transfer failed to complete successfully.
```

A partir de IBM MQ 9.3.0, cuando falla una transferencia desde un archivo, debido a un error de comprobación de tamaño de delimitador, sólo se envía un mensaje vacío. Además, las propiedades de usuario se añaden a este mensaje si la causa del error de transferencia era que el delimitador sobrepasaba el tamaño establecido en el agente de destino.

Conceptos relacionados

"Transferencia de datos de archivos a mensajes" en la página 269
Puede utilizar la característica de archivo a mensaje de Managed File Transfer para transferir datos desde un archivo a un único mensaje o varios mensajes, en una cola de IBM MQ.

Tareas relacionadas

"Configuración de un agente para realizar transferencias de archivo a mensaje" en la página 271
De forma predeterminada, los agentes pueden realizar transferencias de archivo a mensaje o de mensaje a archivo. Para habilitar esta función debe establecer la propiedad de agente enableQueueInputOutput en true (verdadera). Para habilitar la escritura en las colas en clúster de IBM MQ, también debe establecer la propiedad de agente enableClusterQueueInputOutput en true.

"Inicio de una nueva transferencia de archivos" en la página 221
Puede iniciar una nueva de transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos y puede elegir transferir un único archivo o varios archivos de un grupo.

Referencia relacionada

El archivo MFT `agent.properties`

Propiedades de mensaje de MQ establecidas por MFT en mensajes escritos en las colas de destino

Transferir datos de mensajes a archivos

La función de mensaje a archivo de Managed File Transfer le permite transferir datos de uno o más mensajes en una cola de IBM MQ a un archivo , un conjunto de datos (en z/OS) o un espacio de archivos de usuario. Si tiene una aplicación que crea o procesa mensajes de IBM MQ, puede utilizar la función de mensaje a archivo de Managed File Transfer para transferir estos mensajes a un archivo en cualquier sistema de la red de Managed File Transfer.

Para obtener información sobre transferencias de mensaje a archivo, consulte "Transferencia de datos de archivos a mensajes" en la página 269.

Atención: El agente de origen para la transferencia de mensaje a archivo no puede ser un agente de puente de protocolo o un agente de puente Connect:Direct.

Puede transferir datos de mensajes de IBM MQ a un archivo. Se da soporte a los tipos siguientes de transferencias de mensaje a archivo:

- De un único mensaje a un único archivo
- De varios mensajes a un único archivo
- De varios mensajes con el mismo ID de grupo de IBM MQ a un único archivo.
- De varios mensajes a un único archivo, incluyendo un delimitador de texto o binario entre los datos de cada mensaje escrito en el archivo.

Si va a transferir archivos desde mensajes grandes, o desde muchos mensajes pequeños, es posible que tenga que cambiar algunas propiedades de IBM MQ o Managed File Transfer. Para obtener más información sobre, consulte Guía para establecer los atributos de MQ y las propiedades de MFT asociadas con el tamaño del mensaje.

En una transferencia de mensaje a archivo, el agente de origen examina los mensajes de la cola de origen, a diferencia del GET destructivo en versiones anteriores de IBM MQ. Los mensajes se eliminan de la cola de origen después de examinar todos los mensajes (de un grupo si se utiliza la agrupación de mensajes) y tras grabar los datos en el archivo de destino. Esto permite que los mensajes permanezcan en la cola

de origen si falla o se cancela una transferencia. Debido a este cambio, también se debe proporcionan autoridad BROWSE junto con la autoridad GET para ejecutar transferencias de mensaje a archivo.

Managed File Transfer compara el identificador de transferencia y el valor del atributo groupId dentro de la carga útil XML de solicitud de transferencia. Si estos dos identificadores son equivalentes, el agente de origen utiliza el identificador como una opción de coincidencia de identificador de mensaje, a diferencia de una opción de coincidencia de identificador de grupo, en el primer intento de MQGET que se realiza en la cola de entrada para llevar a cabo la transferencia de mensaje a archivo.

Tareas relacionadas

“Ejemplo: configuración de un recurso MFT” en la página 245
Puede especificar una cola de IBM MQ como el recurso que va a supervisar un supervisor de recursos utilizando el parámetro **-mq** con el mandato **fteCreateMonitor**.

Referencia relacionada

Propiedades de mensajes de MQ leídas por MFT de mensajes en las colas de origen

Guía para establecer atributos MQ y propiedades de MFT asociadas al tamaño de mensaje

Configuración de un agente para realizar transferencias de mensaje a archivo

De forma predeterminada, los agentes no pueden realizar transferencias de mensaje a archivo o archivo a mensaje. Para habilitar esta función, deberá establecer la propiedad del agente `enableQueueInputOutput` a true.

Acerca de esta tarea

Si intenta realizar una transferencia de mensaje a archivo desde un agente de origen que no tiene la propiedad `enableQueueInputOutput` establecida en true, la transferencia falla. El mensaje de registro de transferencia publicado en el gestor de cola de coordinación contiene el siguiente mensaje:

```
BFGIO0197E: An attempt to read from a queue was rejected by the source agent.
The agent must have enableQueueInputOutput=true set in the agent.properties file
to support transferring from a queue.
```

Para que el agente pueda escribir y leer en las colas, siga los siguientes pasos:

Procedimiento

1. Detenga el agente de origen utilizando el mandato **fteStopAgent**.
2. Edite el archivo `agent.properties` para incluir la línea `enableQueueInputOutput=true`.

   El archivo `agent.properties` se encuentra en el directorio *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*source_agent_name*.
3. Inicie el agente de origen utilizando el mandato **fteStartAgent**.

Conceptos relacionados

“Transferir datos de mensajes a archivos” en la página 285
La función de mensaje a archivo de Managed File Transfer le permite transferir datos de uno o más mensajes en una cola de IBM MQ a un archivo , un conjunto de datos (en z/OS) o un espacio de archivos de usuario. Si tiene una aplicación que crea o procesa mensajes de IBM MQ, puede utilizar la función de mensaje a archivo de Managed File Transfer para transferir estos mensajes a un archivo en cualquier sistema de la red de Managed File Transfer.

Tareas relacionadas

“Ejemplo: Transferencia desde una cola a un único archivo” en la página 287
Puede especificar una cola de IBM MQ como origen de una transferencia de archivo utilizando el parámetro **-sq** con el mandato **fteCreateTransfer**.

“Ejemplo: transferencia de un grupo de mensajes de una cola a un único archivo” en la página 288
Puede especificar un único grupo completo en una cola de IBM MQ como el origen de una transferencia de archivo utilizando los parámetros **-sq** y **-sqgi** con el mandato **fteCreateTransfer**.

"Ejemplo: inserción de un delimitador de texto antes de los datos de cada mensaje" en la página 289
Cuando haga transferencias en modalidad de texto de una cola de origen a un archivo, puede especificar que se inserte un delimitador de texto antes de los datos de mensajes individuales utilizando los parámetros **-sq**, **-sqdt** y **-sqdp** con el mandato **fteCreateTransfer**.

"Ejemplo: Inserción de un delimitador binario después de los datos de cada mensaje" en la página 290
Cuando haga transferencias en modalidad binaria de una cola de origen a un archivo, puede especificar que se inserte un delimitador binario después de los datos de mensajes individuales utilizando los parámetros **-sq**, **-sqdb** y **-sqdp** con el mandato **fteCreateTransfer**.

"Supervisión de una cola y utilización de sustitución de variables" en la página 251
Puede supervisar una cola y transferir mensajes de la cola supervisada a un archivo utilizando el mandato **fteCreateMonitor**. El valor de cualquier propiedad de mensaje de IBM MQ en el primer mensaje que se va a leer de la cola supervisada se puede sustituir en la definición XML de la tarea y se utiliza para definir el comportamiento de la transferencia.

"Ejemplo: Error al realizar una transferencia de un mensaje a un archivo utilizando las propiedades de mensaje de IBM MQ" en la página 294
Puede hacer que una transferencia de un mensaje a un archivo no se realice correctamente estableciendo la propiedad de mensaje `usr.UserReturnCode` IBM MQ en un valor distinto de cero. También puede especificar información suplementaria sobre la razón del error estableciendo la propiedad del mensaje `usr.UserSupplement` IBM MQ.

Referencia relacionada
El archivo MFT `agent.properties`

Ejemplo: Transferencia desde una cola a un único archivo

Puede especificar una cola de IBM MQ como origen de una transferencia de archivo utilizando el parámetro **-sq** con el mandato **fteCreateTransfer**.

Acerca de esta tarea

Los datos de origen están contenidos en tres mensajes en la cola START_QUEUE. Esta cola debe estar en el gestor de colas del agente de origen, *QM_NEPTUNE*.

Procedimiento

Escriba el siguiente mandato:

```
fteCreateTransfer -sa AGENT_NEPTUNE -sm QM_NEPTUNE
                  -da AGENT_VENUS -df /out/three_to_one.txt
                  -sq START_QUEUE
```

Los datos en los mensajes en la cola *START_QUEUE* se escriben en el archivo `/out/three_to_one.txt` en el sistema donde se está ejecutando AGENT_VENUS.

Conceptos relacionados
"Transferir datos de mensajes a archivos" en la página 285
La función de mensaje a archivo de Managed File Transfer le permite transferir datos de uno o más mensajes en una cola de IBM MQ a un archivo , un conjunto de datos (en z/OS) o un espacio de archivos de usuario. Si tiene una aplicación que crea o procesa mensajes de IBM MQ, puede utilizar la función de mensaje a archivo de Managed File Transfer para transferir estos mensajes a un archivo en cualquier sistema de la red de Managed File Transfer.

Tareas relacionadas
"Configuración de un agente para realizar transferencias de mensaje a archivo" en la página 286
De forma predeterminada, los agentes no pueden realizar transferencias de mensaje a archivo o archivo a mensaje. Para habilitar esta función, deberá establecer la propiedad del agente `enableQueueInputOutput` a true.

"Ejemplo: transferencia de un grupo de mensajes de una cola a un único archivo" en la página 288
Puede especificar un único grupo completo en una cola de IBM MQ como el origen de una transferencia de archivo utilizando los parámetros **-sq** y **-sqgi** con el mandato **fteCreateTransfer**.

"Ejemplo: inserción de un delimitador de texto antes de los datos de cada mensaje" en la página 289
Cuando haga transferencias en modalidad de texto de una cola de origen a un archivo, puede especificar que se inserte un delimitador de texto antes de los datos de mensajes individuales utilizando los parámetros **-sq**, **-sqdt** y **-sqdp** con el mandato **fteCreateTransfer**.

"Ejemplo: Inserción de un delimitador binario después de los datos de cada mensaje" en la página 290
Cuando haga transferencias en modalidad binaria de una cola de origen a un archivo, puede especificar que se inserte un delimitador binario después de los datos de mensajes individuales utilizando los parámetros **-sq**, **-sqdb** y **-sqdp** con el mandato **fteCreateTransfer**.

"Supervisión de una cola y utilización de sustitución de variables" en la página 251
Puede supervisar una cola y transferir mensajes de la cola supervisada a un archivo utilizando el mandato **fteCreateMonitor**. El valor de cualquier propiedad de mensaje de IBM MQ en el primer mensaje que se va a leer de la cola supervisada se puede sustituir en la definición XML de la tarea y se utiliza para definir el comportamiento de la transferencia.

"Ejemplo: Error al realizar una transferencia de un mensaje a un archivo utilizando las propiedades de mensaje de IBM MQ" en la página 294
Puede hacer que una transferencia de un mensaje a un archivo no se realice correctamente estableciendo la propiedad de mensaje `usr.UserReturnCode` IBM MQ en un valor distinto de cero. También puede especificar información suplementaria sobre la razón del error estableciendo la propiedad del mensaje `usr.UserSupplement` IBM MQ.

Referencia relacionada

Propiedades de mensajes de MQ leídas por MFT de mensajes en las colas de origen

fteCreateTransfer: iniciar una nueva transferencia de archivos

Ejemplo: transferencia de un grupo de mensajes de una cola a un único archivo

Puede especificar un único grupo completo en una cola de IBM MQ como el origen de una transferencia de archivo utilizando los parámetros **-sq** y **-sqgi** con el mandato **fteCreateTransfer**.

Acerca de esta tarea

En este ejemplo, existen diez mensajes en la cola START_QUEUE. Esta cola debe estar en el gestor de colas del agente de origen, *QM_NEPTUNE*. Los primeros tres mensajes pertenecen a un grupo con el ID de grupo de IBM MQ `41424b3ef3a2202020202020202020202020202020201111`; este grupo no es un grupo completo. Los siguientes cinco mensajes pertenecen a un grupo con el ID de grupo de IBM MQ `41424b3ef3a2202020202020202020202020202020202222`; este grupo está completo. Los siguientes dos mensajes pertenecen a un grupo con el ID de grupo de IBM MQ `41424b3ef3a2202020202020202020202020202020203333`; este grupo está completo.

Procedimiento

Escriba el siguiente mandato:

```
fteCreateTransfer -sa AGENT_NEPTUNE -sm QM_NEPTUNE -da AGENT_VENUS
                  -df /out/group.txt -sqgi -sq START_QUEUE
```

Los datos en los mensajes pertenecientes al primer grupo completo en la cola *START_QUEUE*, el grupo con el ID de grupo de IBM MQ `41424b3ef3a2202020202020202020202020202020202222`, serán escritos en el archivo `/out/group.txt` en el sistema donde se está ejecutando AGENT_VENUS.

Conceptos relacionados

"Transferir datos de mensajes a archivos" en la página 285
La función de mensaje a archivo de Managed File Transfer le permite transferir datos de uno o más mensajes en una cola de IBM MQ a un archivo , un conjunto de datos (en z/OS) o un espacio de archivos de usuario. Si tiene una aplicación que crea o procesa mensajes de IBM MQ, puede utilizar la función de mensaje a archivo de Managed File Transfer para transferir estos mensajes a un archivo en cualquier sistema de la red de Managed File Transfer.

Tareas relacionadas

"Configuración de un agente para realizar transferencias de mensaje a archivo" en la página 286
De forma predeterminada, los agentes no pueden realizar transferencias de mensaje a archivo o archivo a mensaje. Para habilitar esta función, deberá establecer la propiedad del agente `enableQueueInputOutput` a true.

"Ejemplo: Transferencia desde una cola a un único archivo" en la página 287
Puede especificar una cola de IBM MQ como origen de una transferencia de archivo utilizando el parámetro **-sq** con el mandato **fteCreateTransfer**.

"Ejemplo: inserción de un delimitador de texto antes de los datos de cada mensaje" en la página 289
Cuando haga transferencias en modalidad de texto de una cola de origen a un archivo, puede especificar que se inserte un delimitador de texto antes de los datos de mensajes individuales utilizando los parámetros **-sq**, **-sqdt** y **-sqdp** con el mandato **fteCreateTransfer**.

"Ejemplo: Inserción de un delimitador binario después de los datos de cada mensaje" en la página 290
Cuando haga transferencias en modalidad binaria de una cola de origen a un archivo, puede especificar que se inserte un delimitador binario después de los datos de mensajes individuales utilizando los parámetros **-sq**, **-sqdb** y **-sqdp** con el mandato **fteCreateTransfer**.

"Supervisión de una cola y utilización de sustitución de variables" en la página 251
Puede supervisar una cola y transferir mensajes de la cola supervisada a un archivo utilizando el mandato **fteCreateMonitor**. El valor de cualquier propiedad de mensaje de IBM MQ en el primer mensaje que se va a leer de la cola supervisada se puede sustituir en la definición XML de la tarea y se utiliza para definir el comportamiento de la transferencia.

"Ejemplo: Error al realizar una transferencia de un mensaje a un archivo utilizando las propiedades de mensaje de IBM MQ" en la página 294
Puede hacer que una transferencia de un mensaje a un archivo no se realice correctamente estableciendo la propiedad de mensaje `usr.UserReturnCode` IBM MQ en un valor distinto de cero. También puede especificar información suplementaria sobre la razón del error estableciendo la propiedad del mensaje `usr.UserSupplement` IBM MQ.

Referencia relacionada

fteCreateTransfer: iniciar una nueva transferencia de archivos

Ejemplo: inserción de un delimitador de texto antes de los datos de cada mensaje

Cuando haga transferencias en modalidad de texto de una cola de origen a un archivo, puede especificar que se inserte un delimitador de texto antes de los datos de mensajes individuales utilizando los parámetros **-sq**, **-sqdt** y **-sqdp** con el mandato **fteCreateTransfer**.

Acerca de esta tarea

En este ejemplo, existen cuatro mensajes en la cola START_QUEUE. Esta cola está en el gestor de cola del agente de origen, *QM_NEPTUNE*. El delimitador de texto que hay que insertar delante de los datos de cada mensaje puede expresarse como una cadena literal Java, por ejemplo: `\n\u002D\u002D\u002D\n`.

Procedimiento

Escriba el siguiente mandato:

```
fteCreateTransfer -sa AGENT_NEPTUNE -sm QM_NEPTUNE -da AGENT_VENUS -df /out/output.txt
                  -t text -sqdt "\n\u002D\u002D\u002D\n" -sqdp prefix -sq START_QUEUE
```

El delimitador de texto se añade al principio de los datos de cada uno de los cuatro mensajes en START_QUEUE mediante el agente de origen AGENT_NEPTUNE. Estos datos se escriben en el archivo de destino, `/out/output.txt`.

Conceptos relacionados

"Transferir datos de mensajes a archivos" en la página 285
La función de mensaje a archivo de Managed File Transfer le permite transferir datos de uno o más mensajes en una cola de IBM MQ a un archivo , un conjunto de datos (en z/OS) o un espacio de archivos de usuario. Si tiene una aplicación que crea o procesa mensajes de IBM MQ, puede utilizar la función de mensaje a archivo de Managed File Transfer para transferir estos mensajes a un archivo en cualquier sistema de la red de Managed File Transfer.

Tareas relacionadas

"Configuración de un agente para realizar transferencias de mensaje a archivo" en la página 286
De forma predeterminada, los agentes no pueden realizar transferencias de mensaje a archivo o archivo a mensaje. Para habilitar esta función, deberá establecer la propiedad del agente `enableQueueInputOutput` a true.

"Ejemplo: Transferencia desde una cola a un único archivo" en la página 287
Puede especificar una cola de IBM MQ como origen de una transferencia de archivo utilizando el parámetro **-sq** con el mandato **fteCreateTransfer**.

"Ejemplo: transferencia de un grupo de mensajes de una cola a un único archivo" en la página 288
Puede especificar un único grupo completo en una cola de IBM MQ como el origen de una transferencia de archivo utilizando los parámetros **-sq** y **-sqgi** con el mandato **fteCreateTransfer**.

"Ejemplo: Inserción de un delimitador binario después de los datos de cada mensaje" en la página 290
Cuando haga transferencias en modalidad binaria de una cola de origen a un archivo, puede especificar que se inserte un delimitador binario después de los datos de mensajes individuales utilizando los parámetros **-sq**, **-sqdb** y **-sqdp** con el mandato **fteCreateTransfer**.

"Supervisión de una cola y utilización de sustitución de variables" en la página 251
Puede supervisar una cola y transferir mensajes de la cola supervisada a un archivo utilizando el mandato **fteCreateMonitor**. El valor de cualquier propiedad de mensaje de IBM MQ en el primer mensaje que se va a leer de la cola supervisada se puede sustituir en la definición XML de la tarea y se utiliza para definir el comportamiento de la transferencia.

"Ejemplo: Error al realizar una transferencia de un mensaje a un archivo utilizando las propiedades de mensaje de IBM MQ" en la página 294
Puede hacer que una transferencia de un mensaje a un archivo no se realice correctamente estableciendo la propiedad de mensaje `usr.UserReturnCode` IBM MQ en un valor distinto de cero. También puede especificar información suplementaria sobre la razón del error estableciendo la propiedad del mensaje `usr.UserSupplement` IBM MQ.

Referencia relacionada

fteCreateTransfer: iniciar una nueva transferencia de archivos

Ejemplo: Inserción de un delimitador binario después de los datos de cada mensaje

Cuando haga transferencias en modalidad binaria de una cola de origen a un archivo, puede especificar que se inserte un delimitador binario después de los datos de mensajes individuales utilizando los parámetros **-sq**, **-sqdb** y **-sqdp** con el mandato **fteCreateTransfer**.

Acerca de esta tarea

En este ejemplo, existen tres mensajes en la cola START_QUEUE. Esta cola está en el gestor de cola del agente de origen, *QM_NEPTUNE*. El delimitador binario que insertar después de los datos de cada mensaje debe expresarse como una lista separada por comas de bytes hexadecimales, por ejemplo: `x34,xE7,xAE`.

Procedimiento

Escriba el siguiente mandato:

```
fteCreateTransfer -sa AGENT_NEPTUNE -sm QM_NEPTUNE -da AGENT_VENUS -df /out/binary.file
                  -sqdp postfix -sqdb x34,xE7,xAE -sq START_QUEUE
```

El agente de origen, AGENT_NEPTUNE, añade el delimitador binario se añade a los datos de cada uno de los tres mensajes en START_QUEUE. Estos datos se escriben en el archivo de destino, `/out/binary.file`.

Conceptos relacionados

“Transferir datos de mensajes a archivos” en la página 285
La función de mensaje a archivo de Managed File Transfer le permite transferir datos de uno o más mensajes en una cola de IBM MQ a un archivo , un conjunto de datos (en z/OS) o un espacio de archivos de usuario. Si tiene una aplicación que crea o procesa mensajes de IBM MQ, puede utilizar la función de mensaje a archivo de Managed File Transfer para transferir estos mensajes a un archivo en cualquier sistema de la red de Managed File Transfer.

Tareas relacionadas

“Configuración de un agente para realizar transferencias de mensaje a archivo” en la página 286
De forma predeterminada, los agentes no pueden realizar transferencias de mensaje a archivo o archivo a mensaje. Para habilitar esta función, deberá establecer la propiedad del agente `enableQueueInputOutput` a true.

“Ejemplo: Transferencia desde una cola a un único archivo” en la página 287
Puede especificar una cola de IBM MQ como origen de una transferencia de archivo utilizando el parámetro **-sq** con el mandato **fteCreateTransfer**.

“Ejemplo: transferencia de un grupo de mensajes de una cola a un único archivo” en la página 288
Puede especificar un único grupo completo en una cola de IBM MQ como el origen de una transferencia de archivo utilizando los parámetros **-sq** y **-sqgi** con el mandato **fteCreateTransfer**.

“Ejemplo: inserción de un delimitador de texto antes de los datos de cada mensaje” en la página 289
Cuando haga transferencias en modalidad de texto de una cola de origen a un archivo, puede especificar que se inserte un delimitador de texto antes de los datos de mensajes individuales utilizando los parámetros **-sq**, **-sqdt** y **-sqdp** con el mandato **fteCreateTransfer**.

“Supervisión de una cola y utilización de sustitución de variables” en la página 251
Puede supervisar una cola y transferir mensajes de la cola supervisada a un archivo utilizando el mandato **fteCreateMonitor**. El valor de cualquier propiedad de mensaje de IBM MQ en el primer mensaje que se va a leer de la cola supervisada se puede sustituir en la definición XML de la tarea y se utiliza para definir el comportamiento de la transferencia.

“Ejemplo: Error al realizar una transferencia de un mensaje a un archivo utilizando las propiedades de mensaje de IBM MQ” en la página 294
Puede hacer que una transferencia de un mensaje a un archivo no se realice correctamente estableciendo la propiedad de mensaje `usr.UserReturnCode` IBM MQ en un valor distinto de cero. También puede especificar información suplementaria sobre la razón del error estableciendo la propiedad del mensaje `usr.UserSupplement` IBM MQ.

Referencia relacionada

fteCreateTransfer: iniciar una nueva transferencia de archivos

Supervisión de una cola y utilización de sustitución de variables

Puede supervisar una cola y transferir mensajes de la cola supervisada a un archivo utilizando el mandato **fteCreateMonitor**. El valor de cualquier propiedad de mensaje de IBM MQ en el primer mensaje que se va a leer de la cola supervisada se puede sustituir en la definición XML de la tarea y se utiliza para definir el comportamiento de la transferencia.

Acerca de esta tarea

En este ejemplo, el agente de origen se denomina AGENT_VENUS, que se conecta a QM_VENUS. La cola que AGENT_VENUS supervisa se denomina START_QUEUE y se encuentra en QM_VENUS. El agente sondea la cola cada 30 minutos.

Cuando un grupo completo de mensajes se graba en la cola, la tarea de supervisor envía el grupo de mensajes a un archivo en uno de los muchos agentes de destino, que se conectan todos ellos al gestor de colas QM_MARS. El nombre del archivo al que se transfiere el grupo de mensajes se define mediante la propiedad de mensaje de IBM MQ `usr.fileName` en el primer mensaje del grupo. El nombre del agente al que se envía el grupo de mensajes se define mediante la propiedad de mensaje de IBM MQ `usr.toAgent` en el primer mensaje del grupo. Si la cabecera `usr.toAgent` no está establecida, el valor predeterminado que se va a utilizar para el agente de destino es AGENT_MAGENTA.

Cuando especifica `useGroups="true"`, si no especifica también `groupId="${GROUPID}"`, la transferencia sólo toma el primer mensaje de la cola. Por ejemplo, si está utilizando la sustitución de variables para generar el nombre de archivo (fileName), es posible que el contenido de a `.txt` no sea correcto. Esto se debe a que el nombre de archivo (fileName) lo genera el supervisor, pero la transferencia obtiene realmente un mensaje que no es el que debe generar el archivo denominado fileName.

Procedimiento

1. Cree el XML de tarea que define la tarea que el supervisor lleva a cabo cuando se desencadena.

```
<?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="AGENT_VENUS" QMgr="QM_VENUS" />
     <destinationAgent agent="${toAgent}" QMgr="QM_MARS" />
     <transferSet>
       <item mode="binary" checksumMethod="none">
         <source>
           <queue useGroups="true" groupId="${GROUPID}">START_QUEUE</queue>
         </source>
         <destination type="file" exist="overwrite">
           <file>/reports/${fileName}.rpt</file>
         </destination>
       </item>
     </transferSet>
   </managedTransfer>
</request>
```

   Las variables que se sustituyen por los valores de las cabeceras de mensajes de IBM MQ se resaltan en **negrita**. Este XML de tarea se guarda en el archivo `/home/USER1/task.xml`

2. Cree un supervisor de recursos para supervisar la cola START_QUEUE.

   Someta el siguiente mandato:

```
fteCreateMonitor -ma AGENT_VENUS -mm QM_VENUS -mq START_QUEUE
                 -mn myMonitor -mt /home/USER1/task.xml
                 -tr completeGroups -pi 30 -pu minutes -dv toAgent=AGENT_MAGENTA
```

3. Un usuario o un programa graba un grupo de mensajes en la cola START_QUEUE.

El primer mensaje de este grupo cuenta con las siguientes propiedades de mensajes de IBM MQ establecidas:

```
usr.fileName=larmer
usr.toAgent=AGENT_VIOLET
```

4. El supervisor se desencadena cuando se ha grabado el grupo completo. El agente sustituye las propiedades de mensajes de IBM MQ en el XML de tarea.

   El resultado es la transformación del XML de tarea en:

```
<?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="AGENT_VENUS" QMgr="QM_VENUS" />
     <destinationAgent agent="AGENT_VIOLET" QMgr="QM_MARS" />
     <transferSet>
       <item mode="binary" checksumMethod="none">
         <source>
           <queue useGroups="true" groupId="${GROUPID}">START_QUEUE</queue>
         </source>
         <destination type="file" exist="overwrite">
           <file>/reports/larmer.rpt</file>
         </destination>
       </item>
     </transferSet>
   </managedTransfer>
</request>
```

Resultados

Se realiza la transferencia definida por el XML de tarea. El grupo completo de mensajes leídos desde START_QUEUE por AGENT_VENUS se escribe en un archivo denominado `/reports/larmer.rpt` en el sistema donde AGENT_VIOLET está en ejecución.

Qué hacer a continuación

Transferir cada mensaje a un archivo distinto

Si desea supervisar una cola y hacer que cada mensaje se transfiera a un archivo independiente, puede utilizar una técnica similar a la que se ha descrito anteriormente en este tema.

1. Cree el supervisor tal como se ha descrito anteriormente, especificando el parámetro **-tr completeGroups** en el mandato **fteCreateMonitor** .
2. En el XML de la tarea, especifique lo siguiente:

```
<queue useGroups="true" groupId="${GROUPID}">START_QUEUE</queue>
```

Sin embargo, cuando coloque los mensajes en la cola de origen, no los coloque en un grupo de IBM MQ. Añada propiedades de mensaje de IBM MQ a cada mensaje. Por ejemplo, especifique la propiedad `usr.filename` con un valor de nombre de archivo exclusivo para cada mensaje. Esto hace que el Managed File Transfer Agent trate cada mensaje en la cola de origen como un grupo aparte.

Conceptos relacionados

“Transferir datos de mensajes a archivos” en la página 285
La función de mensaje a archivo de Managed File Transfer le permite transferir datos de uno o más mensajes en una cola de IBM MQ a un archivo , un conjunto de datos (en z/OS) o un espacio de archivos de usuario. Si tiene una aplicación que crea o procesa mensajes de IBM MQ, puede utilizar la función de mensaje a archivo de Managed File Transfer para transferir estos mensajes a un archivo en cualquier sistema de la red de Managed File Transfer.

"Personalización de las tareas del supervisor de recursos de MFT con sustitución de variables" en la página 246
Cuando se cumplen las condiciones desencadenantes de un supervisor de recursos activo, se llama a la tarea definida. Además de llamar a la tarea de transferencia o mandato con el mismo agente de destino o el mismo nombre de archivo de destino cada vez, también puede modificar la definición de tarea durante la ejecución. Para ello, inserte los nombres de variables en el XML de definición de tarea. Cuando el supervisor determina que se cumplen las condiciones de desencadenante y que la definición de tarea contiene nombres de variable, sustituye los nombres de variable por los valores de variable y, a continuación, llama a la tarea.

Qué hacer si los archivos de destino creados por una transferencia iniciada por un supervisor de recursos de cola contienen datos incorrectos

Tareas relacionadas

"Configuración de tareas de supervisión de MFT para iniciar mandatos y scripts" en la página 239
Los supervisores de recursos no se limitan a realizar transferencias de archivos y la tarea asociada. También puede configurar el supervisor para invocar otros mandatos desde el agente de supervisión, incluidos los programas ejecutables, los scripts Ant o los trabajos JCL. Para llamar a los mandatos, edite el formato XML de definición de tarea del supervisor para incluir uno o varios elementos de mandatos con los parámetros de llamada de mandatos correspondientes, como por ejemplo, argumentos y propiedades.

"Ejemplo: configuración de un recurso MFT" en la página 245
Puede especificar una cola de IBM MQ como el recurso que va a supervisar un supervisor de recursos utilizando el parámetro **-mq** con el mandato **fteCreateMonitor**.

Referencia relacionada

fteCreateMonitor: crear un supervisor de recursos de MFT

Propiedades de mensajes de MQ leídas por MFT de mensajes en las colas de origen

Ejemplo: Error al realizar una transferencia de un mensaje a un archivo utilizando las propiedades de mensaje de IBM MQ

Puede hacer que una transferencia de un mensaje a un archivo no se realice correctamente estableciendo la propiedad de mensaje `usr.UserReturnCode` IBM MQ en un valor distinto de cero. También puede especificar información suplementaria sobre la razón del error estableciendo la propiedad del mensaje `usr.UserSupplement` IBM MQ.

Acerca de esta tarea

En este ejemplo, se está realizando una transferencia entre la cola INPUT_QUEUE y el archivo `/home/user/output.file`.

Un usuario crea mensajes y los coloca en la cola INPUT_QUEUE. El agente de origen consume mensajes de la cola INPUT_QUEUE y envía los datos de la transferencia al agente de destino. El agente de destino está escribiendo estos datos en el archivo `/home/user/output.file`.

El usuario que graba mensajes en la cola INPUT_QUEUE desea detener la transferencia que está en curso y suprimir los datos que ya se han grabado en el archivo de destino.

Procedimiento

1. El usuario escribe un mensaje en la cola INPUT_QUEUE que tiene establecidas las siguientes propiedades de mensajes de IBM MQ:

   ```
   usr.UserReturnCode=1
   usr.UserSupplement="Cancelling transfer - sent wrong data."
   ```

2. El agente de origen lee las propiedades de mensajes de IBM MQ y detiene el proceso de mensajes de la cola. El agente de destino suprime los datos de archivo que se han grabado en el directorio de destino.

3. El agente de origen envía un mensaje de registro de transferencia al gestor de colas de coordinación que notifica la anomalía de la transferencia.

   El mensaje contiene la siguiente información:

```
<?xml version="1.0" encoding="UTF-8"?>
<transaction version="1.00"
               ID="414d5120514d312020202020202020207e970d4920008702" agentRole="sourceAgent"
             xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
             xsi:noNamespaceSchemaLocation="TransferLog.xsd"
             xmlns="">
  <action time="2008-11-02T21:28:09.593Z">progress</action>
  <sourceAgent agent="FTEAGENT" QMgr="QM1">
    <systemInfo architecture="x86" name="Windows 7"
             version="6.1 build 7601 Service Pack 1" />
  </sourceAgent>
  <destinationAgent agent="FTEAGENT" QMgr="QM1">
    <systemInfo architecture="x86" name="Windows 7"
            version="6.1 build 7601 Service Pack 1" />
  </destinationAgent>
  <originator>
    <hostName>reportserver.com</hostName>
    <userID>USER1</userID>
    <mqmdUserID>USER1   </mqmdUserID>
  </originator>
  <transferSet index="0" size="1"
               startTime="2008-11-02T21:28:09.281Z"
               total="1">
    <item mode="binary">
      <source>
        <queue>INPUT_QUEUE@QM1</queue>
      </source>
      <destination exist="error">
        <file>/home/user/output.file</file>
      </destination>
      <status resultCode="1">
        <supplement>Cancelling transfer - sent wrong data.</supplement>
      </status>
    </item>
  </transferSet>
</transaction>
```

Conceptos relacionados

“Transferir datos de mensajes a archivos” en la página 285
La función de mensaje a archivo de Managed File Transfer le permite transferir datos de uno o más mensajes en una cola de IBM MQ a un archivo , un conjunto de datos (en z/OS) o un espacio de archivos de usuario. Si tiene una aplicación que crea o procesa mensajes de IBM MQ, puede utilizar la función de mensaje a archivo de Managed File Transfer para transferir estos mensajes a un archivo en cualquier sistema de la red de Managed File Transfer.

Tareas relacionadas

“Configuración de un agente para realizar transferencias de mensaje a archivo” en la página 286
De forma predeterminada, los agentes no pueden realizar transferencias de mensaje a archivo o archivo a mensaje. Para habilitar esta función, deberá establecer la propiedad del agente `enableQueueInputOutput` a true.

“Ejemplo: Transferencia desde una cola a un único archivo” en la página 287
Puede especificar una cola de IBM MQ como origen de una transferencia de archivo utilizando el parámetro **-sq** con el mandato **fteCreateTransfer**.

“Ejemplo: transferencia de un grupo de mensajes de una cola a un único archivo” en la página 288
Puede especificar un único grupo completo en una cola de IBM MQ como el origen de una transferencia de archivo utilizando los parámetros **-sq** y **-sqgi** con el mandato **fteCreateTransfer**.

“Ejemplo: inserción de un delimitador de texto antes de los datos de cada mensaje” en la página 289
Cuando haga transferencias en modalidad de texto de una cola de origen a un archivo, puede especificar que se inserte un delimitador de texto antes de los datos de mensajes individuales utilizando los parámetros **-sq**, **-sqdt** y **-sqdp** con el mandato **fteCreateTransfer**.

“Ejemplo: Inserción de un delimitador binario después de los datos de cada mensaje” en la página 290

Cuando haga transferencias en modalidad binaria de una cola de origen a un archivo, puede especificar que se inserte un delimitador binario después de los datos de mensajes individuales utilizando los parámetros **-sq**, **-sqdb** y **-sqdp** con el mandato **fteCreateTransfer**.

"Supervisión de una cola y utilización de sustitución de variables" en la página 251
Puede supervisar una cola y transferir mensajes de la cola supervisada a un archivo utilizando el mandato **fteCreateMonitor**. El valor de cualquier propiedad de mensaje de IBM MQ en el primer mensaje que se va a leer de la cola supervisada se puede sustituir en la definición XML de la tarea y se utiliza para definir el comportamiento de la transferencia.

Referencia relacionada

Propiedades de mensajes de MQ leídas por MFT de mensajes en las colas de origen

El puente de protocolo

El puente de protocolo permite que la red de Managed File Transfer (MFT) acceda a los archivos almacenados en un servidor de archivos fuera de la red de MFT, bien en su dominio local o bien en una ubicación remota. Este servidor de archivos puede utilizar los protocolos de red FTP, FTPS o SFTP. Cada servidor de archivos necesita al menos un agente dedicado. El agente dedicado se conoce como el agente de puente de protocolo. Un agente de puente puede interactuar con varios servidores de archivos.

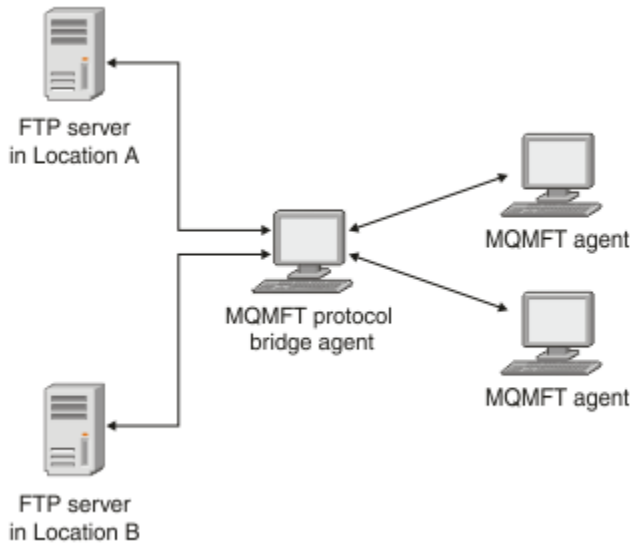
El puente de protocolo está disponible como parte del componente Servicio de Managed File Transfer. Puede tener varios agentes dedicados en un único sistema que ejecute MFT que se conecten a diferentes servidores de archivos.

Puede utilizar un agente de puente de protocolo para transferir archivos a varios puntos finales simultáneamente. MFT proporciona un archivo llamado `ProtocolBridgeProperties.xml` que puede editar para definir los diferentes servidores de archivos de protocolo a los que desea transferir archivos. El mandato **fteCreateBridgeAgent** añade los detalles del servidor de archivos de protocolo predeterminado a `ProtocolBridgeProperties.xml` automáticamente. Este archivo se describe en Formato de archivo de propiedades del puente de protocolo.

Puede utilizar el agente de puente de protocolo para realizar las siguientes acciones:

- Subir archivos de la red de MFT a un servidor remoto, mediante FTP, FTPS o SFTP.
- Descargar archivos de un servidor remoto, mediante FTP, FTPS o SFTP a la red de MFT.

Nota: El agente de puente de protocolo sólo puede soportar servidores FTP, FTPS o SFTP que permiten que se acceda a los archivos a través de la vía de acceso de archivo absoluta. Si se especifica una vía de acceso de archivo relativa en una solicitud de transferencia, el agente de puente de protocolo intentará convertir la vía de acceso relativa en una vía de acceso de archivo absoluta basándose en el directorio inicial utilizado para iniciar la sesión en el servidor de protocolo. Esos servidores de protocolo que permiten acceder a los archivos basándose sólo en el directorio actual no están soportados por el agente de puente de protocolo.



En el diagrama se muestran dos servidores FTP, en ubicaciones distintas. Los servidores FTP se utilizan para intercambiar archivos con los agentes de Managed File Transfer. El agente de puente de protocolo está entre los servidores FTP y el resto de la red de MFT y está configurado para comunicarse con ambos servidores FTP.

Asegúrese de tener otro agente en la red de MFT, además del agente de puente de protocolo. El agente de puente de protocolo es un puente sólo al servidor FTP, FTPS o SFTP y no graba archivos transferidos en el disco local. Si desea transferir archivos a o desde el servidor FTP, FTPS o SFTP, debe utilizar el agente de puente de protocolo como el destino u origen de la transferencia de archivos (en representación del servidor FTP, FTPS o SFTP) y otro agente estándar como el origen o destino correspondiente.

Cuando transfiera archivos utilizando el puente de protocolo, el puente debe tener permiso para leer el directorio de origen o de destino que contiene los archivos que desea transferir. Por ejemplo, si desea transferir archivos del directorio `/home/fte/bridge` que tiene solo permisos de ejecución (d -- x -- x -- x), las transferencias que intente desde este directorio fallan con el siguiente mensaje de error:

```
BFGBR0032E: Attempt to read filename from the protocol file server
  has failed with server error 550. Failed to open file.
```

Configuración de un agente de puente de protocolo

Un agente de puente de protocolo es como un agente de MFT estándar. Un agente de puente de protocolo se crea mediante el mandato **`fteCreateBridgeAgent`**. Puede configurar un agente de puente de protocolo utilizando el archivo `ProtocolBridgeProperties.xml`, que se describe en Formato de archivo de propiedades de puente de protocolo. Si utiliza una versión anterior, configure el agente utilizando las propiedades de puente de protocolo específicas descritas en Propiedades de agente avanzadas: puente de protocolo y Propiedades de agente avanzadas: Registro del agente de puente de protocolo. Para todas las versiones, también puede configurar una correlación de credenciales como se describe en “Correlación de credenciales para un servidor de archivos” en la página 305. Después de que haya configurado un agente de puente de protocolo para un servidor de archivos de protocolo específico, podrá utilizar ese agente con esa finalidad únicamente.

Recuperación del puente de protocolo

Si el agente de puente de protocolo no puede conectarse al servidor de archivos porque el servidor de archivos está no disponible, todas las peticiones de transferencia de archivos se ponen en cola hasta que el servidor de archivos queda disponible. Si el agente de puente de protocolo no puede conectarse al servidor de archivos porque el agente está utilizando las credenciales incorrectas, la transferencia fallará y el mensaje del registro de transferencias reflejará este error. Si el agente de puente de protocolo finaliza

por cualquier motivo, todas las transferencias de archivos solicitadas se retienen y continúan cuando se reinicia el puente de protocolo.

Durante la transferencia de archivos, los archivos suelen grabarse como archivos temporales en el destino y, a continuación, se renombran cuando termina la transferencia. Sin embargo, si el destino de la transferencia es un servidor de archivos de protocolo que está configurado como de grabación limitada (los usuarios pueden subir archivos al servidor de archivos de protocolo pero no pueden cambiar esos archivos subidos de ninguna manera; en realidad los usuarios sólo pueden grabar una vez), los archivos transferidos se graban directamente en el destino. Esto significa que si se produce un problema durante la transferencia, los archivos grabados parcialmente permanecen en el servidor de archivos de protocolo de destino y Managed File Transfer no puede suprimirlos ni editarlos. En esta situación, la transferencia no se realiza satisfactoriamente.

Tareas relacionadas

"Ejemplo: Cómo configurar un agente de puente de protocolo para utilizar credenciales de clave privada con un servidor SFTP UNIX" en la página 310
Este ejemplo muestra cómo puede generar y configurar el archivo `ProtocolBridgeCredentials.xml`. Este ejemplo es un ejemplo típico y los detalles pueden variar en función de la plataforma, pero los principios siguen siendo los mismos.

"Definición de propiedades para servidores de archivos de protocolo utilizando el archivo ProtocolBridgeProperties.xml" en la página 298
Defina las propiedades de uno o más servidores de archivos de protocolo a los que desea transferir archivos y mediante el archivo `ProtocolBridgeProperties.xml`, que Managed File Transfer proporciona en el directorio de configuración del agente.

Referencia relacionada

fteCreateBridgeAgent (crear y configurar un agente de puente de protocolo de MFT)

"Correlación de credenciales para un servidor de archivos" en la página 305
Correlacione credenciales de usuario de Managed File Transfer con credenciales de usuario del servidor de archivos, utilizando la función de correlación de credenciales predeterminada del agente de puente de protocolo o escribiendo su propia salida de usuario. Managed File Transfer proporciona una salida de usuario de ejemplo que realiza la correlación de credenciales de usuario.

Interfaz ProtocolBridgeCredentialExit.java

Ejemplo de salida de usuario de credenciales de puente de protocolo

Soporte de servidor FTPS por el puente de protocolo

Definición de propiedades para servidores de archivos de protocolo utilizando el archivo `ProtocolBridgeProperties.xml`

Defina las propiedades de uno o más servidores de archivos de protocolo a los que desea transferir archivos y mediante el archivo `ProtocolBridgeProperties.xml`, que Managed File Transfer proporciona en el directorio de configuración del agente.

Acerca de esta tarea

El mandato **`fteCreateBridgeAgent`** crea el archivo `ProtocolBridgeProperties.xml` en el directorio de configuración del agente *`MQ_DATA_PATH`*`/mqft/config/`*`coordination_queue_manager`*`/agents/`*`bridge_agent_name`*. El mandato también crea una entrada en el archivo para el servidor de archivos de protocolo predeterminado, si se especificó un valor predeterminado cuando se ejecutó el mandato.

El mensaje BFGCL0392I proporciona la ubicación del archivo `ProtocolBridgeProperties.xml`.

```
<?xml version="1.0" encoding="IBM-1047"?>
<!--
This ProtocolBridgeProperties.xml file determines the protocol servers that will be accessed by
the
MQMFT protocol bridge agent.

Each protocol server is defined using either a <tns:ftpServer>, <tns:ftpsServer>, or
```

```
<tns:sftpServer>
element - depending on the protocol used to communicate with the server.  When the protocol
bridge agent participates in a managed file transfer it will determine which server to used
based on
the prefix (if any) present on the file path.  For example a file path of 'server1:/home/user/
file.txt' would
be interpreted as a request to transfer /home/user/file.txt using 'server1'.  The server name
is compared
to the 'name' attribute of each <tns:ftpServer>, <tns:ftpsServer> or <tns:sftpServer> element
in this
XML document and the first match is used to determine which protocol server the protocol bridge
agent will connect to.  If no match is found then the managed file transfer operation will fail.

If a file path is not prefixed with a server name, for example '/home/user/file.txt' then this
XML
document can specify a default server to use for the managed file transfer.  To specify a
default server use the <tns:defaultServer> element as the first element inside the
<tns:serverProperties>
element.  The default server will be used whenever the protocol bridge agent participates in
a managed file transfer for file names which do not specify a prefix.

An optional <tns:limits> element can be specified within each server definition. This element
contains
attributes that govern the amount of resources used by each defined server.

An optional <tns:credentialsFile> element can be specified within each serverProperties
definition. This
element contains a path to a file containing credentials to be used when connecting to defined
servers.

An example ProtocolBridgeProperties.xml file is as follows:

<?xml version="1.0" encoding="UTF-8"?>
<tns:serverProperties xmlns:tns="http://wmqfte.ibm.com/ProtocolBridgeProperties"
    xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://wmqfte.ibm.com/ProtocolBridgeProperties
ProtocolBridgeProperties.xsd">

    <tns:credentialsFile path="$HOME/ProtocolBridgeCredentials.xml" />

    <tns:defaultServer name="myFTPserver" />

    <tns:ftpServer name="myFTPserver" host="windows.hursley.ibm.com" port="1234"
platform="windows"
        timeZone="Europe/London" locale="en_GB" fileEncoding="UTF-8"
        listFormat="unix" limitedWrite="false">

        <tns:limits maxListFileNames="100" maxListDirectoryLevels="999999999"
            maxReconnectRetry="2"  reconnectWaitPeriod="10"
            maxSessions="60" socketTimeout="30" />

    </tns:ftpServer>

    <tns:ftpsServer name="myFTPSserver" host="unix.hursley.ibm.com" platform="unix"
        timeZone="Europe/London" locale="en_GB" fileEncoding="UTF8"
        listFormat="unix" limitedWrite="false" ftpsType="explicit"
        trustStore="C:\FTE\keystores\myFTPSserver\FTPSKeyStore.jks"
trustStorePassword="password">

        <tns:limits maxReconnectRetry="10" connectionTimeout="10"/>

    </tns:ftpsServer>

    <tns:sftpServer name="mySFTPserver" host="windows.hursley.ibm.com" platform="windows"
        timeZone="Europe/London" locale="en_GB" fileEncoding="UTF-8"
        limitedWrite="false">

        <tns:limits connectionTimeout="60"/>

    </tns:sftpServer>

</tns:serverProperties>

This example shows the outermost <tns:serverProperties> element which must exist for the
document to
be valid, an optional <tns:defaultServer> element, as well as definitions for an FTP, FTPS and
SFTP server.

The attributes of the <tns:ftpServer>, <tns:ftpsServer> and <tns:sftpServer> elements determine
the
characteristics of the connection established to the server.  These attributes correspond to
the command
```

```
line parameters for the 'fteCreateBridgeAgent' command.

The following attributes are valid for all of the <tns:ftpServer>, <tns:ftpsServer> and
<tns:sftpServer>
elements: name, host, port, platform, fileEncoding, limitedWrite and controlEncoding.

The following attributes are valid for the <tns:ftpServer> and <tns:ftpsServer> elements:
timezone, locale,
listFormat, listFileRecentDateFormat, listFileOldDateFormat, and monthShortNames.

The following attributes are valid for the <tns:ftpServer> element only: passiveMode

The following attributes are valid for the <tns:ftpsServer> element only: ftpsType, trustStore,
trustStorePassword,
trustStoreType, keyStore, keyStorePassword, keyStoreType, ccc, protFirst, auth, and
connectTimeout.

The following attributes are valid for the <tns:limits> element within all of the
<tns:ftpServer>, <tns:ftpsServer>
and <tns:sftpServer> elements: maxListFileNames, maxListDirectoryLevels, maxReconnectRetry,
reconnectWaitPeriod,
maxSessions and socketTimeout

-->
<tns:serverProperties xmlns:tns="http://wmqfte.ibm.com/ProtocolBridgeProperties"
        xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://wmqfte.ibm.com/ProtocolBridgeProperties
ProtocolBridgeProperties.xsd">

    <!-- By default the location of the credentials file is in the home directory of the user
that started the -->
    <!-- protocol bridge agent. If you wish to specify a different location use the
credentialsFile element to -->
    <!-- do this. For
example:                                                                              -->
    <!-- <tns:credentialsFile path="/test/
ProtocolBridgeCredentials.xml"/>                                  -->

    <tns:defaultServer name="WINMVSCA.HURSLEY.IBM.COM" />
    <tns:ftpServer name="WINMVSCA.HURSLEY.IBM.COM" host="WINMVSCA.HURSLEY.IBM.COM"
platform="unix"
                timeZone="Europe/London" locale="en-GB" fileEncoding="US-ASCII"
                listFormat="unix" limitedWrite="false"  />

    <!-- Define servers here -->
</tns:serverProperties>
```

El mandato puede generar el mensaje BFGCL0532I siguiente:

```
Para que este agente funcione, debe crearse manualmente un archivo de credenciales adicionales.
De forma predeterminada, el nombre de este archivo es ProtocolBridgeCredentials.xml y se
encuentra en el directorio
de inicio del usuario que inicia el agente. Por ejemplo, si este usuario ha iniciado el agente,
la ubicación será: $HOME/ProtocolBridgeCredentials.xml
```

Si utiliza un archivo de credenciales:

1. Consulte el texto siguiente para obtener más información acerca de cómo crear uno.
2. El archivo de credenciales debe estar en un directorio con permisos restringidos. Por ejemplo, otros usuarios no deben tener acceso de lectura.
3. Especifique la ubicación del directorio del archivo de credenciales en la variable de entorno $HOME para el ID de usuario del agente iniciado, o edite el archivo `ProtocolBridgeProperties.xml` y especifique la ubicación en:

   ```
   <tns:credentialsFile path="/test/ProtocolBridgeCredentials.xml"/>
   ```

Si desea añadir servidores de protocolo no predeterminados adicionales, edite este archivo para definir sus propiedades. Este ejemplo añade un servidor FTP adicional.

Nota: El agente de puente de protocolo no da soporte al bloqueo de archivo. Esto se debe a que Managed File Transfer no da soporte al mecanismo de bloqueo de archivo en un servidor de archivos.

Procedimiento

1. Defina un servidor de archivos de protocolo insertando las líneas siguientes en el archivo como elemento hijo de <tns:serverProperties>:

```
<tns:ftpServer name="myserver" host="myhost.hursley.ibm.com" port="1234"
   platform="windows"
                           timeZone="Europe/London" locale="en-GB" fileEncoding="UTF-8"
                          listFormat="unix" limitedWrite="false" >
<tns:limits maxListFileNames="10" maxListDirectoryLevels="500"/>
```

2. A continuación, cambie el valor de los atributos:
   - name es el nombre de su servidor de archivos de protocolo
   - host es el nombre de host o dirección IP del servidor de archivos de protocolo
   - port es el número de puerto del servidor de archivos de protocolo
   - platform es la plataforma en la que se ejecuta el servidor de archivos de protocolo
   - timeZone es el huso horario en el que se ejecuta el servidor de archivos de protocolo
   - locale es el idioma utilizado en el servidor de archivos de protocolo
   - fileEncoding es la codificación de caracteres del servidor de archivos de protocolo
   - listFormat es el formato de listado de archivos devuelto por el servidor de archivos de protocolo
   - limitedWrite determina si se debe seguir la modalidad predeterminada al grabar en un servidor de archivos, que consiste en crear un archivo temporal y luego renombrar ese archivo cuando la transferencia se ha completado. Para un servidor de archivos que está configurado como de sólo grabación, el archivo se crea directamente con su nombre final. El valor de esta propiedad puede ser true o false. El atributo limitedWrite y la propiedad de agente doNotUseTempOutputFile se usan juntos en el caso de tratarse de agentes de puente de protocolo. Si desea utilizar archivos temporales, no debe establecer el valor de doNotUseTempOutputFile, y debe establecer el valor de limitedWrite en false. Cualquier otra combinación de valores significa que los archivos temporales no se utilizarán.
   - maxListFileNames es el número máximo de nombres que se recopilan al explorar un directorio en el servidor de archivos de protocolo para detectar nombres de archivo.
   - maxListDirectoryLevels es el número máximo de niveles de directorio en los que buscar de forma recursiva al explorar un directorio en el servidor de archivos de protocolo para detectar nombres de archivo.

   Si desea más detalles sobre estos atributos, incluyendo si son necesarios u opcionales y sus valores predeterminados, consulte Formato de archivo de propiedades de puente de protocolo.

Referencia relacionada

Formato del archivo de propiedades de puente de protocolo

Expresiones regulares utilizadas por MFT

Buscar propiedades de servidor de archivos de protocolo: ProtocolBridgePropertiesExit2

Si tiene un gran número de servidores de archivos de protocolo, puede implementar la interfaz com.ibm.wmqfte.exitroutine.api.ProtocolBridgePropertiesExit2 para buscar propiedades de servidor de archivos de protocolo a las que se hace referencia en las transferencias. Puede implementar esta interfaz de preferencia para mantener un archivo ProtocolBridgeProperties.xml.

Acerca de esta tarea

Managed File Transfer proporciona una salida de usuario de ejemplo que busca propiedades de servidor de archivos de protocolo. Para obtener más información, consulte “Utilización de la salida de usuario de ejemplo para buscar propiedades de servidor de archivos de protocolo” en la página 302.

Cualquier salida de usuario que busque propiedades de puente de protocolo debe implementar la interfaz `com.ibm.wmqfte.exitroutine.api.ProtocolBridgePropertiesExit2`. Si desea más información, consulte Interfaz ProtocolBridgePropertiesExit.java.

Puede encadenar varias salidas de propiedades de servidor de protocolo de forma parecida a otras salidas de usuario. Las salidas se invocan en el orden en que se especifican utilizando la propiedad `protocolBridgePropertiesExitClasses` en el archivo de propiedades de agente. Los métodos initialize todos regresan por separado y si uno o más devuelve un valor de false, el agente no se inicia. El error se comunica en el registro de sucesos del agente.

Sólo se devuelve un resultado global para los métodos getProtocolServerProperties de todas las salidas. Si el método devuelve un objeto de propiedades como el código de resultado, este valor es el resultado devuelto y no se llama a los métodos getProtocolServerProperties de las salidas subsiguientes. Si el método devuelve un valor de null como el código de resultado, se llama al método getProtocolServerProperties de la salida siguiente. Si no hay ninguna salida subsiguiente, se devuelve el resultado null. El agente de puente de protocolo considera un código de resultado global de null como un error de búsqueda.

Se recomienda que utilice la interfaz ProtocolBridgePropertiesExit2.java, pero para obtener información sobre la interfaz ProtocolBridgePropertiesExit.java, consulte “Buscar propiedades de servidor de archivos de protocolo: ProtocolBridgePropertiesExit” en la página 303.

Para ejecutar la salida, realice los pasos siguientes:

Procedimiento

1. Compile la salida de usuario de propiedades de servidor de protocolo.
2. Cree un archivo de archivado Java (JAR) que contenga la salida compilada y su estructura de paquetes.
3. Coloque el archivo JAR que contiene la clase de salida en el directorio `exits` del agente de puente de protocolo. Este directorio se encuentra en el directorio *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*bridge_agent_name*.
4. Edite el archivo de propiedades del agente de puente de protocolo para incluir la propiedad `protocolBridgePropertiesExitClasses`. Para el valor de esta propiedad, especifique una lista separada por comas de clases que implementan una salida de usuario de propiedades de servidor de puente de protocolo. Las clases de salida se invocan en el orden en que están especificadas en esta lista. Para obtener más información, consulte El archivo MFT `agent.properties`.
5. Opcionalmente, puede especificar la propiedad `protocolBridgePropertiesConfiguration`. El valor que especifique para esta propiedad se pasa como una Serie al método initialize() de las clases de salida especificadas mediante `protocolBridgePropertiesExitClasses`. Para obtener más información, consulte El archivo MFT `agent.properties`.

Utilización de la salida de usuario de ejemplo para buscar propiedades de servidor de archivos de protocolo

Managed File Transfer proporciona una salida de usuario de ejemplo que busca propiedades de servidor de archivos de protocolo.

Acerca de esta tarea

Se proporciona una salida de usuario de ejemplo que busca propiedades de puente de protocolo en el directorio *MQ_INSTALLATION_PATH*`/mqft/samples/protocolBridge` y en el tema Ejemplo de salida de usuario de propiedades de puente de protocolo.

La salida `SamplePropertiesExit2.java` lee un archivo de propiedades que contiene propiedades para servidores de protocolo. El formato de cada entrada del archivo de propiedades es el siguiente:

```
serverName=type://host:port
```

La ubicación del archivo de propiedades se determina a partir de la propiedad de agente de puente de protocolo `protocolBridgePropertiesConfiguration`.

Para ejecutar la salida de usuario de ejemplo, realice los pasos siguientes:

Procedimiento

1. Compile el archivo `SamplePropertiesExit2.java`.
2. Cree un archivo JAR que contenga la salida compilada y su estructura de paquetes.
3. Coloque el archivo JAR en el directorio *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*bridge_agent*`/exits`.
4. Edite el archivo *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*bridge_agent_name*`/agent.properties` para que contenga la línea:

   ```
   protocolBridgePropertiesExitClasses=SamplePropertiesExit2
   ```

5. Cree un archivo de propiedades de puente de protocolo, por ejemplo `protocol_bridge_properties.properties`, en el directorio *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*bridge_agent*. Edite este archivo para incluir entradas en el formato:

   ```
   serverName=type://host:port
   ```

6. Edite el archivo *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*bridge_agent*`/agent.properties` para que contenga la línea:

   ```
   protocolBridgePropertiesConfiguration=MQ_DATA_PATH/mqft/config/coordination_queue_manager/
   agents/bridge_agent/protocol_bridge_properties.properties
   ```

   Debe utilizar la vía de acceso absoluta al archivo `protocol_bridge_properties.properties`.
7. Inicie el agente de puente de protocolo utilizando el mandato **fteStartAgent**.

Conceptos relacionados

“El puente de protocolo” en la página 296
El puente de protocolo permite que la red de Managed File Transfer (MFT) acceda a los archivos almacenados en un servidor de archivos fuera de la red de MFT, bien en su dominio local o bien en una ubicación remota. Este servidor de archivos puede utilizar los protocolos de red FTP, FTPS o SFTP. Cada servidor de archivos necesita al menos un agente dedicado. El agente dedicado se conoce como el agente de puente de protocolo. Un agente de puente puede interactuar con varios servidores de archivos.

Referencia relacionada

Interfaz ProtocolBridgePropertiesExit.java

Ejemplo de salida de usuario de propiedades de puente de protocolo

El archivo MFT `agent.properties`

fteCreateBridgeAgent (crear y configurar un agente de puente de protocolo de MFT)

Buscar propiedades de servidor de archivos de protocolo: ProtocolBridgePropertiesExit

Si tiene un gran número de servidores de archivos de protocolo, puede implementar la interfaz `com.ibm.wmqfte.exitroutine.api.ProtocolBridgePropertiesExit` para buscar propiedades de servidor de archivos de protocolo a las que se hace referencia en las transferencias.

Acerca de esta tarea

Puede implementar la interfaz `com.ibm.wmqfte.exitroutine.api.ProtocolBridgePropertiesExit` en lugar de mantener un archivo `ProtocolBridgeProperties.xml`. Utilice la interfaz ProtocolBridgePropertiesExit2.java .

El método **getCredentialLocation** de ProtocolBridgePropertiesExit2.java utiliza la ubicación predeterminada del archivo `ProtocolBridgeCredentials.xml` , que es el directorio de inicio.

Cualquier salida de usuario que busque propiedades de puente de protocolo debe implementar la interfaz `com.ibm.wmqfte.exitroutine.api.ProtocolBridgePropertiesExit`:

```
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 ProtocolBridgePropertiesExit {

    /**
     * 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);

    /**
     * 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);

}
```

Puede encadenar varias salidas de propiedades de servidor de protocolo de forma parecida a otras salidas de usuario. Las salidas se invocan en el orden en que se especifican utilizando la propiedad `protocolBridgePropertiesExitClasses` en el archivo de propiedades de agente. Los métodos initialize todos regresan por separado y si uno o más devuelve un valor de false, el agente no se inicia. El error se comunica en el registro de sucesos del agente.

Sólo se devuelve un resultado global para los métodos getProtocolServerProperties de todas las salidas. Si el método devuelve un objeto de propiedades como el código de resultado, este valor

es el resultado devuelto y no se llama a los métodos getProtocolServerProperties de las salidas subsiguientes. Si el método devuelve un valor de null como el código de resultado, se llama al método getProtocolServerProperties de la salida siguiente. Si no hay ninguna salida subsiguiente, se devuelve el resultado null. El agente de puente de protocolo considera un código de resultado global de null como un error de búsqueda.

Procedimiento

Para ejecutar la salida, realice los pasos siguientes:

1. Compile la salida de usuario de propiedades de servidor de protocolo.
2. Cree un archivo de archivado Java (JAR) que contenga la salida compilada y su estructura de paquetes.
3. Coloque el archivo JAR que contiene la clase de salida en el directorio `exits` del agente de puente de protocolo.

   Este directorio se encuentra en el directorio *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*bridge_agent_name*.
4. Edite el archivo de propiedades del agente de puente de protocolo para incluir la propiedad `protocolBridgePropertiesExitClasses`.

   Para el valor de esta propiedad, especifique una lista separada por comas de clases que implementan una salida de usuario de propiedades de servidor de puente de protocolo. Las clases de salida se invocan en el orden en que están especificadas en esta lista. Para obtener más información, consulte El archivo MFT `agent.properties`.
5. Opcionalmente, puede especificar la propiedad `protocolBridgePropertiesConfiguration`.

   El valor que especifique para esta propiedad se pasa como una Serie al método initialize() de las clases de salida especificadas mediante `protocolBridgePropertiesExitClasses`. Para obtener más información, consulte El archivo MFT `agent.properties`.

Correlación de credenciales para un servidor de archivos

Correlacione credenciales de usuario de Managed File Transfer con credenciales de usuario del servidor de archivos, utilizando la función de correlación de credenciales predeterminada del agente de puente de protocolo o escribiendo su propia salida de usuario. Managed File Transfer proporciona una salida de usuario de ejemplo que realiza la correlación de credenciales de usuario.

Conceptos relacionados

“El puente de protocolo” en la página 296
El puente de protocolo permite que la red de Managed File Transfer (MFT) acceda a los archivos almacenados en un servidor de archivos fuera de la red de MFT, bien en su dominio local o bien en una ubicación remota. Este servidor de archivos puede utilizar los protocolos de red FTP, FTPS o SFTP. Cada servidor de archivos necesita al menos un agente dedicado. El agente dedicado se conoce como el agente de puente de protocolo. Un agente de puente puede interactuar con varios servidores de archivos.

Tareas relacionadas

“Correlación de credenciales para un servidor de archivos utilizando el archivo ProtocolBridgeCredentials.xml” en la página 306
Correlacione credenciales de usuario de Managed File Transfer con credenciales de usuario del servidor de archivos, utilizando la función de correlación de credenciales predeterminada del agente de puente de protocolo. Managed File Transfer proporciona un archivo XML que puede editar para incluir información sobre las credenciales.

“Correlacionar credenciales para un servidor de archivos utilizando clases de salida” en la página 308
Si no desea utilizar la función de correlación de credenciales predeterminada del agente de puente de protocolo, puede correlacionar credenciales de usuario de Managed File Transfer con credenciales de usuario del servidor de archivos escribiendo su propia salida de usuario. Si configura salidas de usuario de correlación de credenciales, éstas sustituyen a la función de correlación de credenciales predeterminada.

“Ejemplo: Cómo configurar un agente de puente de protocolo para utilizar credenciales de clave privada con un servidor SFTP UNIX” en la página 310

Este ejemplo muestra cómo puede generar y configurar el archivo `ProtocolBridgeCredentials.xml`. Este ejemplo es un ejemplo típico y los detalles pueden variar en función de la plataforma, pero los principios siguen siendo los mismos.

Referencia relacionada

Interfaz ProtocolBridgeCredentialExit.java

Ejemplo de salida de usuario de credenciales de puente de protocolo

El archivo MFT `agent.properties`

Correlación de credenciales para un servidor de archivos utilizando el archivo `ProtocolBridgeCredentials.xml`

Correlacione credenciales de usuario de Managed File Transfer con credenciales de usuario del servidor de archivos, utilizando la función de correlación de credenciales predeterminada del agente de puente de protocolo. Managed File Transfer proporciona un archivo XML que puede editar para incluir información sobre las credenciales.

Acerca de esta tarea

El usuario debe crear manualmente el archivo `ProtocolBridgeCredentials.xml`. De forma predeterminada, la ubicación de este archivo es el directorio inicial del usuario que ha iniciado el agente de puente de protocolo, pero puede almacenarse en cualquier parte del sistema de archivos al que pueda acceder el agente. Para especificar una ubicación diferente, añada el elemento <credentialsFile> al archivo `ProtocolBridgeProperties.xml`. Por ejemplo,

```
<tns:credentialsFile path="/example/path/to/ProtocolBridgeCredentials.xml"/>
```

Antes de poder utilizar un agente de puente de protocolo, configure la correlación de credenciales editando este archivo para incluir información de host, de usuario y de credenciales. Si desea más información y ejemplos, consulte Formato de archivo de credenciales de puente de protocolo.

Procedimiento

1. • Edite la línea <tns:server name="*server name*"> para cambiar el valor del atributo de nombre por el nombre de servidor en el archivo `ProtocolBridgeProperties.xml` .

   Puede utilizar el atributo de patrón (pattern) para especificar que ha utilizado un nombre de servidor que contiene comodines o expresiones regulares. Por ejemplo,

   ```
   <tns:server name="serverA*" pattern="wildcard">
   ```

2. Inserte el ID de usuario y la información de credenciales en el archivo como elementos hijo de <tns:server>.

   Puede insertar uno o varios de los siguientes elementos en el archivo:

   • Si el servidor de archivos de protocolo es un servidor FTP, FTPS o SFTP, puede utilizar contraseñas para autenticar el usuario que solicita la transferencia. Inserte las siguientes líneas en el archivo:

```
<tns:user name="FTE User ID"
  serverUserId="Server User ID"
  serverPassword="Server Password">
</tns:user>
```

   Cambie el valor de los atributos.

   – name es una expresión regular Java para comparar el ID de usuario MQMD asociado con la solicitud de transferencia de MFT

   – `serverUserId` es el valor que se pasa al servidor de archivos de protocolo como ID de usuario de inicio de sesión. Si el atributo `serverUserId` no se especifica, se utiliza en su lugar el ID de usuario MQMD asociado a la solicitud de transferencia de MFT.

– `serverPassword` es la contraseña que se asocia con `serverUserId`.

El atributo `name` puede contener una expresión regular Java. El correlacionador de credenciales intenta emparejar el ID de usuario MQMD de la solicitud de transferencia de MFT con esta expresión regular. El agente de puente de protocolo intenta emparejar el ID de usuario MQMD con la expresión regular del atributo name de los elementos <tns:user> en el orden en que los elementos existen en el archivo. Cuando se encuentra una coincidencia, el agente de puente de protocolo no busca más coincidencias. Si se encuentra una coincidencia, los valores `serverUserId` y `serverPassword` correspondientes se pasan al servidor de archivos de protocolo como el ID de usuario y la contraseña de inicio de sesión. Las coincidencias de ID de usuario MQMD distinguen entre mayúsculas y minúsculas.

- Si el servidor de archivos de protocolo es un servidor SFTP, puede utilizar claves públicas y privadas para autenticar el usuario que solicita la transferencia. Inserte las líneas siguientes en el archivo y cambie el valor de los atributos. El elemento <tns:user> puede contener uno o varios elementos <tns:privateKey>.

```
<tns:user name="FTE User ID"
    serverUserId="Server User ID"
    hostKey="Host Key">
    <tns:privateKey associationName="association"
      keyPassword="Private key password">
    Private key file text
    </tns:privateKey>
</tns:user>
```

– `name` es una expresión regular Java para comparar el ID de usuario MQMD asociado con la solicitud de transferencia de MFT

– `serverUserId` es el valor que se pasa al servidor de archivos de protocolo como ID de usuario de inicio de sesión. Si el atributo `serverUserId` no se especifica, se utiliza en su lugar el ID de usuario MQMD asociado a la solicitud de transferencia de MFT.

– `hostKey` es la clave esperada que se devuelve del servidor al iniciar la sesión

– `key` es la clave privada de `serverUserId`

– `keyPassword` es la contraseña de la clave para generar claves públicas

– `associationName` es un valor que se utiliza para la identificación para el rastreo y registro cronológico

El atributo `name` puede contener una expresión regular Java. El correlacionador de credenciales intenta emparejar el ID de usuario MQMD de la solicitud de transferencia de MFT con esta expresión regular. El agente de puente de protocolo intenta emparejar el ID de usuario MQMD con la expresión regular del atributo name de los elementos <tns:user> en el orden en que los elementos existen en el archivo. Cuando se encuentra una coincidencia, el agente de puente de protocolo no busca más coincidencias. Si se encuentra una coincidencia, se utilizan los valores de `serverUserId` y `key` correspondientes para autenticar el usuario MFT con el servidor de archivos de protocolo. Las coincidencias de ID de usuario MQMD distinguen entre mayúsculas y minúsculas.

Para obtener más información sobre la utilización de claves privadas con un agente de puente de protocolo, consulte “Ejemplo: Cómo configurar un agente de puente de protocolo para utilizar credenciales de clave privada con un servidor SFTP UNIX” en la página 310.

Nota: z/OS

Cuando la solicitud de transferencia se graba en la cola de mandatos, es posible que el ID de usuario MQMD se convierta a mayúsculas si la cola de mandatos del agente de origen se encuentra en un sistema z/OS o IBM i. En consecuencia, el ID de usuario de MQMD del mismo usuario de origen puede llegar a la salida de credenciales con las letras originales o convertidas a mayúsculas, en función del agente de origen especificado en la solicitud de transferencia. La salida de correlación de credenciales predeterminada realiza una comprobación de coincidencia de mayúsculas/minúsculas en el ID de usuario de MQMD suministrado, que podría tener que habilitar en el archivo de correlación

Referencia relacionada

Formato del archivo de credenciales de puente de protocolo
Formato del archivo de propiedades de puente de protocolo
Expresiones regulares utilizadas por MFT

Correlacionar credenciales para un servidor de archivos utilizando clases de salida

Si no desea utilizar la función de correlación de credenciales predeterminada del agente de puente de protocolo, puede correlacionar credenciales de usuario de Managed File Transfer con credenciales de usuario del servidor de archivos escribiendo su propia salida de usuario. Si configura salidas de usuario de correlación de credenciales, éstas sustituyen a la función de correlación de credenciales predeterminada.

Acerca de esta tarea

Managed File Transfer proporciona una salida de usuario de ejemplo que realiza la correlación de credenciales de usuario. Para obtener más información, consulte "Utilización de la salida de usuario de credenciales de puente de protocolo de ejemplo" en la página 309.

Una salida de usuario para correlacionar credenciales de puente de protocolo debe implementar una de las siguientes interfaces:

- `com.ibm.wmqfte.exitroutine.api.ProtocolBridgeCredentialExit`, que permite a un agente de puente de protocolo transferir archivos a y desde un servidor de archivos de protocolo predeterminado
- `com.ibm.wmqfte.exitroutine.api.ProtocolBridgeCredentialExit2`, que le permite transferir archivos a y desde varios puntos finales.

La interfaz `com.ibm.wmqfte.exitroutine.api.ProtocolBridgeCredentialExit2` contiene la misma función que `com.ibm.wmqfte.exitroutine.api.ProtocolBridgeCredentialExit` y también incluye una función ampliada. Si desea más información, consulte Interfaz ProtocolBridgeCredentialExit.java e Interfaz ProtocolBridgeCredentialExit2.java.

Las salidas de credenciales pueden encadenarse juntas de forma parecida a otras salidas de usuario. Las salidas se invocan en el orden en que se especifican utilizando la propiedad `protocolBridgeCredentialConfiguration` en el archivo de propiedades de agente. Los métodos initialize todos regresan por separado y si uno o más devuelve un valor de false, el agente no se inicia. El error se comunica en el registro de sucesos del agente.

Sólo se devuelve un resultado global para los métodos mapMQUserId de todas las salidas, como se indica a continuación:

- Si el método devuelve un valor de USER_SUCCESSFULLY_MAPPED o USER_DENIED_ACCESS como el código de resultado, este valor es el resultado devuelto y no se llama a los métodos mapMQUserId de las salidas subsiguientes.
- Si el método devuelve un valor NO_MAPPING_FOUND como el código de resultado, se llama al método mqMQUserId de la salida siguiente.
- Si no existe ninguna salida subsiguiente, se devuelve el resultado NO_MAPPING_FOUND.
- El agente de puente considera un código de resultado global de USER_DENIED_ACCESS o NO_MAPPING_FOUND como una anomalía en la transferencia.

Para ejecutar la salida, realice los pasos siguientes:

Procedimiento

1. Compile la salida de usuario de credenciales de puente de protocolo.
2. Cree un archivo de archivado Java (JAR) que contenga la salida compilada y su estructura de paquetes.
3. Coloque el archivo JAR que contiene la clase de salida en el directorio `exits` del agente de puente. El directorio está en el directorio *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*bridge_agent_name*.

4. Edite el archivo de propiedades del agente de puente de protocolo para incluir la propiedad `protocolBridgeCredentialExitClasses`. Para el valor de esta propiedad, especifique una lista de clases, separadas por comas, que implementan una rutina de salida de credenciales de puente de protocolo. Las clases de salida se invocan en el orden en que están especificadas en esta lista. Para obtener más información, consulte El archivo MFT `agent.properties`.
5. Edite el archivo de propiedades del agente de puente de protocolo para incluir:

```
exitClassPath=IBM MQ
installation_directory\mqft\config\configuration_queue_manager\agents\protocol_bridge_agent_n
ame\exits\SampleCredentialExit.jar
```

   El archivo `agent.properties` para un agente se encuentra en el directorio *MQ_DATA_PATH*/mqft/config/*coordination_qmgr_name*/agents/*bridge_agent_name*.

   Si cambia el archivo `agent.properties`, debe reiniciar el agente para recoger los cambios.
6. Puede especificar opcionalmente la propiedad `protocolBridgeCredentialConfiguration`. El valor que especifique para esta propiedad se pasa como un objeto String al método initialize() de las clases de salida especificadas mediante `protocolBridgeCredentialExitClasses`. Para obtener más información, consulte El archivo MFT `agent.properties`.
7. Inicie el agente de puente de protocolo con el mandato **fteStartAgent**.

Utilización de la salida de usuario de credenciales de puente de protocolo de ejemplo
Managed File Transfer proporciona una salida de usuario de ejemplo que realiza la correlación de credenciales de usuario.

Acerca de esta tarea

Se proporciona una salida de credencial de puente de protocolo de ejemplo en el directorio *MQ_INSTALLATION_PATH*/mqft/samples/protocolBridge y en el tema Ejemplo de salida de usuario de credenciales de puente de protocolo. Este ejemplo se basa en la interfaz `com.ibm.wmqfte.exitroutine.api.ProtocolBridgeCredentialExit`.

La salida `SampleCredentialExit.java` lee un archivo de propiedades que correlaciona los ID de usuario MQMD asociados con las solicitudes de transferencia a los ID de usuario del servidor y a las contraseñas del servidor. La ubicación del archivo de propiedades se extrae de la propiedad de agente de puente de protocolo `protocolBridgeCredentialConfiguration`.

Para ejecutar la salida de usuario de ejemplo, realice los pasos siguientes:

Procedimiento

1. Compile el archivo `SampleCredentialExit.java`.
2. Cree un archivo JAR que contenga la salida compilada y su estructura de paquetes.
3. Coloque el archivo JAR en el directorio *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*bridge_agent_name*/exits.
4. Edite el archivo *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*bridge_agent_name*/`agent.properties` para que contenga la línea:

```
protocolBridgeCredentialExitClasses=SampleCredentialExit
```

5. Edite el archivo de propiedades del agente de puente de protocolo para incluir:

```
exitClassPath=IBM MQ
installation_directory\mqft\config\configuration_queue_manager\agents\protocol_bridge_agent_n
ame\exits\SampleCredentialExit.jar
```

   El archivo `agent.properties` para un agente se encuentra en el directorio *MQ_DATA_PATH*/mqft/config/*coordination_qmgr_name*/agents/*agent_name*.

Si cambia el archivo `agent.properties`, debe reiniciar el agente para recoger los cambios.

6. Cree un archivo de propiedades de credenciales (`credentials.properties`) en el directorio *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*bridge_agent* y edítelo para incluir entradas en el formato:

```
mqUserId=serverUserId,serverPassword
```

7. Edite el archivo *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*bridge_agent_name*`/agent.properties` para que contenga la línea:

```
protocolBridgeCredentialConfiguration=MQ_DATA_PATH/mqft/
config/coordination_queue_manager/agents/bridge_agent_name/credentials.properties
```

   Debe utilizar la vía de acceso absoluta al archivo `credentials.properties`.

8. Inicie el agente de puente de protocolo utilizando el mandato **fteStartAgent**.

Conceptos relacionados

"El puente de protocolo" en la página 296
El puente de protocolo permite que la red de Managed File Transfer (MFT) acceda a los archivos almacenados en un servidor de archivos fuera de la red de MFT, bien en su dominio local o bien en una ubicación remota. Este servidor de archivos puede utilizar los protocolos de red FTP, FTPS o SFTP. Cada servidor de archivos necesita al menos un agente dedicado. El agente dedicado se conoce como el agente de puente de protocolo. Un agente de puente puede interactuar con varios servidores de archivos.

Referencia relacionada

Interfaz ProtocolBridgeCredentialExit.java

Interfaz ProtocolBridgeCredentialExit2.java

Ejemplo de salida de usuario de credenciales de puente de protocolo

El archivo MFT `agent.properties`

fteCreateBridgeAgent (crear y configurar un agente de puente de protocolo de MFT)

Ejemplo: Cómo configurar un agente de puente de protocolo para utilizar credenciales de clave privada con un servidor SFTP UNIX

Este ejemplo muestra cómo puede generar y configurar el archivo `ProtocolBridgeCredentials.xml`. Este ejemplo es un ejemplo típico y los detalles pueden variar en función de la plataforma, pero los principios siguen siendo los mismos.

Acerca de esta tarea

Procedimiento

1. Genere una clave pública y privada que se utilizará para autenticarse con el servidor SFTP.

   Por ejemplo, en un sistema host Linux , puede utilizar la herramienta **ssh-keygen**, proporcionada como parte del paquete 'openssh', para crear el par de claves pública/privada.

   De forma predeterminada, sin argumentos, el mandato **ssh-keygen** solicita una ubicación y frase de contraseña para los dos archivos de claves, que toma como valor predeterminado los nombres:

```
id_rsa      <-- Private key
id_rsa.pub  <-- Public key
```

Atención: Si está utilizando el mandato **ssh-keygen** de una versión reciente de OpenSSH, como por ejemplo la que se suministra con RHEL 8, el formato de clave utilizado no es

compatible con el agente de puente de protocolo y los intentos de transferencia al servidor SFTP fallan con el mensaje:

```
BFGBR0216E: Authentication to protocol server 'sftp.host.address' failed
because of invalid private key.
```

Para crear una clave privada compatible con estas versiones más recientes de OpenSSH, especifique el formato de clave con el siguiente argumento en el mandato **ssh-keygen**:

```
ssh-keygen -m PEM
```

El contenido de la clave privada `id_rsa` tiene las primeras y últimas líneas de:

```
-----BEGIN RSA PRIVATE KEY-----
... ... ...
-----END RSA PRIVATE KEY-----
```

que es compatible con el agente de puente de protocolo.

2. Copie todo el contenido del archivo `id_rsa.pub` en el archivo `~/.ssh/authorized_keys` del usuario SFTP en el servidor SFTP.

   Asegúrese de que los permisos de archivo de este archivo y del directorio `~/.ssh` se han establecido correctamente para que el servidor SFTP permita la autenticación de claves. Estos permisos suelen ser:

```
~/.ssh                    Mode 700
~/.ssh/authorized_keys    Mode 600
```

3. Managed File Transfer requiere una huella dactilar ssh de host generada mediante el algoritmo MD5. Ejecute uno de los mandatos siguientes para obtener la huella digital ssh del host del servidor SFTP.

   - Para Red Hat® Enterprise Linux versión 6.x y posteriores, y Linux Ubuntu 14.04, ejecute el mandato siguiente:

     ```
     ssh-keygen -l -f /etc/ssh/ssh_host_rsa_key.pub
     ```

   - A partir de Red Hat Enterprise Linux 7.x, Linux Ubuntu 16.04 y SuSE Linux 12.4, el mandato ssh-keygen genera, de forma predeterminada, la huella dactilar ssh utilizando el algoritmo SHA256. Para generar la huella digital ssh utilizando el algoritmo MD5, ejecute el mandato siguiente:

     ```
     ssh-keygen -l -E MD5 -f /etc/ssh/ssh_host_rsa_key.pub
     ```

     La salida del mandato será parecida a la del ejemplo siguiente:

     ```
     2048 MD5:64:39:f5:49:41:10:55:d2:0b:81:42:5c:87:62:9d:27 no comment (RSA)
     ```

     Extraiga la parte hexadecimal sólo de la salida que se utilizará como clave de host en el archivo `ProtocolBridgeCredentials.xml` (consulte el paso “4” en la página 311). Por lo tanto, en este ejemplo, se extraerá `64:39:f5:49:41:10:55:d2:0b:81:42:5c:87:62:9d:27`.

4. En el sistema de agente de puente de protocolo, edite el archivo `ProtocolBridgeCredentials.xml`. Sustituya los valores que aparecen en cursiva en el ejemplo siguiente por sus propios valores:

```
<tns:credentials xmlns:tns="http://wmqfte.ibm.com/ProtocolBridgeCredentials"
xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://wmqfte.ibm.com/ProtocolBridgeCredentials
 ProtocolBridgeCredentials.xsd ">

<tns:agent name="Agent_name">

<tns:server name="SFTP_name">

<tns:user name="mq_User_ID" serverUserId="SFTP_user_ID"
  hostKey="ssh_host_finger">
```

```
<tns:privateKey associationName="name" keyPassword="pass_phrase">
 Complete contents of the id_rsa file including the entries
 -----BEGIN RSA PRIVATE KEY-----

 -----END RSA PRIVATE KEY-----
</tns:privateKey>
</tns:user>

</tns:server>
</tns:agent>
</tns:credentials>
```

donde:

- *nombre_agente* es el nombre del agente de puente de protocolo.
- *SFTP_host_name* es el nombre del servidor SFTP tal como se muestra en el archivo `ProtocolBridgeProperties.xml`.
- *ID_usuario_mq* es el ID de usuario MQMD asociado a la solicitud de transferencia.
- *SFTP_user_ID* es el ID de usuario SFTP que se utiliza en el paso 2. Es el valor que se pasa al servicio SFTP como ID de usuario de inicio de sesión.
- *huella_host_ssh* es la huella digital que se ha obtenido en el paso 3.
- *nombre* es un nombre que puede especificar para utilizarlo con fines de rastreo y registro cronológico.
- *frase_contraseña* es la frase de contraseña que ha proporcionado en el mandato ssh-keygen en el paso 1.
- *Contenido completo del archivo id_rsa* es el contenido completo del archivo `id_rsa` generado del paso 1. Para evitar un error de conexión, asegúrese de incluir las dos entradas siguientes:

  ```
  -----BEGIN RSA PRIVATE KEY-----

  -----END RSA PRIVATE KEY-----
  ```

  Puede añadir claves adicionales duplicando el elemento <tns:privatekey>.

5. Inicie el agente de puente de protocolo si aún no está iniciado. De forma alternativa, el agente de puente de protocolo sondea periódicamente el archivo `ProtocolBridgeCredentials.xml` y recoge los cambios.

Referencia relacionada

Formato del archivo de credenciales de puente de protocolo

fteCreateBridgeAgent (crear y configurar un agente de puente de protocolo de MFT)

El archivo MFT `agent.properties`

“Correlación de credenciales para un servidor de archivos” en la página 305

Correlacione credenciales de usuario de Managed File Transfer con credenciales de usuario del servidor de archivos, utilizando la función de correlación de credenciales predeterminada del agente de puente de protocolo o escribiendo su propia salida de usuario. Managed File Transfer proporciona una salida de usuario de ejemplo que realiza la correlación de credenciales de usuario.

Configurar un puente de protocolo para un servidor FTPS

Configure un servidor FTPS de forma similar a como configura un servidor FTP: cree un agente de puente para el servidor, defina las propiedades del servidor y correlacione credenciales de usuario.

Acerca de esta tarea

Para configurar un servidor FTPS, realice los pasos siguientes:

Procedimiento

1. Cree un agente de puente de protocolo para el servidor FTPS utilizando el mandato **fteCreateBridgeAgent**. Los parámetros que son aplicables a FTP también son aplicables a FTPS pero además hay tres parámetros obligatorios específicos de FTPS:
   a) El parámetro **-bt** . Especifique FTPS como el valor de este parámetro.
   b) El parámetro **-bts** para el archivo de almacén de confianza. El mandato presupone que solamente es necesaria la autenticación de servidor y debe especificar la ubicación del archivo de almacén de confianza.

   La forma explícita del protocolo FTPS se configura con el mandato **fteCreateBridgeAgent** de forma predeterminada, pero puede configurar la forma implícita cambiando el archivo de propiedades de puente de protocolo. El puente de protocolo siempre se conecta a los servidores FTPS en modalidad pasiva.

   Si desea más información sobre el mandato **fteCreateBridgeAgent**, consulte fteCreateBridgeAgent (crear y configurar un agente de puente de protocolo MFT).

   Si necesita instrucciones sobre cómo crear archivos de almacén de confianza, consulte la información sobre la herramienta de claves en la documentación de la herramienta de claves deOracle.
2. Defina las propiedades del servidor FTPS dentro de un elemento <ftpsServer> en el archivo de propiedades del puente de protocolo: `ProtocolBridgeProperties.xml`. Para obtener más información, consulte “Definición de propiedades para servidores de archivos de protocolo utilizando el archivo ProtocolBridgeProperties.xml” en la página 298. También puede habilitar la autenticación de cliente editando el archivo de propiedades de puente de protocolo. Si desea detalles de todas las opciones de configuración, consulte Formato de archivo de propiedades del puente de protocolo.
3. Correlacione credenciales de usuario de Managed File Transfer con credenciales de usuario del servidor FTPS, utilizando la función de correlación de credenciales predeterminada del agente de puente de protocolo o escribiendo su propia salida de usuario. Para obtener más información, consulte “Correlación de credenciales para un servidor de archivos” en la página 305.
4. De forma predeterminada, el archivo de almacén de confianza está configurado con el formato JKS; si desea cambiar el formato, edite el archivo de propiedades de puente de protocolo.

Ejemplo

A continuación se muestra una entrada de ejemplo para un servidor FTPS en el archivo de propiedades de puente de protocolo:

```
<tns:serverProperties xmlns:tns="http://wmqfte.ibm.com/ProtocolBridgeProperties"
       xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://wmqfte.ibm.com/ProtocolBridgeProperties
       ProtocolBridgeProperties.xsd">
    <tns:defaultServer name="ftpsserver.mycompany.com" />

    <tns:ftpsServer name="ftpsserver.mycompany.com" host="ftpsserver.mycompany.com" port="990"
platform="windows"
       timeZone="Europe/London" locale="en_US" fileEncoding="UTF8"
       listFormat="unix" limitedWrite="false"
       trustStore="c:\mydirec\truststore.jks" />

    <!-- Define servers here -->
</tns:serverProperties>
```

Qué hacer a continuación

Si desea información sobre las partes del protocolo FTPS que están soportadas y que no están soportadas, consulte Soporte del servidor FTPS por el puente de protocolo.

Conceptos relacionados

“El puente de protocolo” en la página 296

El puente de protocolo permite que la red de Managed File Transfer (MFT) acceda a los archivos almacenados en un servidor de archivos fuera de la red de MFT, bien en su dominio local o bien en una ubicación remota. Este servidor de archivos puede utilizar los protocolos de red FTP, FTPS o SFTP. Cada servidor de archivos necesita al menos un agente dedicado. El agente dedicado se conoce como el agente de puente de protocolo. Un agente de puente puede interactuar con varios servidores de archivos.

Tareas relacionadas

"Correlación de credenciales para un servidor de archivos utilizando el archivo ProtocolBridgeCredentials.xml" en la página 306
Correlacione credenciales de usuario de Managed File Transfer con credenciales de usuario del servidor de archivos, utilizando la función de correlación de credenciales predeterminada del agente de puente de protocolo. Managed File Transfer proporciona un archivo XML que puede editar para incluir información sobre las credenciales.

"Definición de propiedades para servidores de archivos de protocolo utilizando el archivo ProtocolBridgeProperties.xml" en la página 298
Defina las propiedades de uno o más servidores de archivos de protocolo a los que desea transferir archivos y mediante el archivo `ProtocolBridgeProperties.xml`, que Managed File Transfer proporciona en el directorio de configuración del agente.

Referencia relacionada

fteCreateBridgeAgent (crear y configurar un agente de puente de protocolo de MFT)

Formato del archivo de credenciales de puente de protocolo

Formato del archivo de propiedades de puente de protocolo

Soporte de servidor FTPS por el puente de protocolo

Escenarios y ejemplo para limitar el número de transferencias de archivos a servidores de archivos individuales

Cómo funciona el agente de puente de protocolo revisado con los atributos **maxActiveDestinationTransfers** y **failTransferWhenCapacityReached**, junto con algunos ejemplos.

Escenarios que muestran el funcionamiento del agente de puente de protocolo basándose en el valor `maxActiveDestinationTransfers`

Escenario 1

El archivo `ProtocolBridgeProperties.xml` para un agente de puente de protocolo contiene dos definiciones de servidor de archivos:

- No ha establecido el atributo global **maxActiveDestinationTransfers** .
- No ha establecido el atributo **maxActiveDestinationTransfers** en `fileServerA` y `FileServerB`.
- Ha establecido el atributo **maxDestinationTransfers** del agente de puente de protocolo en el valor predeterminado.

Si ha establecido el atributo **maxDestinationTransfers** del agente de puente de protocolo en el valor predeterminado de 25, entonces:

- El agente de destino empieza a procesar dos transferencias gestionadas a `fileServerA`.
- Ambas transferencias se completan.

  En este momento, el cliente se da cuenta de que `fileServerA` ha fallado y establece los valores siguientes para `fileServerA` en el archivo `ProtocolBridgeProperties.xml` :

  maxActiveDestinationTransfers = 0
  failTransferWhenCapacityReached =true
- Llega otra transferencia para `fileServerA` y algunas para `fileServerB`:

Basándose en las propiedades establecidas en el paso anterior, la transferencia gestionada para `fileServerA` se rechaza y se marca como fallida, mientras que las transferencias para `fileServerB` se maneja en el flujo estándar existente.

- Pasado algún tiempo, el cliente averigua que `fileServerA` vuelve a estar en ejecución, de manera que el cliente elimina o comenta el valor añadido anteriormente en el archivo `ProtocolBridgeProperties.xml`. Cuando llega una nueva transferencia gestionada para A, `fileServerA`, se maneja en el flujo estándar existente.

Escenario 2

- Ha establecido el atributo **maxActiveDestinationTransfers** de un servidor de archivos y no ha establecido el atributo **failTransferWhenCapacityReached**.
- El agente de puente de protocolo actúa como el agente de destino de este número de transferencias gestionadas para el servidor de archivos.
- El valor del atributo **maxActiveDestinationTransfers** se reduce en 1.

El agente de puente de protocolo actualiza dinámicamente su configuración y establece **maxActiveDestinationTransfers** en el nuevo valor mientras está todavía activo. Las transferencias gestionadas en curso no se ven afectadas por esta actualización y pueden completarse.

Escenario 3

El archivo ProtocolBridgeProperties.xml para un agente de puente de protocolo contiene dos definiciones de servidor de archivos:

- No ha establecido el atributo global **maxActiveDestinationTransfers** .
- No ha establecido el atributo **failTransferWhenCapacityReached** .
- Ha establecido **maxActiveDestinationTransfers** en 1 en `fileServerA`.
- No ha establecido el atributo **maxActiveDestinationTransfers** en `fileServerB`.

Si el agente de puente de protocolo tiene el atributo **maxDestinationTransfers** establecido en 5:

- El número máximo de transferencias de destino desde el agente de puente de protocolo a `fileServerA` es 1 (aunque el agente de destino tenga 5 ranuras de transferencia de destino, solo 1 puede utilizarse para transferencias gestionadas a `fileServerA`).

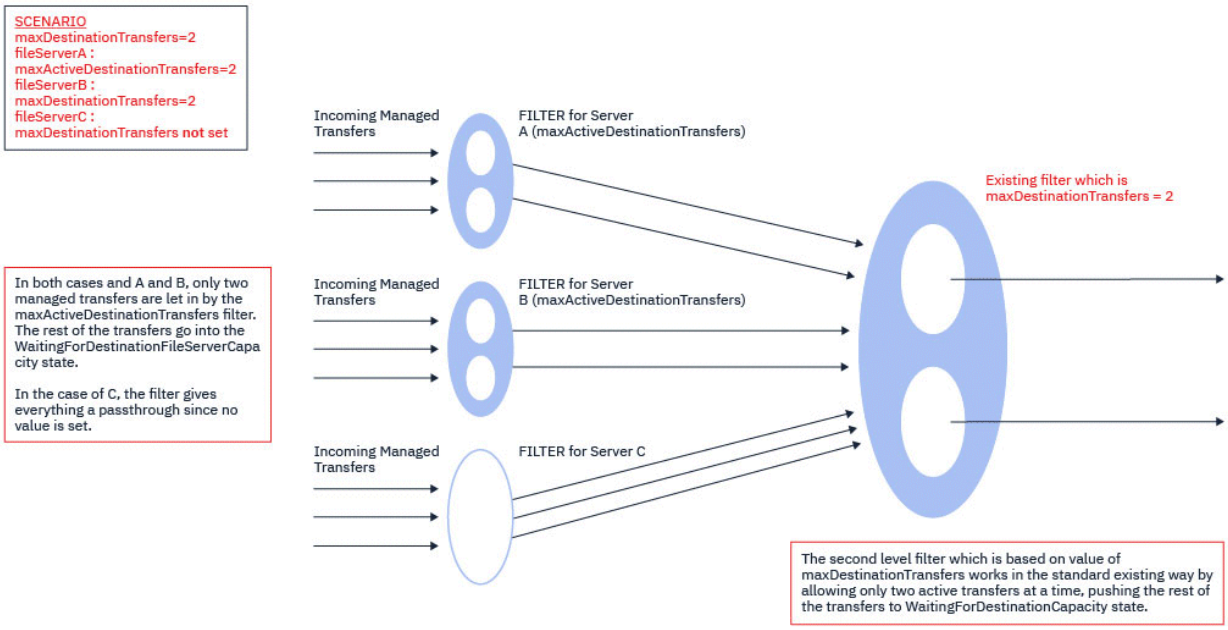
  Esto resulta útil cuando `fileServerA` falla. Una vez que `fileServerA` se está ejecutando de nuevo, el valor de **maxActiveDestinationTransfers** se puede aumentar a 5 para permitir la capacidad total de las transferencias de destino permitidas.
- El número máximo de transferencias de destino desde el agente de puente de protocolo a `fileServerB` es 5.

  Puesto que **maxActiveDestinationTransfers** no se ha establecido para este servidor, el agente de puente de protocolo puede utilizar las 5 ranuras de transferencia de destino para realizar transferencias gestionadas al mismo.

Escenario 4

En el diagrama siguiente:

- Ha establecido el atributo **maxDestinationTransfers** en 2 en el archivo `agent.properties` .
- Ha establecido **maxActiveDestinationTransfers** en 2 en `fileServerA`.
- Ha establecido el atributo **maxActiveDestinationTransfers** en 2 en `fileServerB`.
- No ha establecido el atributo **maxActiveDestinationTransfers** en `fileServerC`.



Como muestra el diagrama, los atributos **maxActiveDestinationTransfers** y **maxDestinationTransfers** son independientes entre sí.

Los valores de **maxActiveDestinationTransfers** para cada uno de los servidores se comprueban. En función de este valor, se permite que las trasferencias continúen o se pasan al estado **WaitingForDestinationFileServerCapacity**.

Las transferencias a las que se permite continuar aplican el flujo de comprobación estándar existente para **maxDestinationTransfers**.

Escenario 5

Atención: Debe actuar con precaución al establecer los valores de los atributos **maxActiveDestinationTransfers**, ya que debe tener siempre presente el valor del atributo **maxDestinationTransfers**.

Si no lo hace, se puede producir una situación tal como se describe en el texto siguiente:

- No ha establecido un valor para el atributo **maxActiveDestinationTransfers** global.
- Ha establecido un valor de **maxDestinationTransfers**= 2 en el archivo `agent.properties` .
- Ha establecido un valor de **maxActiveDestinationTransfers**= 2 en `fileServerA`.
- No ha establecido un valor para **maxActiveDestinationTransfers** en `fileServerB`.

Suponga que tiene lugar la siguiente secuencia de sucesos:

- El agente de puente de protocolo recibe una solicitud para transferir un archivo a `fileServerA`. El agente de puente de protocolo no está haciendo nada en este momento, por lo que acepta esta solicitud de transferencia gestionada.

  Las ranuras de transferencia estarían en esta situación:

  - Transferencias de destino: 1
  - Transferencias de destino para `fileServerA`: 1
  - Transferencias de destino para `fileServerB`: 0
- Ahora, el agente de puente de protocolo recibe otra solicitud para actuar como agente de destino de una transferencia gestionada en la que interviene `fileServerA`. De nuevo, acepta la solicitud, por lo que la situación de las ranuras de transferencia sería la siguiente:

  - Transferencias de destino: 2
  - Transferencias de destino para `fileServerA`: 2
  - Transferencias de destino para `fileServerB`: 0

Las dos ranuras `Destination Transfer` del agente están ahora ocupadas y, por lo tanto, el agente no puede participar en más transferencias gestionadas hasta que haya finalizado una de las transferencias a `fileServerA` .

- Poco después, `fileServerA` falla, lo cual provoca que las dos transferencias gestionadas entren en recuperación. Las ranuras de `Destination transfer` que estas transferencias gestionadas están utilizando permanecen en uso durante este tiempo.
- A continuación, el agente de puente de protocolo recibe una solicitud para transferir un archivo a `fileServerB`. Hay un espacio para esta transferencia en las ranuras de `Destination Transfers for fileServerB` , sin embargo, se están utilizando todas las ranuras de `Destination Transfer` para el agente y, por lo tanto, la transferencia se coloca en el registro de reserva para que se pueda volver a intentar más tarde.

Como resultado, la transferencia a `fileServerB` se bloquea hasta que al menos una de las transferencias a `fileServerA` se ha completado y ha liberado su ranura `Destination Transfer` .

Para impedir que esto suceda:

- Establezca el valor de **`maxActiveDestinationTransfers`** en los servidores de archivos para que sea menor que el valor de **`maxDestinationTransfers`** , para que permanezcan las ranuras libres.
- O distribuya uniformemente el valor del atributo **`maxActiveDestinationTransfers`** entre todos los servidores de punto final.

Comportamiento del agente de puente de protocolo basado en los valores del atributo `maxActiveDestinationTransfers`

Nota: En todo los casos listados en la tabla siguiente, si el atributo **`maxActiveDestinationTransfers`** se ha establecido en un valor que no es válido, el agente de puente de protocolo presupone que este atributo no se ha establecido.

maxActiveDestinationTransfers	**Valor de ejemplo**	**Descripción**
No especificado	No especificado	Las transferencias funcionan de forma normal. No se hay ningún límite en cuanto al número de transferencias para el punto final *ftp*.
Especificado	0	No se permiten transferencias con este punto final *ftp* específico.
Valor negativo	-1	Se registra un error en output0.log. El valor `-1` no es válido para un entero no negativo. El agente de puente de protocolo presupone que el atributo no se ha establecido.
Valor no entero	abc	Se registra un error en output0.log. El valor `abc` no es válido para un entero. El agente de puente de protocolo presupone que el atributo no se ha establecido.
Vacío	""	El valor `' '` del atributo maxActiveDestinationTransfers no es válido para un entero no negativo.

maxActiveDestinationTransfers	Valor de ejemplo	Descripción
Especificado	5	Solo permite la ejecución de cinco transferencias activas en cualquier momento para este punto final *ftp*. Las transferencias excesivas se reintentan o se rechazan, en función del valor del atributo failTransferWhenCapacityReached.

Comportamiento del agente de puente de protocolo para la combinación de los atributos `maxActiveDestinationTransfers` y `failTransferWhenCapacityReached`

Valor de failTransferWhenCapacityReached	Valor de maxActiveDestinationTransfers	Resultado
False	3	Se permiten tres transferencias activas para este servidor de punto final. Si hay más transferencias, se reintentan.
Sí	3	Se permiten tres transferencias activas para este servidor de punto final. Si hay más transferencias, se rechazan y se marcan como fallidas.
No especificado	3	Se toma el valor predeterminado **false** para failTransferWhenCapacityReached. El resultado es que se permiten tres transferencias activas para este servidor de punto final. Si hay más transferencias, se reintentan.
Otros valores distintos del valor booleano	Especificado	El error se anota en output.log. El valor especificado para failTransferWhenCapacityReached no es un valor booleano. Se toma el valor predeterminado para failTransferWhenCapacityReached.

Comportamiento del agente de puente de protocolo para la combinación de los atributos `maxDestinationTransfers` y `failTransferWhenCapacityReached`

Valor de failTransferWhenCapacityReached	Valor de maxDestinationTransfers	Resultado
Sí	10	Cuando el número de transferencias activas simultáneas alcanza 10, el agente de puente de protocolo falla en la 11[a] transferencia gestionada.
False	10	Comportamiento existente Cuando el número de transferencias activas simultáneas alcanza 10, la 11[a] transferencia gestionada se pone en cola a la espera de que se libere una ranura.
No especificado	10	Comportamiento existente

Mensajes de error

Mensaje existente:

BFGS0082I
Se anota en el archivo output0.log del agente de origen cuando el agente de puente de protocolo rechaza la transferencia cuando el agente de puente de protocolo ya está ejecutando el número máximo de transferencias definido en el atributo **`maxDestinationTransfers`**.

Mensajes nuevos:

BFGSS0085I
Se anota en el archivo output0.log file del agente de origen cuando el agente de puente de protocolo rechaza y reintenta una transferencia gestionada

BFGSS0086I
Se anota en el archivo output0.log file del agente de origen cuando el agente de puente de protocolo rechaza y reintenta una transferencia gestionada, y el elemento de destino no incluye el nombre del servidor de archivos

BFGSS0084E
Se anota en el Explorador y el archivo audit.xml cuando el agente de puente de protocolo rechaza la operación porque sobrepasa el número de transferencia simultáneas especificado en el atributo **`maxActiveDestinationTransfers`** y marca la transferencia gestionada como fallida.

BFGSS0087E
Se anota en el Explorador y el archivo audit.xml cuando el agente de puente de protocolo rechaza la operación porque sobrepasa el número de transferencia de destino especificado en el atributo **`maxActiveDestinationTransfers`** y marca la transferencia gestionada como fallida.

BFGSS0088W
Se anota en el archivo output0.log cuando el valor del atributo **`maxActiveDestinationTransfers`** sobrepasa el valor del atributo **`maxDestinationTransfers`**.

BFGSS0089I
Se anota en el archivo output0.log del agente de puente de protocolo de destino cuando está trabajando en un agente de origen que no es de IBM MQ 9.3.0 ni posterior.

Conceptos relacionados

"El puente de protocolo" en la página 296

El puente de protocolo permite que la red de Managed File Transfer (MFT) acceda a los archivos almacenados en un servidor de archivos fuera de la red de MFT, bien en su dominio local o bien en una ubicación remota. Este servidor de archivos puede utilizar los protocolos de red FTP, FTPS o SFTP. Cada servidor de archivos necesita al menos un agente dedicado. El agente dedicado se conoce como el agente de puente de protocolo. Un agente de puente puede interactuar con varios servidores de archivos.

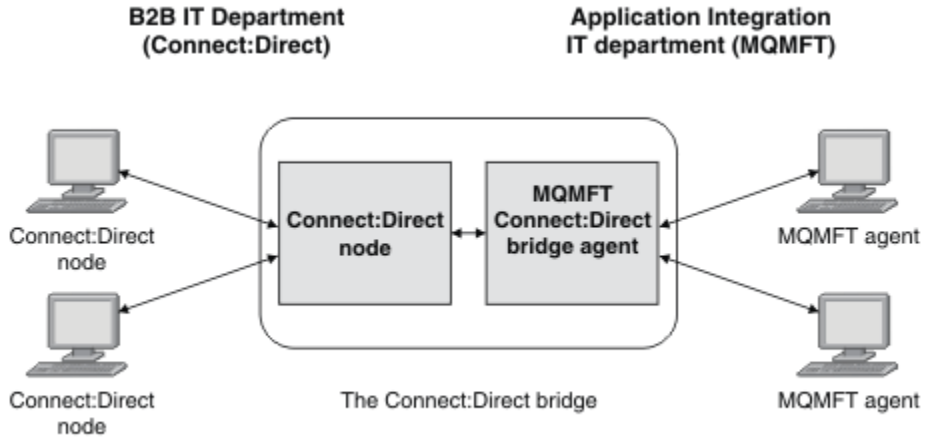
Tareas relacionadas

"Definición de propiedades para servidores de archivos de protocolo utilizando el archivo ProtocolBridgeProperties.xml" en la página 298

Defina las propiedades de uno o más servidores de archivos de protocolo a los que desea transferir archivos y mediante el archivo `ProtocolBridgeProperties.xml`, que Managed File Transfer proporciona en el directorio de configuración del agente.

El puente Connect:Direct

Puede transferir archivos a y desde una red de IBM Sterling Connect:Direct existente. Utilice el puente Connect:Direct, que es un componente de Managed File Transfer, para transferir archivos entre MFT y IBM Sterling Connect:Direct.



El diagrama muestra un puente MFT Connect:Direct entre dos departamentos, el departamento de TI de B2B y el departamento de TI de Application Integration. El departamento de TI de B2B utiliza Connect:Direct para transferir archivos a y desde los business partners de la empresa. El departamento Application Integration IT utiliza IBM MQ como la infraestructura de mensajería y por ello ha elegido recientemente Managed File Transfer como su solución de transferencia de archivos.

Utilizando el puente MFT Connect:Direct , los dos departamentos pueden transferir archivos entre la red Connect:Direct en el departamento de TI B2B y la red MFT en el departamento de TI de integración de aplicaciones. El puente Connect:Direct es un componente de Managed File Transfer, que incluye un agente de MFT que se comunica con un nodo Connect:Direct. El agente MFT se dedica a transferencias con el agente nodo Connect:Direct y se conoce como agente de puente Connect:Direct.

El puente Connect:Direct está disponible como parte de los componentes Servicio y Agente de Managed File Transfer y se puede utilizar en las tareas siguientes:

1. Utilizar mandatos Managed File Transfer para iniciar una transferencia de un archivo o varios archivos, desde un agente de MFT a un nodo Connect:Direct.
2. Utilizar mandatos Managed File Transfer para iniciar una transferencia de un archivo o varios archivos, desde un nodo Connect:Direct a un agente de MFT.
3. Utilice los mandatos Managed File Transfer para iniciar una transferencia de archivos que inicie un proceso Connect:Direct definido por el usuario.
4. Utilizar un proceso Connect:Direct para someter una solicitud de transferencia de archivos de MFT.

Un puente Connect:Direct puede transferir archivos a o desde únicamente nodos Connect:Direct. El puente Connect:Direct puede transferir archivos a o desde su sistema de archivos local sólo como parte de una transferencia sometida por un proceso Connect:Direct.

z/OS Puede utilizar el puente Connect:Direct para transferir a o desde un conjunto de datos que se encuentre en un nodo Connect:Direct de un sistema z/OS. Existen algunas diferencias de comportamiento en comparación con las transferencias de conjuntos de datos en las que sólo participan agentes de Managed File Transfer. Para obtener más información, consulte z/OS Transferencia de conjuntos de datos a y desde nodos Connect:Direct.

Plataformas soportadas

El puente Connect:Direct está formado por un agente de puente MFT Connect:Direct y un nodo Connect:Direct . El agente recibe soporte en Windows y Linux para x86-64. El nodo recibe soporte en plataformas admitidas para IBM Sterling Connect:Direct para Windows y IBM Sterling Connect:Direct para UNIX. Si desea instrucciones sobre cómo crear el agente de puente Connect:Direct y configurar un nodo Connect:Direct para que el agente se comunique con él, consulte Configuración del puente Connect:Direct.

El puente Connect:Direct puede transferir archivos a y desde nodos Connect:Direct que se ejecuten como parte de una instalación del Servicio Connect:Direct para Windows o Connect:Direct para UNIX o Connect:Direct para z/OS. Para obtener información de las versiones de Connect:Direct admitidas, consulte la página web Requisitos del sistema para IBM MQ.

El agente y el nodo que forman el puente Connect:Direct deben estar en el mismo sistema, o tener acceso al mismo sistema de archivos, por ejemplo a través de un montaje NFS compartido. Este sistema de archivos se utiliza para almacenar temporalmente archivos durante las transferencias de archivos que implican el puente Connect:Direct, en un directorio definido por el parámetro **cdTmpDir**. El agente de puente Connect:Direct y el nodo de puente Connect:Direct deben poder acceder a este directorio utilizando el mismo nombre de vía de acceso. Por ejemplo, si el agente y el nodo están en sistemas Windows distintos, los sistemas deben utilizar la misma letra de unidad para montar el sistema de archivos compartidos. Las siguientes configuraciones permiten que el agente y el nodo utilicen el mismo nombre de vía de acceso:

- El agente y el nodo se hallan en el mismo sistema, que está ejecutando Windows o Linux para x86-64
- El agente está en Linux para x86-64 y el nodo está en AIX
- El agente está en un sistema Windows y el nodo está en otro sistema Windows

Las siguientes configuraciones no permiten que el agente y el nodo utilicen el mismo nombre de vía de acceso:

- El agente está en Linux para x86-64 y el nodo está en Windows
- El agente está en Windows y el nodo está en UNIX

Tenga en cuenta esta restricción al planificar la instalación del puente Connect:Direct.

Conceptos relacionados

“Recuperación y reinicio de transferencias a y desde nodos Connect:Direct” en la página 329
Managed File Transfer podría no ser capaz de conectarse al nodo IBM Sterling Connect:Direct durante una transferencia; por ejemplo, si el nodo deja de estar disponible. Managed File Transfer intenta recuperar la transferencia, o la transferencia falla y se genera un mensaje de error.

“Cómo someter un proceso Connect:Direct definido por el usuario desde una solicitud de transferencia de archivos” en la página 330
Puede someter una solicitud de transferencia para una transferencia que pasa a través del agente de puente Connect:Direct que llame a un proceso Connect:Direct definido por el usuario como parte de la transferencia de archivos.

“Utilización de procesos Connect:Direct para someter solicitudes de transferencia de Managed File Transfer” en la página 334

Puede someter una solicitud de transferencia al agente de puente Connect:Direct desde un proceso Connect:Direct. Managed File Transfer proporciona mandatos a los que se puede llamar desde una sentencia **RUN TASK** en un proceso Connect:Direct .

Tareas relacionadas

Configurar el puente Connect:Direct

"Transferir un archivo a un nodo Connect:Direct" en la página 322
Se puede transferir un archivo desde un agente de Managed File Transfer a un nodo Connect:Direct utilizando el puente Connect:Direct. Especifique un nodo Connect:Direct como destino de la transferencia especificando el agente de puente Connect:Direct como el agente de destino y especificando el archivo de destino en el formato *connect_direct_node_name*:*file_path*.

"Transferir un archivo desde un nodo Connect:Direct" en la página 323
Se puede transferir un archivo desde un nodo Connect:Direct a un Managed File Transfer Agent utilizando el puente Connect:Direct. Puede especificar un nodo Connect:Direct como origen de la transferencia especificando el agente de puente Connect:Direct como el agente de origen y especificando la especificación de origen en el formato*connect_direct_node_name*:*file_path*.

"Transferencia de varios archivos a un nodo Connect:Direct" en la página 325
Se pueden transferir varios archivos desde un Managed File Transfer Agent a un nodo Connect:Direct utilizando el puente Connect:Direct. Para utilizar un nodo Connect:Direct como destino de la transferencia de varios archivos, especifique el agente de puente Connect:Direct como agente de destino y especifique el directorio de destino en el formato *connect_direct_node_name*:*directory_path*.

"Transferring multiple files from a Connect:Direct node" en la página 326
You can transfer multiple files from a Connect:Direct node to a Managed File Transfer Agent by using the Connect:Direct bridge. You can specify a Connect:Direct node as the source of the multiple file transfer by specifying the Connect:Direct bridge agent as the source agent and specifying one or more source specifications in the form *connect_direct_node_name*:*file_path*.

"Transferencia de varios archivos a Connect:Direct utilizando comodines" en la página 327
Para transferir varios archivos desde un agente de Managed File Transfer a un nodo Connect:Direct, utilice el puente Connect:Direct. Puede utilizar caracteres comodín en la especificación de origen que suministra en el mandato **fteCreateTransfer**. Al igual que todas las transferencias de Managed File Transfer con comodines, sólo la última parte de la vía de acceso al archivo puede contener un carácter comodín. Por ejemplo, `/abc/def*` es una vía de acceso de archivo válida y `/abc*/def` no es válida.

Resolución de problemas con el puente Connect:Direct

Referencia relacionada

fteCreateCDAgent: crear un agente de puente Connect:Direct

Restricciones del agente de puente Connect:Direct

Transferir un archivo a un nodo Connect:Direct

Se puede transferir un archivo desde un agente de Managed File Transfer a un nodo Connect:Direct utilizando el puente Connect:Direct. Especifique un nodo Connect:Direct como destino de la transferencia especificando el agente de puente Connect:Direct como el agente de destino y especificando el archivo de destino en el formato *connect_direct_node_name*:*file_path*.

Antes de empezar

Para poder transferir un archivo, antes debe configurar el puente Connect:Direct, que es un componente de Managed File Transfer. Si desea más información, consulte Configuración del puente Connect:Direct.

Acerca de esta tarea

En este ejemplo, el agente de puente Connect:Direct se denomina CD_BRIDGE. El agente de origen se denomina FTE_AGENT y puede ser de cualquier versión de WMQFTE. El nodo Connect:Direct de destino se denomina CD_NODE1. El archivo que se va a transferir se encuentra en la vía de acceso del archivo `/home/helen/file.log` en el sistema donde se encuentra FTE_AGENT. El archivo se transfiere a la vía de acceso del archivo `/files/data.log` en el sistema donde se ejecuta CD_NODE1.

Procedimiento

1. Utilice el mandato fteCreateTransfer con el valor para el parámetro **-df** (archivo de destino) en el formato *connect_direct_node_name*:*file_path* y el valor del parámetro **-da** (agente de destino) especificado como el nombre del agente de puente Connect:Direct .

   Nota: El nodo Connect:Direct especificado por *connect_direct_node_name* es el nodo al que desea transferir el archivo, no el nodo Connect:Direct que funciona como parte del puente Connect:Direct..

   ```
   fteCreateTransfer -sa FTE_AGENT -da CD_BRIDGE
     -df CD_NODE1:/files/data.log /home/helen/file.log
   ```

   Si desea más información, consulte **fteCreateTransfer**: iniciar una nueva transferencia de archivos.
2. El agente de origen FTE_AGENT transfiere el archivo al agente de puente Connect:Direct CD_BRIDGE. El archivo se almacena temporalmente en el sistema en el que se ejecuta el agente de puente Connect:Direct, en la ubicación que define la propiedad de agente cdTmpDir. El agente de puente Connect:Direct transfiere el archivo al nodo Connect:Direct CD_NODE1.

Conceptos relacionados

"El puente Connect:Direct" en la página 320
Puede transferir archivos a y desde una red de IBM Sterling Connect:Direct existente. Utilice el puente Connect:Direct, que es un componente de Managed File Transfer, para transferir archivos entre MFT y IBM Sterling Connect:Direct.

Tareas relacionadas

"Transferir un archivo desde un nodo Connect:Direct" en la página 323
Se puede transferir un archivo desde un nodo Connect:Direct a un Managed File Transfer Agent utilizando el puente Connect:Direct. Puede especificar un nodo Connect:Direct como origen de la transferencia especificando el agente de puente Connect:Direct como el agente de origen y especificando la especificación de origen en el formato*connect_direct_node_name*:*file_path*.

Referencia relacionada

El archivo MFT `agent.properties`

Transferir un archivo desde un nodo Connect:Direct

Se puede transferir un archivo desde un nodo Connect:Direct a un Managed File Transfer Agent utilizando el puente Connect:Direct. Puede especificar un nodo Connect:Direct como origen de la transferencia especificando el agente de puente Connect:Direct como el agente de origen y especificando la especificación de origen en el formato*connect_direct_node_name*:*file_path*.

Antes de empezar

Para poder transferir un archivo, antes debe configurar el puente Connect:Direct, que es un componente de Managed File Transfer. Consulte Configuración del puente Connect:Direct.

Acerca de esta tarea

En este ejemplo, el agente de puente Connect:Direct se denomina CD_BRIDGE. El agente de destino se denomina FTE_AGENT y puede ser de cualquier versión de Managed File Transfer. El nodo Connect:Direct de origen se denomina CD_NODE1. El archivo que se va a transferir se encuentra en la vía de acceso del archivo `/home/brian/in.file` en el sistema donde se encuentra CD_NODE1. El archivo se transfiere a la vía de acceso del archivo `/files/out.file` en el sistema en el que se está ejecutando FTE_AGENT.

Procedimiento

Utilice el mandato **fteCreateTransfer** con el valor para la especificación de origen en el formato *connect_direct_node_name*:*file_path* y el valor del parámetro **-sa** especificado como el nombre del agente de puente Connect:Direct .

Nota: El nodo Connect:Direct especificado por *connect_direct_node_name* es el nodo al que desea transferir el archivo, no el nodo Connect:Direct que opera como parte del puente Connect:Direct. Por ejemplo:

```
fteCreateTransfer -sa CD_BRIDGE -da FTE_AGENT
                  -df /files/out.file CD_NODE1:/home/brian/in.file
```

Si desea más información, consulte **fteCreateTransfer**: iniciar una nueva transferencia de archivos.

Resultados

El agente de puente Connect:Direct CD_BRIDGE solicita el archivo del nodo Connect:Direct CD_NODE1. El nodo Connect:Direct envía el archivo al puente Connect:Direct. Mientras el archivo se transfiere desde el nodo Connect:Direct, el puente Connect:Direct almacena el archivo temporalmente en la ubicación definida por la propiedad de agente `cdTmpDir`. Cuando el archivo ha terminado de transferirse desde el nodo Connect:Direct al puente Connect:Direct, el puente Connect:Direct envía el archivo al agente de destino FTE_AGENT y luego suprime el archivo de la ubicación temporal.

Conceptos relacionados

"El puente Connect:Direct" en la página 320

Puede transferir archivos a y desde una red de IBM Sterling Connect:Direct existente. Utilice el puente Connect:Direct, que es un componente de Managed File Transfer, para transferir archivos entre MFT y IBM Sterling Connect:Direct.

Referencia relacionada

El archivo MFT `agent.properties`

z/OS Transferencia de un conjunto de datos a un nodo Connect:Direct en z/OS

Se puede transferir un conjunto de datos desde un agente de Managed File Transfer en z/OS a un nodo Connect:Direct en z/OS utilizando un puente Connect:Direct que se halle en un sistema Windows o Linux.

Antes de empezar

Para poder transferir un archivo, antes debe configurar el puente Connect:Direct, que es un componente de Managed File Transfer. Consulte Configuración del puente Connect:Direct.

Acerca de esta tarea

En este ejemplo, el parámetro **-df** se utiliza para especificar el destino de la transferencia. El uso del parámetro **-df** es válido cuando el agente de origen de la transferencia es de cualquier versión de Managed File Transfer. En su lugar, puede utilizar el parámetro **-ds** . El agente de origen se denomina FTE_ZOS1 y es un agente de Managed File Transfer. El agente de puente de Connect:Direct se denomina CD_BRIDGE y se halla en un sistema Linux. El nodo Connect:Direct de destino se denomina CD_ZOS2. El nodo Connect:Direct tanto del agente de origen como del agente de destino se hallan en los sistemas z/OS. El conjunto de datos que se va a transferir se encuentra en `//FTEUSER.SOURCE.LIB` en el sistema donde está instalado FTE_ZOS1. El conjunto de datos se transfiere al conjunto de datos `//CDUSER.DEST.LIB` en el sistema donde está instalado CD_ZOS2.

Procedimiento

1. Utilice el mandato fteCreateTransfer con el valor para el parámetro **-df** con el formato: *connect_direct_node_name*:*data_set_name*;*attributes* y el valor del parámetro **-da** (agente de destino) especificado como el nombre del agente de puente Connect:Direct .

   El nodo Connect:Direct que especifica *connect_direct_node_name* es el nodo al que desea transferir el conjunto de datos, no el nodo Connect:Direct que opera como parte del puente Connect:Direct.

El nombre de conjunto de datos especificado mediante *nombre_conjunto_datos* debe ser absoluto, no relativo. Connect:Direct no antepone el nombre del usuario como prefijo del nombre de conjunto de datos.

```
fteCreateTransfer -sa FTE_ZOS1 -sm QM_ZOS
                  -da CD_BRIDGE -dm QM_BRIDGE
                  -df CD_ZOS2://'CDUSER.DEST.LIB;BLKSIZE(8000);LRECL(80)'
                  //'FTEUSER.SOURCE.LIB'
```

Si desea más información, consulte **fteCreateTransfer**: iniciar una nueva transferencia de archivos.

2. El agente de origen FTE_ZOS1 transfiere los datos del conjunto de datos al agente de puente Connect:Direct CD_BRIDGE. Los datos se almacenan temporalmente en un archivo sin formato en el sistema donde se ejecuta el agente de puente Connect:Direct, en la ubicación definida por la propiedad de agente cdTmpDir. El agente de puente Connect:Direct transfiere los datos al nodo Connect:Direct CD_ZOS2. Cuando se completa la transferencia, el archivo sin formato se suprime del sistema en el que se ejecuta el agente de puente Connect:Direct.

Conceptos relacionados

"El puente Connect:Direct" en la página 320
Puede transferir archivos a y desde una red de IBM Sterling Connect:Direct existente. Utilice el puente Connect:Direct, que es un componente de Managed File Transfer, para transferir archivos entre MFT y IBM Sterling Connect:Direct.

Tareas relacionadas

Transferencia de conjuntos de datos a y desde nodos Connect:Direct

Referencia relacionada

Propiedades de BPXWDYN que no se deben utilizar con MFT

Transferencia de varios archivos a un nodo Connect:Direct

Se pueden transferir varios archivos desde un Managed File Transfer Agent a un nodo Connect:Direct utilizando el puente Connect:Direct. Para utilizar un nodo Connect:Direct como destino de la transferencia de varios archivos, especifique el agente de puente Connect:Direct como agente de destino y especifique el directorio de destino en el formato *connect_direct_node_name*:*directory_path*.

Antes de empezar

Para poder transferir archivos, antes debe configurar el puente Connect:Direct, que es un componente de Managed File Transfer. Consulte Configuración del puente Connect:Direct.

Acerca de esta tarea

En este ejemplo, el agente de origen se denomina FTE_AGENT. El agente de puente Connect:Direct se denomina CD_BRIDGE. El nodo Connect:Direct de destino se denomina CD_NODE1. Los archivos que se van a transferir son `/home/jack/data.log`, `/logs/log1.txt` y `/results/latest` en el sistema donde se encuentra FTE_AGENT. Los archivos se transfieren al directorio `/in/files` del sistema donde se ejecuta CD_NODE1.

Procedimiento

Utilice el mandato fteCreateTransfer con el valor del parámetro **-dd** (directorio de destino) en el formato *connect_direct_node_name*:*directory_path*. Especifique el valor del parámetro **-da** (agente de destino) como nombre del agente de puente Connect:Direct.

Nota: El nodo Connect:Direct que especifica *connect_direct_node_name* es el nodo al que desea que se transfieran los archivos, no el nodo Connect:Direct que funciona como parte del puente Connect:Direct..

```
fteCreateTransfer -sa FTE_AGENT -da CD_BRIDGE
                  -dd CD_NODE1:/in/files /home/jack/data.log
                  /logs/log1.txt /results/latest
```

Si desea más información, consulte **fteCreateTransfer**: iniciar una nueva transferencia de archivos.

Resultados

El agente de origen FTE_AGENT transfiere el primer archivo al agente de puente Connect:Direct CD_BRIDGE. El agente de puente Connect:Direct almacena temporalmente el archivo en la ubicación definida por la propiedad cdTmpDir. Cuando el archivo se ha transferido por completo desde el agente de origen al puente Connect:Direct, el agente de puente Connect:Direct envía el archivo al nodo Connect:Direct que define la propiedad de agente cdNode. Este nodo envía el archivo al nodo Connect:Direct de destino CD_NODE1. El agente de puente Connect:Direct suprime el archivo de la ubicación temporal cuando finaliza la transferencia entre los nodos Connect:Direct. Este proceso se repite para cada archivo de origen especificado.

Conceptos relacionados

"El puente Connect:Direct" en la página 320
Puede transferir archivos a y desde una red de IBM Sterling Connect:Direct existente. Utilice el puente Connect:Direct, que es un componente de Managed File Transfer, para transferir archivos entre MFT y IBM Sterling Connect:Direct.

Tareas relacionadas

"Transferir un archivo a un nodo Connect:Direct" en la página 322
Se puede transferir un archivo desde un agente de Managed File Transfer a un nodo Connect:Direct utilizando el puente Connect:Direct. Especifique un nodo Connect:Direct como destino de la transferencia especificando el agente de puente Connect:Direct como el agente de destino y especificando el archivo de destino en el formato *connect_direct_node_name*:*file_path*.

"Transferencia de varios archivos a Connect:Direct utilizando comodines" en la página 327
Para transferir varios archivos desde un agente de Managed File Transfer a un nodo Connect:Direct, utilice el puente Connect:Direct. Puede utilizar caracteres comodín en la especificación de origen que suministra en el mandato **fteCreateTransfer**. Al igual que todas las transferencias de Managed File Transfer con comodines, sólo la última parte de la vía de acceso al archivo puede contener un carácter comodín. Por ejemplo, `/abc/def*` es una vía de acceso de archivo válida y `/abc*/def` no es válida.

"Transferir un archivo desde un nodo Connect:Direct" en la página 323
Se puede transferir un archivo desde un nodo Connect:Direct a un Managed File Transfer Agent utilizando el puente Connect:Direct. Puede especificar un nodo Connect:Direct como origen de la transferencia especificando el agente de puente Connect:Direct como el agente de origen y especificando la especificación de origen en el formato*connect_direct_node_name*:*file_path*.

"Transferring multiple files from a Connect:Direct node" en la página 326
You can transfer multiple files from a Connect:Direct node to a Managed File Transfer Agent by using the Connect:Direct bridge. You can specify a Connect:Direct node as the source of the multiple file transfer by specifying the Connect:Direct bridge agent as the source agent and specifying one or more source specifications in the form *connect_direct_node_name*:*file_path*.

Referencia relacionada

El archivo MFT `agent.properties`

z/OS Transferring multiple files from a Connect:Direct node

You can transfer multiple files from a Connect:Direct node to a Managed File Transfer Agent by using the Connect:Direct bridge. You can specify a Connect:Direct node as the source of the multiple file transfer by specifying the Connect:Direct bridge agent as the source agent and specifying one or more source specifications in the form *connect_direct_node_name*:*file_path*.

Before you begin

Before transferring a file, you must configure the Connect:Direct bridge, which is a component of Managed File Transfer. See Configuring the Connect:Direct bridge.

About this task

In this example, the Connect:Direct bridge agent is called CD_BRIDGE. The destination agent is called FTE_Z, and is running on a z/OS system. The source Connect:Direct node is called CD_NODE1. The files to be transferred are located at the file paths `/in/file1`, `/in/file2`, and `/in/file3` on the system where CD_NODE1 is located. The files are transferred to the partitioned data set `//OBJECT.LIB` on the system where FTE_Z is running.

Procedure

Use the fteCreateTransfer command with the values for the source specifications in the form *connect_direct_node_name*:*file_path* and the value of the **-sa** parameter specified as the name of the Connect:Direct bridge agent.

Note: The Connect:Direct node specified by *connect_direct_node_name* is the node that you want the files to be transferred from, not the Connect:Direct node that operates as part of the Connect:Direct bridge.

```
fteCreateTransfer -sa CD_BRIDGE -da FTE_Z
                  -dp //'OBJECT.LIB' CD_NODE1:/in/file1
                  CD_NODE1:/in/file2 CD_NODE1:/in/file3
```

For more information, see **fteCreateTransfer**: start a new file transfer.

Results

The Connect:Direct bridge agent CD_BRIDGE requests the first file from the Connect:Direct node CD_NODE1. The Connect:Direct node sends the file to the Connect:Direct bridge. While the file is being transferred from the Connect:Direct node, the Connect:Direct bridge stores the file temporarily in the location defined by the `cdTmpDir` agent property. When the file has finished transferring from the Connect:Direct node to the Connect:Direct bridge, the Connect:Direct bridge sends the file to the destination agent FTE_Z and then deletes the file from the temporary location. This process is repeated for each specified source file.

Related concepts

“El puente Connect:Direct” on page 320

Puede transferir archivos a y desde una red de IBM Sterling Connect:Direct existente. Utilice el puente Connect:Direct, que es un componente de Managed File Transfer, para transferir archivos entre MFT y IBM Sterling Connect:Direct.

Related reference

The MFT `agent.properties` file

Transferencia de varios archivos a Connect:Direct utilizando comodines

Para transferir varios archivos desde un agente de Managed File Transfer a un nodo Connect:Direct, utilice el puente Connect:Direct. Puede utilizar caracteres comodín en la especificación de origen que suministra en el mandato **fteCreateTransfer**. Al igual que todas las transferencias de Managed File Transfer con comodines, sólo la última parte de la vía de acceso al archivo puede contener un carácter comodín. Por ejemplo, `/abc/def*` es una vía de acceso de archivo válida y `/abc*/def` no es válida.

Antes de empezar

Para poder transferir un archivo, antes debe configurar el puente Connect:Direct, que es un componente de Managed File Transfer. Si desea más información, consulte Configuración del puente Connect:Direct.

Acerca de esta tarea

En este ejemplo, el agente de origen se denomina FTE_AGENT y el agente de puente Connect:Direct se denomina CD_BRIDGE. El nodo Connect:Direct de destino se denomina CD_NODE1. Los archivos que se van a transferir se encuentran en el directorio `/reports` del sistema donde se encuentra FTE_AGENT. Sólo se transfieren los archivos con nombres que empiezan por `report`, seguidos de dos caracteres y el sufijo `.log`. Por ejemplo, se transfiere el archivo `/reports/report01.log`, pero el archivo `/reports/report1.log` no se transfiere. Los archivos se transfieren al directorio `/home/fred` del sistema donde se ejecuta CD_NODE1.

Procedimiento

1. Utilice el mandato fteCreateTransfer con el valor del parámetro **-dd** (directorio de destino) en el formato *connect_direct_node_name*:*directory_path*. Para el parámetro **-da** (agente de destino), especifique el agente de puente Connect:Direct.

   Nota: El nodo Connect:Direct que especifica *connect_direct_node_name* es el nodo al que desea que se transfieran los archivos, no el nodo Connect:Direct que funciona como parte del puente Connect:Direct..

   ```
   fteCreateTransfer -sa FTE_AGENT -da CD_BRIDGE
                     -dd CD_NODE1:/home/fred "/reports/report??.log"
   ```

   Si desea más información, consulte **fteCreateTransfer**: iniciar una nueva transferencia de archivos.
2. El agente de origen FTE_AGENT transfiere el primer archivo que coincide con el patrón de `/reports/report??.log` con el agente de puente Connect:Direct CD_BRIDGE. El agente de puente Connect:Direct almacena temporalmente el archivo en la ubicación definida por la propiedad cdTmpDir. Cuando el archivo se ha transferido por completo desde el agente de origen al puente Connect:Direct, el agente de puente Connect:Direct envía el archivo al nodo Connect:Direct que define la propiedad de agente cdNode. Este nodo envía el archivo al nodo Connect:Direct de destino CD_NODE1. El agente de puente Connect:Direct suprime el archivo de la ubicación temporal cuando finaliza la transferencia entre los nodos Connect:Direct.Este proceso se repite para cada archivo fuente que coincida con el patrón de comodín `/reports/report??.log`.

   Nota: La lista de archivos que coinciden con el patrón de `/reports/report??.log` varía en función del sistema operativo del sistema en el que se encuentra el agente de origen FTE_AGENT.

   - Si el agente de origen se encuentra en un sistema con un sistema operativo Windows, la coincidencia del patrón no distingue entre mayúsculas y minúsculas. El patrón coincide con todos los archivos del directorio `/reports` con un nombre de archivo del formato `report` seguido de dos caracteres y un sufijo de `.log`, independientemente de las mayúsculas y minúsculas en el que se encuentren las letras. Por ejemplo, `Report99.Log` es una coincidencia.
   - Si el agente de origen se halla en un sistema con un sistema operativo Linux o UNIX, la coincidencia del patrón distingue entre mayúsculas y minúsculas. El patrón sólo coincide con los archivos del directorio `/reports` con un nombre de archivo del formato `report` seguido de dos caracteres y un sufijo de `.log`. Por ejemplo, `reportAB.log` es una coincidencia, pero `reportAB.LOG` y `Report99.Log` no son coincidencias.

Conceptos relacionados

"El puente Connect:Direct" en la página 320
Puede transferir archivos a y desde una red de IBM Sterling Connect:Direct existente. Utilice el puente Connect:Direct, que es un componente de Managed File Transfer, para transferir archivos entre MFT y IBM Sterling Connect:Direct.

Tareas relacionadas

Utilización de caracteres comodín con MFT

"Transferir un archivo a un nodo Connect:Direct" en la página 322
Se puede transferir un archivo desde un agente de Managed File Transfer a un nodo Connect:Direct utilizando el puente Connect:Direct. Especifique un nodo Connect:Direct como destino de la transferencia

especificando el agente de puente Connect:Direct como el agente de destino y especificando el archivo de destino en el formato *connect_direct_node_name*:*file_path*.

"Transferencia de varios archivos a un nodo Connect:Direct" en la página 325
Se pueden transferir varios archivos desde un Managed File Transfer Agent a un nodo Connect:Direct utilizando el puente Connect:Direct. Para utilizar un nodo Connect:Direct como destino de la transferencia de varios archivos, especifique el agente de puente Connect:Direct como agente de destino y especifique el directorio de destino en el formato *connect_direct_node_name*:*directory_path*.

"Transferring multiple files from a Connect:Direct node" en la página 326
You can transfer multiple files from a Connect:Direct node to a Managed File Transfer Agent by using the Connect:Direct bridge. You can specify a Connect:Direct node as the source of the multiple file transfer by specifying the Connect:Direct bridge agent as the source agent and specifying one or more source specifications in the form *connect_direct_node_name*:*file_path*.

Referencia relacionada

El archivo MFT `agent.properties`

Recuperación y reinicio de transferencias a y desde nodos Connect:Direct

Managed File Transfer podría no ser capaz de conectarse al nodo IBM Sterling Connect:Direct durante una transferencia; por ejemplo, si el nodo deja de estar disponible. Managed File Transfer intenta recuperar la transferencia, o la transferencia falla y se genera un mensaje de error.

Si el nodo Connect:Direct no está disponible

Si el nodo Connect:Direct deja de estar disponible; por ejemplo, debido a una interrupción de la alimentación o de la red, Managed File Transfer recupera una transferencia de archivos de las siguientes maneras:

- Si Managed File Transfer no se ha conectado correctamente anteriormente al nodo Connect:Direct como parte de esta solicitud de transferencia, la transferencia se vuelve a intentar durante un periodo de tiempo determinado por los valores de **cdMaxConnectionRetries** y **recoverableTransferRetryInterval properties**. Estas propiedades se especifican en el archivo `agent.properties` para el agente de puente Connect:Direct. La transferencia falla y se genera un mensaje de error, después de que el número de intentos fallidos alcance el valor de **cdMaxConnectionRetries property**. De forma predeterminada, se intenta la transferencia indefinidamente, con una espera de 60 segundos entre cada intento.
- Si Managed File Transfer ya se ha conectado correctamente al nodo Connect:Direct como parte de esta solicitud de transferencia, se intenta de nuevo la transferencia durante un tiempo determinado por los valores de las propiedades **cdMaxPartialWorkConnectionRetries** y **recoverableTransferRetryInterval**. La transferencia falla, y se genera un mensaje de error, después de que el número de intentos anómalos alcance el valor de la propiedad **cdMaxPartialWorkConnectionRetries**. De forma predeterminada, se intenta la transferencia indefinidamente, con una espera de 60 segundos entre cada intento.
- Para determinados tipos de anomalías de nodo Connect:Direct, por ejemplo, si el nodo se detiene de forma forzosa, los procesos Connect:Direct pasan a tener el estado `Held Due to Error (HE)` cuando se recupera el nodo. Una vez que se recupera el nodo, Managed File Transfer reanuda automáticamente cualquier proceso Connect:Direct que esté relacionado con la transferencia de archivos y tenga un estado de HE.
- Si la transferencia falla, los archivos temporales relacionados con la transferencia se suprimen del sistema que aloja el puente Connect:Direct. La ubicación de estos archivos temporales está definida por la propiedad **cdTmpDir**.
- Si la transferencia es de Managed File Transfer a Connect:Direct y se especifica una disposición de origen de supresión, los archivos de origen no se suprimen si la transferencia falla.

Si las credenciales de usuario del nodo Connect:Direct no son válidas

Si Managed File Transfer no puede conectarse al nodo Connect:Direct porque las credenciales del usuario son rechazadas por el nodo, la transferencia falla y se genera un mensaje de error. En esta situación, compruebe que ha proporcionado las credenciales de usuario correctas para el nodo Connect:Direct. Si desea más información, consulte Correlación de credenciales para Connect:Direct.

Si el agente de puente Connect:Direct no está disponible

Si el agente de puente Connect:Direct deja de estar disponible, las transferencias de archivos en curso se recuperan de la misma manera que las transferencias de Managed File Transfer estándar. Para obtener más información, consulte "Recuperación y reinicio de MFT" en la página 337.

Conceptos relacionados

"El puente Connect:Direct" en la página 320
Puede transferir archivos a y desde una red de IBM Sterling Connect:Direct existente. Utilice el puente Connect:Direct, que es un componente de Managed File Transfer, para transferir archivos entre MFT y IBM Sterling Connect:Direct.

"Recuperación y reinicio de MFT" en la página 337
Si el agente o el gestor de colas están no disponibles por algún motivo, por ejemplo, debido a un fallo en el suministro o en la red, Managed File Transfer se recupera tal como se indica a continuación en los siguientes escenarios:

Tareas relacionadas

Configurar el puente Connect:Direct

Referencia relacionada

El archivo MFT `agent.properties`

Cómo someter un proceso Connect:Direct definido por el usuario desde una solicitud de transferencia de archivos

Puede someter una solicitud de transferencia para una transferencia que pasa a través del agente de puente Connect:Direct que llame a un proceso Connect:Direct definido por el usuario como parte de la transferencia de archivos.

De forma predeterminada, cuando se somete una solicitud de transferencia de archivos para una transferencia que pasa a través del agente de puente Connect:Direct, el agente de puente Connect:Direct genera el proceso Connect:Direct que se utiliza para transferir el archivo a o desde el nodo Connect:Direct remoto.

Sin embargo, puede configurar el agente de puente Connect:Direct para que en su lugar llame a un proceso definido por el usuario Connect:Direct utilizando el archivo `ConnectDirectProcessDefinition.xml`.

El archivo `ConnectDirectProcessDefinition.xml`

El mandato **`fteCreateCDAgent`** crea el archivo `ConnectDirectProcessDefinitions.xml` en el directorio de configuración del agente *MQ_DATA_PATH*`/mqft/config/`*`coordination_queue_manager`*`/agents/`*`cd_bridge_agent_name`*. Antes de poder llamar a procesos Connect:Direct definidos por el usuario desde el agente de puente Connect:Direct, debe configurar definiciones de proceso editando este archivo.

El archivo define uno o más conjuntos de procesos que incluyen la ubicación de uno o más procesos Connect:Direct a los que se llama como parte de una transferencia. Cada conjunto de procesos incluye una serie de condiciones. Si la transferencia cumple todas las condiciones del conjunto de procesos, el conjunto de procesos se utiliza para especificar a qué procesos Connect:Direct llama la transferencia. Para obtener más información, consulte "Especificación del proceso Connect:Direct que se debe iniciar utilizando el archivo ConnectDirectProcessDefinition.xml" en la página 331.

Variables simbólicas intrínsecas

Puede utilizar las variables simbólicas intrínsecas definidas por Managed File Transfer para sustituir valores en procesos Connect:Direct definidos por el usuario. Para seguir el convenio de denominación de Connect:Direct, todas las variables simbólicas intrínsecas utilizadas por Managed File Transfer tienen el formato %FTE seguido de cinco caracteres alfanuméricos en mayúsculas.

Al crear un proceso para transferir archivos desde un nodo Connect:Direct al sistema del puente Connect:Direct, debe utilizar la variable intrínseca %FTETFILE como el valor de TO FILE en el proceso Connect:Direct. Al crear un proceso para transferir archivos a un nodo Connect:Direct desde el sistema del puente Connect:Direct, debe utilizar la variable intrínseca %FTEFFILE como el valor de FROM FILE en el proceso Connect:Direct. Estas variables contienen las vías de acceso de archivos temporales que el agente de puente Connect:Direct utiliza para las transferencias dentro y fuera de la red de Managed File Transfer.

Para obtener más información sobre las variables simbólicas intrínsecas, consulte la documentación del producto Connect:Direct.

Procesos Connect:Direct de ejemplo

Managed File Transfer proporciona procesos de Connect:Direct de ejemplo. Estos ejemplos se encuentran en el directorio siguiente: *MQ_INSTALLATION_PATH*/mqft/samples/ConnectDirectProcessTemplates.

Tareas relacionadas

"Especificación del proceso Connect:Direct que se debe iniciar utilizando el archivo ConnectDirectProcessDefinition.xml" en la página 331
Especifique qué proceso Connect:Direct hay que iniciar como parte de una transferencia de Managed File Transfer. Managed File Transfer proporciona un archivo XML que puede editar para especificar definiciones de proceso.

"Utilizar variables simbólicas intrínsecas en procesos Connect:Direct invocados por Managed File Transfer" en la página 332
Puede llamar a un proceso Connect:Direct definido por el usuario desde una transferencia de Managed File Transfer y pasar información de la transferencia al proceso Connect:Direct, utilizando variables simbólicas intrínsecas en la definición de proceso.

Referencia relacionada

Formato del archivo de definiciones de proceso Connect:Direct

Variables de sustitución para usar con procesos Connect:Direct definidos por el usuario

Especificación del proceso Connect:Direct que se debe iniciar utilizando el archivo ConnectDirectProcessDefinition.xml

Especifique qué proceso Connect:Direct hay que iniciar como parte de una transferencia de Managed File Transfer. Managed File Transfer proporciona un archivo XML que puede editar para especificar definiciones de proceso.

Acerca de esta tarea

El mandato **fteCreateCDAgent** crea el archivo ConnectDirectProcessDefinitions.xml en el directorio de configuración del agente *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*cd_bridge_agent_name*. Antes de poder llamar a procesos Connect:Direct definidos por el usuario desde el agente de puente Connect:Direct, debe configurar definiciones de proceso editando este archivo.

Para cada proceso que desee especificar para llamarlo como parte de una transferencia a través del puente Connect:Direct, realice los pasos siguientes:

Procedimiento

1. Defina el proceso Connect:Direct al que desea que el agente de puente Connect:Direct llame como parte de la transferencia y guarde la plantilla de proceso en un archivo.
2. Abra el archivo *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*cd_bridge_agent_name*/ConnectDirectProcessDefinitions.xml en un editor de texto.
3. Cree un elemento <processSet>.
4. Dentro del elemento <processSet>, cree un elemento <condition>.
5. Dentro del elemento <condition>, cree uno o más elementos que definan una condición de que la solicitud de transferencia debe coincidir para llamar al proceso de Connect:Direct que ha definido en el Paso 1. Estos elementos pueden ser elementos <match> o elementos <defined>.
   - Utilice un elemento <match> para especificar que el valor de una variable debe coincidir con un patrón. Cree el elemento <match> con los atributos siguientes:
     - variable - el nombre de la variable cuyo valor se compara. La variable es un símbolo intrínseco. Si desea más información, consulte Variables de sustitución para su uso con procesos Connect:Direct definidos por el usuario.
     - value - el patrón con el que comparar el valor de la variable especificada.
     - Opcional: pattern - el tipo de patrón utilizado por el valor del atributo value. Este tipo de patrón puede ser wildcard o regex. Este atributo es opcional y el valor predeterminado es wildcard.
   - Utilice un elemento <defined> para especificar que una variable debe tener un valor definido. Cree el elemento <defined> con el atributo siguiente:
     - variable - el nombre de la variable que debe tener un valor definido. La variable es un símbolo intrínseco. Si desea más información, consulte Variables de sustitución para su uso con procesos Connect:Direct definidos por el usuario.

   Las condiciones especificadas en el elemento <condition> se combinan con un AND lógico. Se deben cumplir todas las condiciones para que el agente de puente Connect:Direct llame al proceso especificado por este elemento <processSet>. Si no especifica un elemento <condition>, el conjunto de procesos coincide con todas las transferencias.
6. Dentro del elemento <processSet>, cree un elemento <process>.
7. Dentro del elemento <process>, cree un elemento <transfer>.

   El elemento transfer especifica el proceso Connect:Direct al que el agente de puente Connect:Direct llama como parte de la transferencia. Cree el elemento <transfer> con el atributo siguiente:

   - process- -la ubicación del proceso Connect:Direct que ha definido en el paso 1. La ubicación de este archivo se especifica con una vía de acceso absoluta o relativa al directorio *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*cd_bridge_agent_name*.

Resultados

Al buscar una coincidencia de condición, el agente de puente Connect:Direct busca desde el principio del archivo al final del archivo. La primera coincidencia que se encuentra es la que se utiliza.

Tareas relacionadas

Configurar el puente Connect:Direct

Referencia relacionada

Formato del archivo de definiciones de proceso Connect:Direct

fteCreateCDAgent: crear un agente de puente Connect:Direct

Utilizar variables simbólicas intrínsecas en procesos Connect:Direct invocados por Managed File Transfer

Puede llamar a un proceso Connect:Direct definido por el usuario desde una transferencia de Managed File Transfer y pasar información de la transferencia al proceso Connect:Direct, utilizando variables simbólicas intrínsecas en la definición de proceso.

Acerca de esta tarea

Este ejemplo utiliza variables simbólicas intrínsecas para pasar información de una transferencia de Managed File Transfer a un proceso Connect:Direct definido por el usuario. Si desea más información sobre variables simbólicas intrínsecas utilizadas por Managed File Transfer, consulte Variables de sustitución para ser utilizadas con procesos Connect:Direct definidos por el usuario.

En este ejemplo, el archivo se transfiere desde un Managed File Transfer Agent a un nodo de puente Connect:Direct. La primera parte de la transferencia la realiza Managed File Transfer. La segunda parte de la transferencia la realiza un proceso Connect:Direct definido por el usuario.

Procedimiento

1. Cree un proceso Connect:Direct que utilice variables simbólicas intrínsecas.

```
%FTEPNAME PROCESS
    SNODE=%FTESNODE
    PNODEID=(%FTEPUSER,%FTEPPASS)
    SNODEID=(%FTESUSER,%FTESPASS)

COPY001 COPY
    FROM (
        FILE=%FTEFFILE
        DISP=%FTEFDISP
    )
    TO (
        FILE=%FTETFILE
        DISP=%FTETDISP
    )
PEND
```

2. Guarde este proceso en un archivo de texto en la siguiente ubicación: *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*cd_bridge_agent*/Example.cdp
3. Edite el archivo ConnectDirectProcessDefinition.xml para incluir una regla que llame al proceso Connect:Direct que ha creado en el paso 1.

```
<?xml version="1.0" encoding="UTF-8"?>
<tns:cdprocess xmlns:tns="http://wmqfte.ibm.com/ConnectDirectProcessDefinitions"
               xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
               xsi:schemaLocation="http://wmqfte.ibm.com/
                ConnectDirectProcessDefinitions ConnectDirectProcessDefinitions.xsd">


    <tns:processSet>
        <tns:condition>
            <tns:match variable="%FTESNODE" value="TOBERMORY" pattern="wildcard" />
        </tns:condition>
        <tns:process>
            <tns:transfer process="Example.cdp" />
        </tns:process>
    </tns:processSet>

</tns:cdprocess>
```

   En este ejemplo, si se envía una solicitud de transferencia al agente de puente Connect:Direct que tiene TOBERMORY como su nodo Connect:Direct de origen o de destino, se llama al proceso Example.cdpConnect:Direct.

4. Envíe una solicitud de transferencia de archivos que satisfaga las condiciones que ha definido en el archivo ConnectDirectProcessDefinition.xml en el Paso 3.

   Por ejemplo,

```
fteCreateTransfer -sa ORINOCO -da CD_BRIDGE
                  -sm QM_WIMBLEDON -dm QM_COMMON
                  -de overwrite -df TOBERMORY:/home/bulgaria/destination.txt
                  -sd leave c:\bungo\source.txt
```

   En este ejemplo, el nodo Connect:Direct de destino es TOBERMORY. Este nodo es el nodo secundario de la transferencia y el valor de %FTESNODE se establece en TOBERMORY. Este mandato coincide con la condición establecida en el archivo ConnectDirectProcessDefinition.xml.

5. Managed File Transfer transfiere el archivo de origen a una ubicación temporal en el mismo sistema que el agente de puente Connect:Direct.
6. El agente de puente Connect:Direct establece los valores de las variables simbólicas intrínsecas a partir de la información de la solicitud de transferencia y la información de configuración.

   Las variables simbólicas intrínsecas se establecen en los siguientes valores:

   - %FTEPNAME=*nombre_proceso* - Este valor es un nombre de proceso de 8 caracteres generado por el agente de puente Connect:Direct.
   - %FTESNODE=TOBERMORY - Este valor se establece a partir del parámetro **-df** del mandato **fteCreateTransfer**.
   - %FTEPUSER,=*primary_node_user* - Esta información se toma del archivo `ConnectDirectCredentials.xml`.
   - %FTEPPASS=*primary_node_user_password* - Esta información se toma del archivo `ConnectDirectCredentials.xml`.
   - %FTESUSER, =*secondary_node_user* - Esta información se toma del archivo `ConnectDirectCredentials.xml`.
   - %FTESPASS=*secondary_node_user_password* - Esta información se toma del archivo `ConnectDirectCredentials.xml` .
   - %FTEFFILE =*ubicación_temporal* - Este valor es la ubicación temporal del archivo en el mismo sistema que el agente de puente Connect:Direct.
   - %FTEFDISP=leave - Este valor se establece a partir del parámetro **-sd** del mandato **fteCreateTransfer**.
   - %FTETFILE=`/home/bulgaria/destination.txt` - Este valor se establece a partir del parámetro **-df** del mandato **fteCreateTransfer**.
   - %FTETDISP=overwrite - Este valor se establece a partir del parámetro **-de** del mandato **fteCreateTransfer**.
7. El proceso Connect:Direct se inicia en el nodo de puente Connect:Direct. Connect:Direct transfiere el archivo de la ubicación temporal en el sistema del puente Connect:Direct al destino `/home/bulgaria/destination.txt` en el sistema donde se ejecuta el nodo Connect:Direct TOBERMORY.

Conceptos relacionados

"Cómo someter un proceso Connect:Direct definido por el usuario desde una solicitud de transferencia de archivos" en la página 330

Puede someter una solicitud de transferencia para una transferencia que pasa a través del agente de puente Connect:Direct que llame a un proceso Connect:Direct definido por el usuario como parte de la transferencia de archivos.

Referencia relacionada

Variables de sustitución para usar con procesos Connect:Direct definidos por el usuario

Utilización de procesos Connect:Direct para someter solicitudes de transferencia de Managed File Transfer

Puede someter una solicitud de transferencia al agente de puente Connect:Direct desde un proceso Connect:Direct. Managed File Transfer proporciona mandatos a los que se puede llamar desde una sentencia **RUN TASK** en un proceso Connect:Direct .

Managed File Transfer proporciona los siguientes mandatos para utilizarlos con procesos Connect:Direct:

ftetag
: Especifique este mandato en un paso que preceda al mandato **ftebxfer** o **ftecxfer** para crear la información de auditoría necesaria para la transferencia. Este mandato acepta la especificación de origen de la transferencia como parámetro. Para obtener información sobre el formato de la especificación de origen, consulte **fteCreateTransfer**: iniciar una nueva transferencia de archivos.

ftebxfer
Especifique este mandato para crear una solicitud de transferencia de archivos cuando el gestor de colas al que se somete la solicitud de transferencia esté en el mismo sistema que el nodo Connect:Direct que somete el mandato. Este mandato acepta los mismos parámetros que el mandato **fteCreateTransfer**. Si desea más información sobre estos parámetros, consulte **fteCreateTransfer**: iniciar una nueva transferencia de archivos. Este mandato también tiene un parámetro adicional:

-qmgrname
Necesario. El nombre del gestor de colas al que someter el mandato.

ftecxfer
Especifique este mandato para crear una solicitud de transferencia de archivos cuando el gestor de colas al que se somete la solicitud de transferencia no esté ubicado en el mismo sistema que el nodo Connect:Direct que somete el mandato. Este mandato acepta los mismos parámetros que el mandato **fteCreateTransfer**. Si desea información sobre los parámetros, consulte **fteCreateTransfer**: iniciar una nueva transferencia de archivos. Este mandato tiene también tres parámetros adicionales:

-qmgrname
Necesario. El nombre del gestor de colas al que someter el mandato.

-connname
Necesario. El host y el puerto del gestor de colas al que someter el mandato, especificados en formato CONNAME de IBM MQ. Por ejemplo, `host.example.com(1337)`.

-channelname
Opcional. El nombre del canal que se va a utilizar para conectarse al gestor de colas al que someter el mandato. Si no se especifica este parámetro, se utiliza un valor predeterminado de SYSTEM.DEF.SVRCONN.

Tareas relacionadas

"Crear y someter un proceso Connect:Direct que llame a Managed File Transfer utilizando Connect:Direct Requester" en la página 335
Connect:Direct Requester es una interfaz gráfica de usuario que puede utilizar para crear y someter un proceso Connect:Direct que llame a Managed File Transfer.

Referencia relacionada

Ejemplo: un archivo de proceso Connect:Direct que llama a mandatos MFT

Crear y someter un proceso Connect:Direct que llame a Managed File Transfer utilizando Connect:Direct Requester

Connect:Direct Requester es una interfaz gráfica de usuario que puede utilizar para crear y someter un proceso Connect:Direct que llame a Managed File Transfer.

Acerca de esta tarea

Esta tarea describe cómo crear un proceso Connect:Direct que llame al mandato Managed File Transfer **ftecxfer** o el mandato **ftebxfer**. Utilice el mandato **ftecxfer** cuando el gestor de colas al que se envía la solicitud de transferencia se encuentre en un sistema distinto de el del nodo Connect:Direct que somete el mandato. Utilice el mandato **ftebxfer** cuando el gestor de colas al que se envía la solicitud de transferencia se encuentre en el mismo sistema que el del nodo Connect:Direct que somete el mandato. El mandato **ftecxfer** establece una conexión de cliente al gestor de colas de agente del agente de origen de la transferencia. Antes de llamar al mandato **ftecxfer**, debe llamar al mandato **ftetag** y pasarle la información de especificación de origen. Esto permite registrar y auditar el proceso de la misma manera que las transferencias iniciadas desde Managed File Transfer.

Procedimiento

1. Inicie Connect:Direct Requester.
2. En la pestaña **Nodos** del panel, seleccione el nodo Connect:Direct que se utiliza como el nodo primario del proceso.

3. Seleccione **Archivo** > **Nuevo** > **Proceso**. Se abre la ventana **Propiedades de proceso**.
4. En el campo **Nombre:**, escriba el nombre del proceso.
5. Seleccione el nodo secundario en la lista **Nombre de** > **Snode:**.
6. Seleccione el sistema operativo del nodo secundario en la lista **Sistema operativo de** > **Snode:**.
7. Opcional: Rellene cualquier información adicional de esta ventana que necesite.
8. Pulse **Aceptar**. Se cierra la ventana **Propiedades de proceso**.
9. Cree una sentencia que ejecute el mandato Managed File Transfer **ftetag**.
   a) En la ventana **Proceso**, pulse el botón derecho del ratón en la sentencia **End**.
   b) Seleccione **Insertar** > **Run Task**. Se abre la ventana **Sentencia Run Task**.
   c) En el campo **Etiqueta:**, escriba `Tag`.
   d) En el campo **Parámetros o mandatos opcionales** , escriba `pgm(`*MQ_INSTALLATION_PATH*`/bin/ftetag) args(`*source_specification*`)`. Si desea más información sobre el formato de *especificación_origen*, consulte **fteCreateTransfer**: iniciar una nueva transferencia de archivos.
   e) Pulse **Aceptar**. Se cierra la ventana **Sentencia Run Task**.
10. Cree una sentencia que ejecute los mandatos Managed File Transfer **ftecxfer** o **ftebxfer**.
    a) En la ventana **Proceso**, pulse el botón derecho del ratón en la sentencia **End**.
    b) Seleccione **Insertar** > **Run Task**. Se abre la ventana **Sentencia Run Task**.
    c) En el campo **Etiqueta:**, escriba `Transfer`.
    d) En el campo **Parámetros o mandatos opcionales** , escriba `pgm(`*MQ_INSTALLATION_PATH*`/bin/ftecxfer) args(`*parameters*`)` o `pgm(`*MQ_INSTALLATION_PATH*`/bin/ftebxfer) args(`*parameters*`)` en función del mandato que elija. Los parámetros utilizados por los mandatos **ftecxfer** y **ftebxfer** son los mismos que los parámetros utilizados por el mandato **fteCreateTransfer**, además de algunos parámetros adicionales específicos de **ftecxfer** y **ftebxfer**. Si desea más información, consulte **fteCreateTransfer**: iniciar una nueva transferencia de archivos y “Utilización de procesos Connect:Direct para someter solicitudes de transferencia de Managed File Transfer” en la página 334.
    e) Pulse **Aceptar**. Se cierra la ventana **Sentencia Run Task**.
11. Opcional: Cree todas las sentencias adicionales que necesite.
12. Someta el proceso.
    a) Pulse el botón derecho del ratón en la ventana **Proceso**.
    b) Seleccione **Someter**. Se abre la ventana **Adjunto Connect:Direct**.
    c) Entre el nombre de usuario y la contraseña que va a utilizar para ejecutar el proceso.
    d) Pulse **Aceptar**.

Conceptos relacionados

“Utilización de procesos Connect:Direct para someter solicitudes de transferencia de Managed File Transfer” en la página 334

Puede someter una solicitud de transferencia al agente de puente Connect:Direct desde un proceso Connect:Direct. Managed File Transfer proporciona mandatos a los que se puede llamar desde una sentencia **RUN TASK** en un proceso Connect:Direct .

Cómo trabajar con MFT desde IBM Integration Bus

Puede trabajar con Managed File Transfer desde IBM Integration Bus utilizando los nodos FTEOutput y FTEInput.

- Utilice el nodo FTEInput para transferir un archivo a través de la red utilizando Managed File Transfer y luego procesar dicho archivo como parte de un flujo de bus de integración.
- Utilice el nodo FTEOutput para transferir un archivo resultado de un flujo de bus de integración a otra ubicación en la red.

Los agentes que transfieren archivos a o desde el agente de intermediario pueden estar a cualquier nivel de Managed File Transfer.

Para obtener más información, consulte la documentación del productoIBM Integration Bus.

Recuperación y reinicio de MFT

Si el agente o el gestor de colas están no disponibles por algún motivo, por ejemplo, debido a un fallo en el suministro o en la red, Managed File Transfer se recupera tal como se indica a continuación en los siguientes escenarios:

- Normalmente, si se produce un problema mientras se está transfiriendo un archivo, Managed File Transfer recupera y reinicia esa transferencia de archivos una vez que se ha solucionado el problema.
- Si un archivo que estaba en proceso de ser transferido se suprime o se modifica mientras el agente o el gestor de colas no están disponibles, la transferencia falla y recibirá un mensaje en el registro de transferencias que proporciona detalles sobre la anomalía.
- Si un proceso de agente falla durante una transferencia de archivos, la transferencia continuará cuando reinicie el agente.
- Si un agente pierde la conexión con el gestor de colas de agente, el agente espera mientras intenta volver a conectarse al gestor de colas. Cuando el agente se reconecta satisfactoriamente al gestor de colas, la transferencia actual continúa.
- Si por algún motivo se detiene el agente, los supervisores de recursos asociados a un agente dejan de sondear. Cuando el agente se recupera, los supervisores también se reinician y se reanuda el sondeo de recursos.
- Para una transferencia de archivos que tenga una disposición de origen de `delete`, si se produce una recuperación tras enviar todos los datos de un agente de origen a un agente de destino, el archivo de origen se desbloquea antes de la supresión. Este desbloqueo significa que el archivo de origen podría modificarse posiblemente antes de que se suprima el archivo. Por tanto, si se considera que no es seguro suprimir el archivo de origen, se muestra el siguiente aviso:

  ```
  BFGTR0075W: The source file has not been deleted because it is possible that the source file
  was modified after the source file was transferred.
  ```

  En este caso, verifique que el contenido del archivo de origen no se haya modificado y, a continuación, suprímalo manualmente.

Puede comprobar el estado de las transferencias en IBM MQ Explorer. Si alguna transferencia aparece como `Stalled`, es posible que tenga que realizar una acción correctiva porque el estado detenido indica un problema con el agente o entre los dos agentes implicados en la transferencia.

Tareas relacionadas

“Establecimiento de un tiempo de espera para la recuperación de transferencias estancadas” en la página 337

Puede establecer un tiempo de espera de recuperación de transferencia para las transferencias de archivos estancadas que se aplique a todas las transferencias para un agente de origen. También puede establecer un tiempo de espera de recuperación de transferencia para una transferencia individual. Si establece una cantidad de tiempo específica, en segundos, durante la cual un agente de origen sigue intentando recuperar una transferencia de archivos estancada y la transferencia no se realiza correctamente cuando el agente alcanza el tiempo de espera, la transferencia falla.

Establecimiento de un tiempo de espera para la recuperación de transferencias estancadas

Puede establecer un tiempo de espera de recuperación de transferencia para las transferencias de archivos estancadas que se aplique a todas las transferencias para un agente de origen. También puede establecer un tiempo de espera de recuperación de transferencia para una transferencia individual. Si establece una cantidad de tiempo específica, en segundos, durante la cual un agente de origen

sigue intentando recuperar una transferencia de archivos estancada y la transferencia no se realiza correctamente cuando el agente alcanza el tiempo de espera, la transferencia falla.

Acerca de esta tarea

Puede establecer un tiempo de espera de recuperación de transferencia que se aplique a todas las transferencias para un agente de origen añadiendo un parámetro de tiempo de espera de recuperación de transferencia al archivo `agent.properties` del agente. También puede establecer un tiempo de espera de recuperación de transferencia para una transferencia individual desde la línea de mandatos o con IBM MQ Explorer o utilizando las tareas de Apache Ant. Si hay establecido un valor de tiempo de espera de recuperación de transferencia en el archivo `agent.properties`, el establecimiento del tiempo de espera de recuperación de transferencia para una transferencia individual prevalece sobre el valor en el archivo `agent.properties`.

Hay tres opciones pera el tiempo de espera de recuperación de transferencia:

- El agente continúa intentando recuperar la transferencia estancada hasta que se completa correctamente. Este comportamiento es el mismo que el comportamiento predeterminado del agente si el tiempo de espera de recuperación de transferencia no está establecido.
- El agente marca la transferencia como fallada inmediatamente al entrar en la recuperación.
- El agente sigue reintentando la transferencia estancada durante el tiempo especificado antes de que la transferencia se marque como fallada.

Es opcional establecer el tiempo de espera de recuperación de transferencia. Si no lo establece, las transferencias siguen el comportamiento predeterminado, donde el agente sigue intentando recuperar una transferencia estancada hasta que se realiza correctamente.

Conceptos relacionados

“Recuperación y reinicio de MFT” en la página 337
Si el agente o el gestor de colas están no disponibles por algún motivo, por ejemplo, debido a un fallo en el suministro o en la red, Managed File Transfer se recupera tal como se indica a continuación en los siguientes escenarios:

Conceptos de tiempo de espera de recuperación de transferencia

Puede establecer la cantidad de tiempo, en segundos, durante el cual un agente de origen sigue intentando recuperar una transferencia de archivo estancada. Si la transferencia no resulta satisfactoria cuando el agente alcanza el tiempo de espera para el intervalo de reintento, falla la transacción.

Prioridad de tiempo de espera de recuperación

El valor de tiempo de espera de recuperación de transferencia para una transferencia individual especificado a través de los mandatos **fteCreateTransfer**, **fteCreateTemplate** o **fteCreateMonitor**, o utilizando IBM MQ Explorer, o tal como se ha especificado en el elemento anidado **fte:filespec**, tiene prioridad sobre el valor que se ha especificado para el parámetro **transferRecoveryTimeout** en el archivo `agent.properties` para el agente de origen.

Por ejemplo, si se inicia el mandato **fteCreateTransfer** sin el par parámetro **-rt** y el par de valor, el agente de origen AGENT1 comprueba en el archivo `agent.properties` el valor de **transferRecoveryTimeout** con el fin de determinar el comportamiento del tiempo de espera de recuperación:

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -df C:\import\transferredfile.txt
C:\export\originalfile.txt
```

Si el parámetro **transferRecoveryTimeout** del archivo `agent.properties` no se ha establecido o se ha establecido en `-1`, el agente sigue el comportamiento predeterminado e intenta recuperar la transferencia hasta que se lleve a cabo de forma satisfactoria.

Sin embargo, si el mandato **fteCreateTransfer** incluye el parámetro **-rt**, el valor de este parámetro tiene prioridad sobre el valor del archivo `agent.properties` y se utiliza como valor de tiempo de espera de recuperación para la transferencia:

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -rt 21600 -df C:\import\transferredfile.txt
C:\export\originalfile.txt
```

Contador de tiempo de espera de recuperación

El contador del tiempo de espera de recuperación se inicia cuando la transferencia entra en estado de recuperación. Se publica un mensaje de registro de transferencia en SYSTEM.FTE con la serie de tema `Log/agent_name/transfer_ID` para indicar que el estado de la transferencia ha cambiado a recuperación y la hora del reloj del agente de origen a la que ha cambiado el estado. Si la transferencia se reanuda dentro del intervalo de reintento y no alcanza el tiempo de espera de recuperación (counter<=recovery timeout), se restablecerá el contador en 0, listo para volver a iniciarse si la transferencia entra en fase de recuperación.

Si el contador alcanza el valor máximo establecido para el tiempo de espera de recuperación (counter==recovery timeout), la recuperación de la transferencia se detendrá y el agente de origen informará de que la transferencia ha fallado. Este tipo de anomalía de transferencia, provocada por el hecho de que la transferencia ha alcanzado el tiempo de espera de recuperación, se indica mediante el código de mensaje, `RECOVERY TIMEOUT (69)`. Se publica otro mensaje de registro de trasferencias en el tema SYSTEM.FTE, con una serie de tema de `Log/agent_name/transfer_ID`, para indicar que ha fallado la transferencia e incluye un mensaje, el código de retorno y el registro de sucesos del agente de origen. El registro de sucesos del agente de origen se actualiza con un mensaje cuando se produce alguna de las situaciones siguientes durante la recuperación:

- Cuando el parámetro de tiempo de espera de recuperación se establece en un valor superior a `-1`, la transferencia entra en fase de recuperación. El registro de sucesos del agente se actualiza para indicar el inicio del temporizador de la recuperación para **TransferId** y la cantidad de tiempo que espera el agente de origen hasta que inicia el proceso de tiempo de espera de recuperación.
- Cuando se reanuda la transferencia, se actualiza el registro de sucesos del agente de origen con un mensaje nuevo para indicar que se ha reanudado **TransferId** que estaba en recuperación.
- Cuando se agota el tiempo de espera de recuperación de transferencia, el registro de sucesos del agente de origen se actualiza para indicar que ha fallado **TransferId** durante la recuperación debido al tiempo de espera de recuperación.

Estos mensajes de registro permiten que los usuarios (suscriptores y registradores) identifiquen las transferencias que han fallado debido al tiempo de espera de recuperación de transferencias.

El contador del tiempo de espera de recuperación siempre está en el agente de origen. Sin embargo, si el agente de destino no consigue recibir información del agente de origen de forma puntual, puede enviar una solicitud al agente de origen para que ponga la transferencia en fase de recuperación. Para una transferencia cuya opción de tiempo de espera de recuperación se ha establecido, el agente de origen inicia el contador del tiempo de espera de recuperación cuando recibe la solicitud del agente de destino.

Sigue necesitándose una gestión manual para las transferencias que no utilizan la opción de tiempo de espera de recuperación, las transferencias fallidas y las que se han completado parcialmente.

Para conjuntos de transferencias cuya solicitud de transferencia única se emite para varios archivos, y algunos archivos se han completado correctamente pero uno se ha completado parcialmente, la transferencia seguirá marcada como fallida porque no se ha completado según lo previsto. Es posible que el agente de origen haya agotado el tiempo de espera durante la transferencia parcial del archivo.

Asegúrese de que el agente de destino y el servidor de archivos estén listos y que tengan un estado para aceptar transferencias de archivos.

Debe emitir de nuevo la solicitud de transferencia para todo el conjunto pero para evitar problemas debido a que algunos archivos permanecen en el destino desde el intento inicial de transferencia, puede emitir la nueva solicitud especificando la opción `sobrescribir si existe`. De este modo se asegura

de que se borre el conjunto de archivos incompletos del intento de transferencia anterior como parte de la transferencia nueva, antes de grabar de nuevo los archivos en el destino.

A partir de IBM MQ 9.1.5, ya no es necesario eliminar manualmente los archivos de partes que quedan en un destino después de que un intento de transferencia inicial haya fallado. Si se establece un tiempo de espera de recuperación de transferencia para una transferencia, el agente de origen cambia la transferencia al estado RecoveryTimedOut si se excede el tiempo de espera de la recuperación de transferencia. Después de que se haya resincronizado la transferencia, el agente de destino elimina los archivos de partes que se crearon durante la transferencia y envía un mensaje de finalización al agente de origen.

Rastreos y mensajes

Los puntos de rastreo se incluyen con fines de diagnóstico. Se registra el valor de tiempo de espera de recuperación, el inicio del intervalo de reintentos, el inicio del periodo de reanudación y el restablecimiento del contador, así como si se ha agotado el tiempo de espera de la transferencia y si ha resultado fallida. En el caso de que hubiera problemas o un comportamiento inesperado, puede recopilar el registro de salida del agente de origen y los archivos de rastreo, y proporcionarlos cuando se lo solicite el personal de soporte de IBM para ayudarle a resolver problemas.

Los mensajes le avisan cuando:

- Una transferencia entra en recuperación (BFGTR0081I)
- Una transferencia ha terminado porque ha excedido el tiempo de espera de la recuperación (BFGSS0081E)
- Se reanuda una transferencia después de haber estado en recuperación (BFGTR0082I)

Conceptos relacionados

“Recuperación y reinicio de MFT” en la página 337
Si el agente o el gestor de colas están no disponibles por algún motivo, por ejemplo, debido a un fallo en el suministro o en la red, Managed File Transfer se recupera tal como se indica a continuación en los siguientes escenarios:

Establecimiento del tiempo de espera de recuperación de transferencia para todas las transferencias de un agente de origen

Puede establecer un tiempo de espera de recuperación de transferencia que se aplique a todas las transferencias de un agente de origen añadiendo el parámetro **transferRecoveryTimeout** al archivo `agent.properties`.

Acerca de esta tarea

Para establecer un tiempo de espera de recuperación de transferencia que se aplique a todas las transferencias de un agente de origen, añada el par parámetro y valor para **transferRecoveryTimeout** al archivo agent.properties.

Hay tres opciones para el parámetro **transferRecoveryTimeout**:

-1
El agente sigue intentando recuperar la transferencia estancada hasta que ésta resulta satisfactoria. La utilización de esta opción equivale al comportamiento predeterminado del agente cuando la propiedad no se ha establecido.

0
El agente detiene la transferencia de archivo tan pronto como se inicia la recuperación.

>0
El agente sigue intentando recuperar la transferencia estancada durante el periodo de tiempo en segundos según se haya establecido mediante el valor entero positivo especificado.

Los cambios que realice en el archivo `agent.properties` entrarán en vigor sólo después de reiniciar el agente.

Si es necesario, puede alterar temporalmente el valor de tiempo de espera de recuperación de transferencia en el archivo `agent.properties` para una transferencia individual. Para obtener más información, consulte "Establecimiento del tiempo de espera de recuperación de transferencia para transferencias individuales" en la página 341.

Procedimiento

- Para especificar que el agente sigue intentando recuperar la transferencia estancada hasta que se complete correctamente, establezca un valor de tiempo de espera de recuperación de transferencia de `-1`, tal como se muestra en el ejemplo siguiente:

  ```
  transferRecoveryTimeout=-1
  ```

- Para especificar que el agente marca la transferencia como fallida inmediatamente después de entrar en la recuperación, establezca un valor de tiempo de espera de recuperación de transferencia de `0`, tal como se muestra en el ejemplo siguiente:

  ```
  transferRecoveryTimeout=0
  ```

- Para especificar que el agente sigue reintentando una transferencia estancada durante una cantidad determinada de tiempo antes de que la transferencia se marque como fallida, establezca un valor de tiempo de espera de recuperación de transferencia durante el periodo de tiempo, en segundos, que desea que el agente siga reintentando.

  Por ejemplo, si se establece un valor de tiempo de espera de recuperación de transferencia en `21600` significa que el agente sigue intentando recuperar la transferencia durante seis horas desde el momento en que entra en recuperación:

  ```
  transferRecoveryTimeout=21600
  ```

  El valor máximo para este parámetro es `999999999`.

Establecimiento del tiempo de espera de recuperación de transferencia para transferencias individuales

Puede establecer un tiempo de espera de recuperación de transferencia para una transferencia individual desde la línea de mandatos o con IBM MQ Explorer o utilizando las tareas de Apache Ant. Si hay un valor de tiempo de espera de recuperación de transferencia establecido en el archivo `agent.properties`, establecer el tiempo de espera de recuperación de transferencia para una transferencia individual altera temporalmente el valor establecido en el archivo `agent.properties`.

Acerca de esta tarea

Puede establecer el parámetro de tiempo de espera de recuperación de transferencia para una transferencia individual cuando:

- Esté creando una transferencia utilizando el mandato **`fteCreateTransfer`** o utilizando IBM MQ Explorer.
- Esté creando una plantilla de transferencia utilizando el mandato **`fteCreateTemplate`** o utilizando IBM MQ Explorer.
- Esté creando un supervisor de recursos utilizando el mandato **`fteCreateMonitor`** o utilizando IBM MQ Explorer.
- Copiar o mover archivos utilizando las tareas fte: filecopy o fte: filemove Ant .

Si establece un valor de tiempo de espera de recuperación de transferencia para una transferencia individual, este valor altera temporalmente el valor de tiempo de espera de recuperación de transferencia establecido en el archivo `agent.properties` (consulte"Establecimiento del tiempo de espera de recuperación de transferencia para todas las transferencias de un agente de origen" en la página 340).

Procedimiento

- Para utilizar el mandato **fteCreateTransfer** o **fteCreateTemplate** para establecer el tiempo de espera de recuperación de transferencia, especifique la opción adecuada para el parámetro **-rt**:

  -1
  El agente sigue intentando recuperar la transferencia estancada hasta que ésta resulta satisfactoria. La utilización de esta opción equivale al comportamiento predeterminado del agente cuando la propiedad no se ha establecido.

  0
  El agente detiene la transferencia de archivo tan pronto como se inicia la recuperación.

  >0
  El agente sigue intentando recuperar la transferencia estancada durante el periodo de tiempo en segundos.

  Ejemplos del mandato **fteCreateTransfer**

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -rt -1 -df C:\import\transferredfile.txt
C:\export\originalfile.txt
```

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -rt 0 -df C:\import\transferredfile.txt
C:\export\originalfile.txt
```

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -rt 21600 -df C:\import\transferredfile.txt
C:\export\originalfile.txt
```

  Ejemplos del mandato **fteCreateTemplate**

```
fteCreateTemplate -tn "payroll accounts monthly report template" -rt -1 -sa PAYROLL -sm
QM_PAYROLL1 -da ACCOUNTS
-dm QM_ACCOUNTS -df C:\payroll_reports\*.xls C:\out\*.xls
```

```
fteCreateTemplate -tn "payroll accounts monthly report template" -rt 0 -sa PAYROLL -sm
QM_PAYROLL1 -da ACCOUNTS
-dm QM_ACCOUNTS -df C:\payroll_reports\*.xls C:\out\*.xls
```

```
fteCreateTemplate -tn "payroll accounts monthly report template" -rt 21600 -sa PAYROLL -sm
QM_PAYROLL1 -da ACCOUNTS
-dm QM_ACCOUNTS -df C:\payroll_reports\*.xls C:\out\*.xls
```

  No hay ningún parámetro **-rt** para el mandato **fteCreateMonitor**. Si establece el parámetro **-rt** con el mandato **fteCreateTransfer** y también establece el parámetro **-gt**, se incluirá el parámetro de tiempo de espera de recuperación en el documento XML con la definición de transferencia que se genera cuando se ejecuta el mandato **fteCreateTransfer**. A continuación, el supervisor de recursos utiliza este documento XML cuando ejecuta el mandato **fteCreateMonitor**. En el ejemplo siguiente, los detalles de tiempo de espera de recuperación de transferencia se incluirán en el archivo `task.xml`:

```
fteCreateMonitor -ma AgentName -md C:\mqmft\monitors -mn Monitor_Name -mt task.xml -tr
"fileSize>=5MB,*.zip"
```

- Para utilizar la página del asistente de Nueva transferencia, Nuevo supervisor o Nueva plantilla de IBM MQ Explorer para establecer el tiempo de espera de recuperación de transferencia, seleccione la opción necesaria en el campo **Tiempo de espera excedido de recuperación de transferencia** (segundos):

  Como agente de origen
  Si selecciona **Como agente de origen**, se utiliza el valor del parámetro **transferRecoveryTimeout** del archivo `agent.properties` si está establecido, de lo contrario se aplica el comportamiento predeterminado para el tiempo de espera de recuperación de transferencia.

Recuadro de lista numérico
Si especifica un tiempo en segundos en el recuadro de lista numérico, el agente continúa intentando recuperar la transferencia estancada durante el periodo de tiempo especificado.

Ninguna
Si selecciona **None**, no se establece ningún tiempo de espera de recuperación de transferencia y el agente sigue intentando recuperar la transferencia estancada hasta que la transferencia resulta satisfactoria.

- Para establecer el tiempo de espera de recuperación utilizando tareas de Ant . incluir la opción **transferRecoveryTimeout** y el valor, con los elementos **fte:filecopy** o **fte:filemove** para mover o copiar archivos, por ejemplo:

  Ejemplo de **fte:filecopy**

```
<fte:filecopy cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
              src="agent1@qm1" dst="agent2@qm2"
              rcproperty="copy.result" transferRecoveryTimeout="0">

    <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/file.bin"/>

</fte:filecopy>
```

  Ejemplo de **fte:filemove**

```
<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>
```

Conceptos relacionados

"Recuperación y reinicio de MFT" en la página 337
Si el agente o el gestor de colas están no disponibles por algún motivo, por ejemplo, debido a un fallo en el suministro o en la red, Managed File Transfer se recupera tal como se indica a continuación en los siguientes escenarios:

Tareas relacionadas

"Inicio de una nueva transferencia de archivos" en la página 221
Puede iniciar una nueva de transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos y puede elegir transferir un único archivo o varios archivos de un grupo.

"Creación de una plantilla de transferencia de archivos mediante IBM MQ Explorer" en la página 268
Puede crear una plantilla de transferencia de archivos desde IBM MQ Explorer o desde la línea de mandatos. A continuación, puede utilizar la plantilla para crear nuevas transferencias de archivos utilizando los detalles de plantilla o someter la plantilla para empezar la transferencia de archivos.

"Supervisión de recursos de MFT" en la página 233
Puede supervisar recursos de Managed File Transfer; por ejemplo, una cola o un directorio. Cuando se cumple una condición en este recurso, el supervisor de recursos inicia una tarea, por ejemplo una transferencia de archivos. Puede crear un supervisor de recursos utilizando el mandato **fteCreateMonitor** o la vista **Supervisores** en el plug-in de Managed File Transfer para IBM MQ Explorer.

"Visualización del estado de transferencias de archivos en el Registro de transferencias" en la página 231
Puede ver los detalles de las transferencias de archivos utilizando el **Registro de transferencias** en IBM MQ Explorer. Puede tratarse de transferencias iniciadas desde la línea de mandatos o desde IBM MQ Explorer. También puede personalizar lo que aparece en el **Registro de transferencias**.

Referencia relacionada

El archivo MFT `agent.properties`
fteCreateTransfer: iniciar una nueva transferencia de archivos
fteCreateTemplate: crear nueva plantilla de transferencia de archivos
fteCreateMonitor: crear un supervisor de recursos de MFT

Tarea Ant fte:filecopy
Tarea Ant fte:filemove

Windows Linux AIX Administración de MQ Telemetry

MQ Telemetry se administra mediante IBM MQ Explorer o en una línea de mandatos. Utilice el explorador para configurar canales de telemetría, controlar el servicio de telemetría y supervisar los clientes MQTT que se conecten a IBM MQ. Configure la seguridad de MQ Telemetry utilizando JAAS, TLS y el gestor de autorizaciones sobre objetos de IBM MQ.

Administración utilizando IBM MQ Explorer

Utilice el explorador para configurar canales de telemetría, controlar el servicio de telemetría y supervisar los clientes MQTT que se conecten a IBM MQ. Configure la seguridad de MQ Telemetry utilizando JAAS, TLS y el gestor de autorizaciones sobre objetos de IBM MQ.

Administración utilizando la línea de mandatos

MQ Telemetry se puede administrar por completo en la línea de mandatos mediante los mandatos MQSC.

La documentación de MQ Telemetry también contiene scripts de ejemplo en los que se demuestra la utilización básica de la aplicación cliente de IBM MQ Telemetry Transport v3.

Lea y comprenda los ejemplos en Programas de ejemplo de IBM MQ Telemetry Transport antes de utilizarlos.

Conceptos relacionados

MQ Telemetry

Referencia relacionada

Propiedades de MQXR

Linux AIX Configurar un gestor de colas para telemetría en Linux y AIX

Siga estos pasos para configurar MQ Telemetry manualmente. Si sólo necesita una configuración simple que utilice el ID de usuario invitado, en su lugar puede ejecutar el asistente de soporte de MQ Telemetry en IBM MQ Explorer.

Antes de empezar

Si sólo necesita una configuración simple, considere la posibilidad de utilizar el soporte de MQ Telemetry en IBM MQ Explorer. Este soporte incluye un asistente y un procedimiento de mandato de ejemplo `sampleMQM`. Estos recursos configuran una configuración inicial utilizando el ID de usuario invitado. Consulte Verificación de la instalación de MQ Telemetry utilizando los programas de ejemplo IBM MQ Explorer y IBM MQ Telemetry Transport.

Si necesita una configuración más compleja que utilice un método de autenticación diferente, utilice los pasos de esta tarea. Empiece con los pasos iniciales siguientes:

1. Consulte Consideraciones sobre la instalación para MQ Telemetry para obtener información sobre cómo instalar IBM MQy la característica MQ Telemetry .
2. Cree e inicie un gestor de colas. El gestor de colas se conoce como *qMgr* en esta tarea.
3. Como parte de esta tarea, debe configurar el servicio de telemetría (MQXR). Los valores de propiedad MQXR se almacenan en un archivo de propiedades específico de la plataforma: `mqxr_win.properties`. Normalmente, no es necesario editar el archivo de propiedades MQXR directamente, ya que casi todos los valores se pueden configurar mediante mandatos de administración de MQSC o IBM MQ Explorer. Si decide editar directamente el archivo, detenga el gestor de colas antes de realizar los cambios. Consulte Propiedades MQXR.

Acerca de esta tarea

Siga los pasos de esta tarea para configurar MQ Telemetry manualmente, utilizando distintos esquemas de autorización.

Procedimiento

1. Abra una ventana de mandatos en el directorio de ejemplos de telemetría.

   El directorio de ejemplos de telemetría es `/opt/mqm/mqxr/samples`.
2. Cree la cola de transmisión de telemetría.

   Si `SYSTEM.MQTT.TRANSMIT.QUEUE` no existe, se crea automáticamente cuando se inicia por primera vez el servicio de telemetría (MQXR) y se establece para utilizar el ID de usuario invitado. Sin embargo, esta tarea configura MQ Telemetry para utilizar un esquema de autorización diferente. Para esta tarea, cree `SYSTEM.MQTT.TRANSMIT.QUEUE`y configure el acceso al mismo antes de iniciar el servicio de telemetría (MQXR).

   Ejecute el siguiente mandato:

   ```
   echo "DEFINE QLOCAL('SYSTEM.MQTT.TRANSMIT.QUEUE') USAGE(XMITQ) MAXDEPTH(100000)" | runmqsc qMgr
   ```
3. Establezca la cola de transmisión predeterminada.

   Es más fácil enviar mensajes directamente a los clientes MQTT si `SYSTEM.MQTT.TRANSMIT.QUEUE` es la cola de transmisión predeterminada. De lo contrario, debe añadir una definición de cola remota para cada cliente que reciba mensajes de IBM MQ ; consulte “Envío de un mensaje a un cliente directamente” en la página 350. Tenga en cuenta que la modificación de la cola de transmisión predeterminada puede interferir con la configuración existente.

   Cuando el servicio de telemetría (MQXR) se inicia por primera vez, no establece `SYSTEM.MQTT.TRANSMIT.QUEUE` como cola de transmisión predeterminada para el gestor de colas. Para configurar este valor, altere la propiedad de cola de transmisión predeterminada. Puede hacerlo utilizando la IBM MQ Explorero ejecutando el mandato siguiente:

   ```
   echo "ALTER QMGR DEFXMITQ('SYSTEM.MQTT.TRANSMIT.QUEUE')" | runmqsc qMgr
   ```
4. Siga el procedimiento indicado en “Autorizar a clientes MQTT a acceder a objetos de IBM MQ” en la página 352 para crear uno o varios ID de usuario. Los ID de usuario tienen autorización para publicar, suscribirse y enviar publicaciones a clientes MQTT.
5. Edite el archivo `installMQXRService_unix.mqsc` para configurar el archivo de claves que se utiliza para cifrar la frase de contraseña para canales MQTT TLS:
   a) Abra el archivo *`WMQ program installation directory`*`/mqxr/samples/installMQXRService_unix.mqsc`.
   b) Localice la línea que incluye el parámetro **STARTARG**y edite la opción **-sf** para especificar la ubicación del archivo de claves de credenciales.

      De forma predeterminada, el archivo `installMQXRService_unix.mqsc` utiliza un archivo de claves predeterminado denominado `[DEFAULT]`. El archivo de claves predeterminado es el mismo para todas las instalaciones de IBM MQ , por lo que debe proporcionar un archivo de claves que sea exclusivo para la instalación cuando cifre las frases de contraseña.

      Consulte también el código de ejemplo en “Creación del SYSTEM.MQXR.SERVICE” en la página 346.
6. Instale el servicio de telemetría (MQXR) ejecutando el mandato siguiente:

   ```
   cat /opt/<install_dir>/mqxr/samples/installMQXRService_unix.mqsc | runmqsc queue_manager
   ```
7. Inicie el servicio.

```
echo "START SERVICE(SYSTEM.MQXR.SERVICE)" | runmqsc qMgr
```

El servicio de telemetría (MQXR) se inicia automáticamente cuando se inicia el gestor de colas. En esta tarea se inicia manualmente, porque el gestor de colas ya se está ejecutando.

8. Utilizando IBM MQ Explorer, configure los canales de telemetría para que acepten las conexiones de clientes MQTT.

   Los canales de telemetría deben configurarse de forma que sus identidades sean uno de los ID de usuario definidos en el paso “4” en la página 345.

   Consulte también DEFINE CHANNEL (MQTT).

9. Verifique la configuración ejecutando el cliente de ejemplo.

   Para que el cliente de ejemplo funcione con el canal de telemetría, el canal debe autorizar al cliente a publicar, suscribirse y recibir publicaciones. El cliente de ejemplo se conecta al canal de telemetría en el puerto 1883 de forma predeterminada. Consulte también Programas de ejemplo de IBM MQ Telemetry Transport.

Creación del SYSTEM.MQXR.SERVICE

Utilice el mandato **runMQXRService** para crear el SYSTEM.MQXR.SERVICE.

```
DEF    SERVICE(SYSTEM.MQXR.SERVICE) +
CONTROL(QMGR) +
DESCR('Manages clients using MQXR protocols such as MQTT') +
SERVTYPE(SERVER) +
STARTCMD('+MQ_INSTALL_PATH+/mqxr/bin/runMQXRService.sh') +
STARTARG('-m +QMNAME+ -d "+MQ_Q_MGR_DATA_PATH+" -g "+MQ_DATA_PATH+" -sf "/home/keyFileLocation/
keyFile.txt"') +
STOPCMD('+MQ_INSTALL_PATH+/mqxr/bin/endMQXRService.sh') +
STOPARG('-m +QMNAME+') +
STDOUT('+MQ_Q_MGR_DATA_PATH+/mqxr.stdout') +
STDERR('+MQ_Q_MGR_DATA_PATH+/mqxr.stderr')
```

Windows Configurar un gestor de colas para telemetría en Windows

Siga estos pasos para configurar MQ Telemetry manualmente. Si sólo necesita una configuración simple que utilice el ID de usuario invitado, en su lugar puede ejecutar el asistente de soporte de MQ Telemetry en IBM MQ Explorer.

Antes de empezar

Si sólo necesita una configuración simple, considere la posibilidad de utilizar el soporte de MQ Telemetry en IBM MQ Explorer. Este soporte incluye un asistente y un procedimiento de mandato de ejemplo sampleMQM. Estos recursos configuran una configuración inicial utilizando el ID de usuario invitado. Consulte Verificación de la instalación de MQ Telemetry utilizando los programas de ejemplo IBM MQ Explorer y IBM MQ Telemetry Transport.

Si necesita una configuración más compleja que utilice un método de autenticación diferente, utilice los pasos de esta tarea. Empiece con los pasos iniciales siguientes:

1. Consulte Consideraciones sobre la instalación para MQ Telemetry para obtener información sobre cómo instalar IBM MQy la característica MQ Telemetry .
2. Cree e inicie un gestor de colas. El gestor de colas se conoce como *qMgr* en esta tarea.
3. Como parte de esta tarea, debe configurar el servicio de telemetría (MQXR). Los valores de propiedad MQXR se almacenan en un archivo de propiedades específico de la plataforma: mqxr_win.properties. Normalmente, no es necesario editar el archivo de propiedades MQXR directamente, ya que casi todos los valores se pueden configurar mediante mandatos de administración de MQSC o IBM MQ Explorer. Si decide editar directamente el archivo, detenga el gestor de colas antes de realizar los cambios. Consulte Propiedades MQXR.

Acerca de esta tarea

Siga los pasos de esta tarea para configurar MQ Telemetry manualmente, utilizando distintos esquemas de autorización.

Procedimiento

1. Abra una ventana de mandatos en el directorio de ejemplos de telemetría.

   El directorio de ejemplos de telemetría es *WMQ program installation directory*\mqxr\samples.
2. Cree la cola de transmisión de telemetría.

   Si SYSTEM.MQTT.TRANSMIT.QUEUE no existe, se crea automáticamente cuando se inicia por primera vez el servicio de telemetría (MQXR) y se establece para utilizar el ID de usuario invitado. Sin embargo, esta tarea configura MQ Telemetry para utilizar un esquema de autorización diferente. Para esta tarea, cree SYSTEM.MQTT.TRANSMIT.QUEUEy configure el acceso al mismo antes de iniciar el servicio de telemetría (MQXR).

   Ejecute el siguiente mandato:

   ```
   echo DEFINE QLOCAL('SYSTEM.MQTT.TRANSMIT.QUEUE') USAGE(XMITQ) MAXDEPTH(100000) | runmqsc qMgr
   ```
3. Establezca la cola de transmisión predeterminada.

   Es más fácil enviar mensajes directamente a los clientes MQTT si SYSTEM.MQTT.TRANSMIT.QUEUE es la cola de transmisión predeterminada. De lo contrario, debe añadir una definición de cola remota para cada cliente que reciba mensajes de IBM MQ ; consulte “Envío de un mensaje a un cliente directamente” en la página 350. Tenga en cuenta que la modificación de la cola de transmisión predeterminada puede interferir con la configuración existente.

   Cuando el servicio de telemetría (MQXR) se inicia por primera vez, no establece SYSTEM.MQTT.TRANSMIT.QUEUE como cola de transmisión predeterminada para el gestor de colas. Para configurar este valor, altere la propiedad de cola de transmisión predeterminada. Puede hacerlo utilizando la IBM MQ Explorero ejecutando el mandato siguiente:

   ```
   echo ALTER QMGR DEFXMITQ('SYSTEM.MQTT.TRANSMIT.QUEUE') | runmqsc qMgr
   ```
4. Siga el procedimiento indicado en “Autorizar a clientes MQTT a acceder a objetos de IBM MQ” en la página 352 para crear uno o varios ID de usuario. Los ID de usuario tienen autorización para publicar, suscribirse y enviar publicaciones a clientes MQTT.
5. Edite el archivo `installMQXRService_win.mqsc` para configurar el archivo de claves que se utiliza para cifrar la frase de contraseña para canales MQTT TLS:
   a) Abra el archivo *WMQ program installation directory*\mqxr\samples\installMQXRService_win.mqsc.
   b) Localice la línea que incluye el parámetro **STARTARG**y edite la opción **-sf** para especificar la ubicación del archivo de claves de credenciales.

      De forma predeterminada, el archivo `installMQXRService_win.mqsc` utiliza un archivo de claves predeterminado denominado `[DEFAULT]`. El archivo de claves predeterminado es el mismo para todas las instalaciones de IBM MQ , por lo que debe proporcionar un archivo de claves que sea exclusivo para la instalación cuando cifre las frases de contraseña.

      Consulte también el código de ejemplo en “Creando SYSTEM.MQXR.SERVICE” en la página 348.
6. Instale el servicio de telemetría (MQXR) ejecutando el mandato siguiente:

   ```
   type installMQXRService_win.mqsc | runmqsc qMgr
   ```
7. Inicie el servicio.

```
echo START SERVICE(SYSTEM.MQXR.SERVICE) | runmqsc qMgr
```

El servicio de telemetría (MQXR) se inicia automáticamente cuando se inicia el gestor de colas. En esta tarea se inicia manualmente, porque el gestor de colas ya se está ejecutando.

8. Utilizando IBM MQ Explorer, configure los canales de telemetría para que acepten las conexiones de clientes MQTT.

   Los canales de telemetría deben configurarse de forma que sus identidades sean uno de los ID de usuario definidos en el paso “4” en la página 347.

   Consulte también DEFINE CHANNEL (MQTT).

9. Verifique la configuración ejecutando el cliente de ejemplo.

   Para que el cliente de ejemplo funcione con el canal de telemetría, el canal debe autorizar al cliente a publicar, suscribirse y recibir publicaciones. El cliente de ejemplo se conecta al canal de telemetría en el puerto 1883 de forma predeterminada. Consulte también Programas de ejemplo de IBM MQ Telemetry Transport.

Creando SYSTEM.MQXR.SERVICE

Utilice el mandato **runMQXRService** para crear el SYSTEM.MQXR.SERVICE.

```
DEF    SERVICE(SYSTEM.MQXR.SERVICE) +
CONTROL(QMGR) +
DESCR('Manages clients using MQXR protocols such as MQTT') +
SERVTYPE(SERVER) +
STARTCMD('+MQ_INSTALL_PATH+\mqxr\bin\runMQXRService.bat') +
STARTARG('-m +QMNAME+ -d "+MQ_Q_MGR_DATA_PATH+\." -g "+MQ_DATA_PATH+\." -sf
"c:\keyFileLocation\keyFile.txt"') +
STOPCMD('+MQ_INSTALL_PATH+\mqxr\bin\endMQXRService.bat') +
STOPARG('-m +QMNAME+') +
STDOUT('+MQ_Q_MGR_DATA_PATH+\mqxr.stdout') +
STDERR('+MQ_Q_MGR_DATA_PATH+\mqxr.stderr')
```

Windows Linux AIX Configuración de las colas distribuidas para enviar mensajes a clientes MQTT

Las aplicaciones de IBM MQ pueden enviar mensajes de clientes MQTT v3 publicando en una suscripción creada por un cliente, o enviando un mensaje directamente. Cualquiera que sea el método que se utilice, el mensaje se coloca en SYSTEM.MQTT.TRANSMIT.QUEUE y se envía al cliente mediante el servicio de telemetría (MQXR). Existen varias formas de colocar un mensaje en SYSTEM.MQTT.TRANSMIT.QUEUE.

Publicación de un mensaje en respuesta a una suscripción de cliente MQTT

El servicio de telemetría (MQXR) crea una suscripción en nombre del cliente MQTT. El cliente es el destino de las publicaciones que coincidan con la suscripción que envíe el cliente. Los servicios de telemetría reenvían las publicaciones coincidentes al cliente.

Un cliente MQTT se conecta a IBM MQ como un gestor de colas, con su nombre de gestor de colas establecido en su >IdentificadorCliente. El destino de las publicaciones que se envían al cliente es una cola de transmisión, SYSTEM.MQTT.TRANSMIT.QUEUE. El servicio de telemetría reenvía mensajes en SYSTEM.MQTT.TRANSMIT.QUEUE a clientes MQTT , utilizando el nombre del gestor de colas de destino como clave para un cliente específico.

El servicio de telemetría (MQXR) abre la cola de transmisión utilizando IdentificadorCliente como el nombre del gestor de colas. El servicio de telemetría (MQXR) pasa el manejador de objeto de la cola a la llamada MQSUB, para reenviar las publicaciones que coincidan con la suscripción del cliente. En la resolución del nombre de objeto, se crea el ClientIdentifier como el nombre del gestor de colas remoto, y la cola de transmisión debe resolverse en SYSTEM.MQTT.TRANSMIT.QUEUE. Utilizando la resolución de nombres de objeto IBM MQ estándar, *ClientIdentifier* se resuelve de la forma siguiente; consulte Tabla 16 en la página 349.

1. *ClientIdentifier* no coincide con nada.

   ClientIdentifier es un nombre de gestor de colas remoto. No coincide con el nombre del gestor de colas local, con un alias de gestor de colas, ni con un nombre de cola de transmisión.
   El nombre de cola no se ha definido. Actualmente, el servicio de telemetría (MQXR) establece SYSTEM.MQTT.PUBLICATION.QUEUE como el nombre de la cola. Un cliente MQTT v3 no da soporte a colas, por lo que el cliente ignora el nombre de cola que se ha resuelto.
   El nombre de la propiedad del gestor de colas local, `Cola de transmisión predeterminada`, se debe establecer en SYSTEM.MQTT.TRANSMIT.QUEUE, de modo que la publicación se coloque en SYSTEM.MQTT.TRANSMIT.QUEUE, desde donde se enviará al cliente.

2. *ClientIdentifier* coincide con un alias de gestor de colas denominado *ClientIdentifier*.

   ClientIdentifier es un nombre de gestor de colas remoto. Coincide con el nombre de un alias de gestor de colas.
   El alias del gestor de colas debe definirse con *ClientIdentifier* como el nombre del gestor de colas remoto.
   Al establecer el nombre de cola de transmisión en la definición de alias de gestor de colas, no es necesario que la transmisión predeterminada se establezca en SYSTEM.MQTT.TRANSMIT.QUEUE.

Tabla 16. Resolución de nombres de un alias de gestor de colas MQTT

	Entrada		**Salida**		
ClientIdentifier	**Nombre del gestor de colas**	**Nombre de cola**	**Nombre del gestor de colas**	**Nombre de cola**	**Cola de transmisión**
No coincide con nada	*ClientIdentifier*	*sin definir*	*ClientIdentifier*	*sin definir*	Cola de transmisión predeterminada. SYSTEM.MQTT.TRANSMIT.QUEUE
Coincide con un alias de gestor de colas denominado *ClientIdentifier*	*ClientIdentifier*	*sin definir*	*ClientIdentifier*	*sin definir*	SYSTEM.MQTT.TRANSMIT.QUEUE

Para obtener más información sobre la resolución de nombres, consulte Resolución de nombres.

Cualquier programa de IBM MQ puede publicar en el mismo tema. La publicación se envía a sus suscriptores, incluidos los clientes MQTT v3 que tengan una suscripción al tema.

Si se crea un tema administrativo en un clúster, con el atributo CLUSTER(*clusterName*), cualquier aplicación del clúster puede publicar en el cliente; por ejemplo:

```
echo DEFINE TOPIC('MQTTExamples') TOPICSTR('MQTT Examples') CLUSTER(MQTT) REPLACE | runmqsc qMgr
```

Nota: No proporcione a SYSTEM.MQTT.TRANSMIT.QUEUE un atributo de clúster.

Los suscriptores y publicadores de cliente MQTT pueden conectarse a gestores de colas distintos. Los suscriptores y publicadores pueden formar parte del mismo clúster, o pueden conectarse mediante una jerarquía de publicación/suscripción. La publicación se entrega del publicador al suscriptor utilizando IBM MQ.

Envío de un mensaje a un cliente directamente

Una alternativa a que un cliente cree una suscripción y reciba una publicación que coincida con el tema de suscripción, es enviar un mensaje a un cliente MQTT v3 directamente. Las aplicaciones de cliente MQTT V3 no pueden enviar mensajes directamente, pero otras aplicaciones como, por ejemplo, las aplicaciones de IBM MQ, sí pueden.

La aplicación de IBM MQ debe conocer el `IdentificadorCliente` del cliente MQTT v3. Como los clientes MQTT v3 no tienen colas, el nombre de cola de destino se pasa al método MQTT v3 cliente de aplicaciones `messageArrived` como nombre de tema. Por ejemplo, en un programa MQI, cree un descriptor de objetos con el cliente como `ObjectQmgrName`:

```
MQOD.ObjectQmgrName = ClientIdentifier ;
MQOD.ObjectName = name ;
```

Si la aplicación se escribe utilizando JMS, cree un destino de punto a punto; por ejemplo:

JM 3.0

```
jakarta.jms.Destination jmsDestination =
(jakarta.jms.Destination)jmsFactory.createQueue
("queue://ClientIdentifier/name");
```

JMS 2.0

```
javax.jms.Destination jmsDestination =
(javax.jms.Destination)jmsFactory.createQueue
("queue://ClientIdentifier/name");
```

Para enviar un mensaje no solicitado a un cliente MQTT, utilice una definición de cola remota. El nombre del gestor de colas remoto debe resolverse en el `ClientIdentifier` del cliente. La cola de transmisión debe resolverse en `SYSTEM.MQTT.TRANSMIT.QUEUE`; consulte Tabla 17 en la página 350. El nombre de la cola remota puede ser cualquier cosa. El cliente lo recibe como una serie de tema.

Tabla 17. Resolución de nombres de una definición de cola remota de cliente MQTT

Entrada		Salida		
Nombre de cola	**Nombre del gestor de colas**	**Nombre de cola**	**Nombre del gestor de colas**	**Cola de transmisión**
Nombre de la definición de la cola remota	En blanco o nombre de gestor de colas local	Nombre de cola remota utilizado como una serie de tema	*ClientIdentifier*	SYSTEM.MQTT.TRANSMIT.QUEUE

Si el cliente está conectado, el mensaje se envía directamente al cliente MQTT, que llama al método `messageArrived`; consulte Método messageArrived.

Si el cliente se ha desconectado con una sesión persistente, el mensaje se almacena en `SYSTEM.MQTT.TRANSMIT.QUEUE` ; consulte Sesiones sin estado y con estado deMQTT. Se reenvía al cliente cuando éste se vuelve a conectar a la sesión.

Si envía un mensaje no persistente, éste se envía al cliente con una calidad de servicio de `Como máximo una vez`, `QoS=0`. Si envía un mensaje persistente directamente a un cliente, de forma predeterminada, se envía con la calidad de servicio `exactamente una vez`, `QoS=2`. Como es posible que el cliente no tenga un mecanismo de persistencia, el cliente puede reducir la calidad de servicio que acepta para los mensajes enviados directamente. Para reducir la calidad de servicio para los mensajes enviados directamente a un cliente, suscríbase al tema `DEFAULT.QoS`. Especifique la calidad de servicio máxima a la que el cliente puede dar soporte.

Windows Linux AIX Identificación, autorización y autenticación de clientes MQTT

El servicio de telemetría (MQXR) publica temas de IBM MQ, o se suscribe a ellos, en nombre de clientes MQTT, utilizando canales MQTT. El administrador de IBM MQ configura la identidad de canal MQTT que se utiliza para la autorización de IBM MQ. El administrador debe definir una identidad común para el canal o utilizar el `Username` o `ClientIdentifier` de un cliente conectado al canal.

El servicio de telemetría (MQXR) puede autenticar el cliente utilizando el valor `Username` que proporciona el cliente o bien utilizando un certificado de cliente. El valor de `Username` se autentica utilizando una contraseña que proporciona el cliente.

En resumen: la identificación de cliente es la selección de la identidad de cliente. En función del contexto, el cliente se identifica con el valor `ClientIdentifier`, el valor `Username`, una identidad común de cliente que crea el administrador o un certificado de cliente. El identificador de cliente utilizado para la comprobación de la autenticidad no tiene que ser el mismo identificador que se utiliza para la autorización.

Los programas de cliente MQTT establecen el valor de `Username` y de `Password` que se envían al servidor utilizando un canal MQTT. También pueden establecer las propiedades TLS que se necesitan para cifrar y autenticar la conexión. El administrador decide si autenticar el canal MQTT y cómo autenticarlo.

Para autorizar a un cliente MQTT a acceder a objetos de IBM MQ, autorice el valor `ClientIdentifier`, o el valor `Username` del cliente, o autorice una identidad común del cliente. Para permitir que un cliente se conecte a IBM MQ, autentique el valor de `Username`, o utilice un certificado de cliente. Configure JAAS para autenticar el valor de `Username`, y configure TLS para autenticar un certificado de cliente.

Si define un valor `Password` en el cliente, cifre la conexión utilizando VPN, o configure el canal MQTT para que utilice TLS, para mantener la confidencialidad de la contraseña.

Resulta difícil gestionar certificados de clientes. Por este motivo, si los riesgos que conlleva la autenticación por contraseña resultan aceptables, ésta se utiliza a menudo para autenticar a los clientes.

Si existe una manera segura de gestionar y almacenar el certificado de cliente, se puede confiar en la autenticación con certificados. Sin embargo los certificados no suelen gestionarse de forma segura en los entornos en los que se utiliza la telemetría. En su lugar la autenticación de dispositivos que utilizan certificados de cliente se complementa con la autenticación de las contraseñas de cliente en el servidor. Debido a la complejidad adicional, la utilización de certificados de cliente se limita a aplicaciones altamente confidenciales. El uso de formas de autenticación se conoce como autenticación por dos factores. Debe conocer uno de los factores, por ejemplo, una contraseña, y tener otro, por ejemplo, un certificado.

En una aplicación en la que se necesite mucha confidencialidad como, por ejemplo, los dispositivos que utilicen la tecnología "chip and pin", el dispositivo se bloquea durante la fabricación para evitar intrusiones no autorizadas en el hardware y software internos. En el dispositivo se copia un certificado de cliente de confianza y duración limitada. El dispositivo se despliega en la ubicación en la que va a utilizarse. Cada vez que se utiliza el dispositivo, se realiza más autenticación, usando una contraseña u otro certificado de una tarjeta inteligente.

Windows Linux AIX Identidad y autorización de cliente MQTT

Utilice el ID de cliente, `Username`, o una identidad de cliente común para autorizar el acceso a objetos de IBM MQ.

El administrador de IBM MQ tiene tres opciones para seleccionar la identidad del canal MQTT. El administrador hace la elección al definir o modificar el canal MQTT que utiliza el cliente. La identidad se utiliza para autorizar el acceso a temas de IBM MQ. La elección se realiza en el orden siguiente:

1. El ID de cliente (consulte USECLNTID).
2. Una identidad que el administrador proporciona para el canal (el MCAUSER del canal). Consulte MCAUSER).

3. Si ninguna de las opciones anteriores es aplicable, el `Username` que se pasa desde el cliente MQTT (`Username` es un atributo de la clase `MqttConnectOptions`. Debe establecerse antes de que el cliente se conecte al servicio. Su valor predeterminado es nulo.

Evitar problemas: A la identidad elegida mediante este proceso se le hace referencia posteriormente en, por ejemplo, el mandato DISPLAY CHSTATUS (MQTT), como el MCAUSER del cliente. Tenga en cuenta que esta no es necesariamente la misma identidad que el MCAUSER del canal al que se hace referencia en la opción (2).

Utilice el mandato de IBM MQ **setmqaut** para seleccionar qué objetos, y qué acciones, están autorizados para que los utilice la identidad asociada al canal MQTT. Por ejemplo, el siguiente código autoriza a la identidad del canal `MQTTClient`, proporcionada por el administrador del gestor de colas `QM1`:

```
 setmqaut -m QM1 -t q -n SYSTEM.MQTT.TRANSMIT.QUEUE -p MQTTClient -all +put
setmqaut -m QM1 -t topic -n SYSTEM.BASE.TOPIC -p MQTTClient -all +pub +sub
```

Windows Linux AIX

Autorizar a clientes MQTT a acceder a objetos de IBM MQ

Siga estos pasos para autorizar a clientes MQTT a publicar y suscribirse a objetos de IBM MQ. Los pasos siguen cuatro patrones de control de acceso alternativos.

Antes de empezar

A los clientes MQTT se les autoriza a acceder a objetos en IBM MQ asignándoles una identidad cuando se conectan a un canal de telemetría. El administrador de IBM MQ configura el canal de telemetría utilizando IBM MQ Explorer para asignar a un cliente uno de los tres tipos de identidad siguientes:

1. `ClientIdentifier`
2. `Nombre de usuario`
3. Un nombre que el administrador asigna al canal.

Sea cual sea el tipo que se utilice, la identidad debe estar definida en IBM MQ como principal por el servicio de autorización instalado. El servicio de autorización predeterminado en Windows o Linux se denomina OAM (Gestor de autorizaciones sobre objetos). Si va a utilizar el OAM, la identidad debe definirse como un ID de usuario.

Utilice la identidad para otorgar a un cliente, o grupo de clientes, permiso para publicar o suscribirse a temas definidos en IBM MQ. Si un cliente MQTT se ha suscrito a un tema, utilice la identidad para otorgarle permiso para recibir las publicaciones resultantes.

Resulta difícil gestionar un sistema con decenas de miles de clientes MQTT, ya que cada uno necesita permiso de acceso individual. Una solución es definir identidades comunes y asociar clientes MQTT individuales a una de las identidades comunes. Defina tantas identidades comunes como sea necesario para definir diferentes combinaciones de permisos. Otra solución es escribir un servicio de autorización propio que maneje los miles de usuarios de forma más sencilla a como lo hace el sistema operativo.

Puede combinar clientes MQTT en identidades comunes de dos formas, utilizando el OAM:

1. Defina varios canales de telemetría, cada uno con un ID de usuario diferente que el administrador asigna utilizando IBM MQ. Los clientes que se conectan utilizando diferentes números de puertos TCP/IP se asocian a diferentes canales de telemetría y se les asignan diferentes identidades.
2. Defina un único canal de telemetría, pero haga que cada cliente seleccione un valor de `Username` entre un conjunto pequeño de ID de usuario. El administrador configura el canal de telemetría para seleccionar el valor de `Username` del cliente como su identidad.

En esta tarea, la identidad del canal de telemetría se denomina *`mqttUser`*, independientemente de cómo se establezca. Si las colecciones de clientes utilizan identidades diferentes, utilice varios *`mqttUsers`*, uno para cada colección de clientes. Puesto que la tarea utiliza OAM, cada *`mqttUser`* debe ser un ID de usuario.

Acerca de esta tarea

En esta tarea puede elegir entre cuatro patrones de control de acceso, que puede adaptar a requisitos específicos. Los patrones se diferencian en su granularidad del control de acceso.

- "Sin control de acceso" en la página 353
- "Control de acceso de granularidad gruesa" en la página 353
- "Control de acceso de granularidad media" en la página 353
- "Control de acceso de granularidad precisa" en la página 353

El resultado de los modelos es asignar a *mqttUsers* conjuntos de permisos para publicarlos y suscribirlos a IBM MQ y recibir publicaciones de IBM MQ.

Sin control de acceso
A los clientes MQTT se les asigna autorización administrativa de IBM MQ, y pueden realizar cualquier acción en cualquier objeto.

Procedimiento

1. Cree un ID de usuario *mqttUser* para que actúe como la identidad de todos los clientes de MQTT.
2. Añada *mqttUser* al grupo mqm; consulte Adición de un usuario a un grupo en Windows o Creación y gestión de grupos en Linux

Control de acceso de granularidad gruesa
Los clientes MQTT tienen autorización para publicar y suscribirse, y para enviar mensajes a clientes MQTT. No tienen autorización para realizar otras acciones, ni para acceder a otros objetos.

Procedimiento

1. Cree un ID de usuario *mqttUser* para que actúe como la identidad de todos los clientes de MQTT.
2. Autorice a *mqttUser* a publicar y suscribirse a todos los temas y a enviar publicaciones a los clientes de MQTT.

```
setmqaut -m qMgr -t topic -n SYSTEM.BASE.TOPIC -p mqttUser -all +pub +sub
setmqaut -m qMgr -t q -n SYSTEM.MQTT.TRANSMIT.QUEUE -p mqttUser -all +put
```

Control de acceso de granularidad media
Los clientes MQTT se dividen en diferentes grupos para publicar y suscribirse a diferentes conjuntos de temas, y para enviar mensajes a clientes MQTT.

Procedimiento

1. Cree varios ID de usuario *mqttUsers*, y varios temas administrativos en el árbol de temas de publicación/suscripción.
2. Autorice a diferentes *mqttUsers* para temas diferentes.

```
setmqaut -m qMgr -t topic -n topic1 -p mqttUserA -all +pub +sub
setmqaut -m qMgr -t topic -n topic2 -p mqttUserB -all +pub +sub
```

3. Cree un grupo *mqtt*, y añada todos los *mqttUsers* al grupo.
4. Autorice a *mqtt* a enviar temas a clientes de MQTT.

```
setmqaut -m qMgr -t q -n SYSTEM.MQTT.TRANSMIT.QUEUE -p mqtt -all +put
```

Control de acceso de granularidad precisa
Los clientes MQTT se incorporan en un sistema de control de acceso existente que autoriza a los grupos a realizar acciones sobre objetos.

Acerca de esta tarea

Se asigna un ID de usuario a uno o más grupos de sistema operativo en función de las autorizaciones que necesite. Si las aplicaciones de IBM MQ publican o se suscriben al mismo espacio de temas que los clientes MQTT, utilice este modelo. Se hace referencia a los grupos como `Publish X`, `Subscribe Y` y `mqtt`.

`Publish X`
Los miembros de grupos de `Publish X` pueden publicar en *`topicX`*.

`Subscribe Y`
Los miembros de los grupos `Subscribe Y` pueden suscribirse a *`topicY`*.

`mqtt`
Los miembros del grupo *`mqtt`* pueden enviar publicaciones a clientes de MQTT.

Procedimiento

1. Cree varios grupos, `Publish X` y `Subscribe Y`, que estén asignados a varios temas administrativos en el árbol de temas de publicación/suscripción.
2. Cree un grupo `mqtt`.
3. Cree varios ID de usuario *`mqttUsers`* y agregue los usuarios a cualquiera de los grupos, dependiendo qué estén autorizados a hacer.
4. Autorice distintos grupos `Publish X` y `Subscribe X` a distintos temas, y autorice al grupo *`mqtt`* para enviar mensajes a clientes MQTT .

```
setmqaut -m qMgr -t topic -n topic1 -p Publish X -all +pub
setmqaut -m qMgr -t topic -n topic1 -p Subscribe X -all +pub +sub
setmqaut -m qMgr -t q -n SYSTEM.MQTT.TRANSMIT.QUEUE -p mqtt -all +put
```

Windows Linux AIX Autenticación de cliente MQTT mediante una contraseña

Autentique el valor de `Username` utilizando la contraseña del cliente. Puede autenticar el cliente utilizando una identidad diferente a la utilizada para autorizar al cliente a publicar temas y a suscribirse a los mismos.

El servicio de telemetría (MQXR) utiliza JAAS para autenticar el valor `Username` del cliente. JAAS utiliza la `Contraseña` proporcionada por el cliente MQTT.

El administrador de IBM MQ decide si se debe autenticar el valor de `Username`, o no autenticar nada, configurando el canal MQTT al que se conecta un cliente. Los clientes pueden asignarse a diferentes canales y cada canal puede configurarse para que autentique sus clientes de formas diferentes. Mediante JAAS, puede configurar qué métodos deben autenticar el cliente y cuáles pueden hacerlo de forma opcional.

La opción que elija para autenticar la identidad no afecta a la opción que elija para autorizar la identidad. Es posible que desee configurar una identidad común para la autorización para facilitar las tareas administrativas, pero que se autentique cada usuario para que utilice dicha identidad. En el procedimiento siguiente se describen los pasos para autenticar los usuarios individuales para que utilice una identidad común:

1. El administrador de IBM MQ establece la identidad de canal de MQTT en cualquier nombre, por ejemplo `MQTTClientUser`, utilizando IBM MQ Explorer.
2. El administrador de IBM MQ autoriza a `MQTTClient` a publicar y a suscribirse a cualquier tema:

```
 setmqaut -m QM1 -t q -n SYSTEM.MQTT.TRANSMIT.QUEUE -p MQTTClient -all +put
setmqaut -m QM1 -t topic -n SYSTEM.BASE.TOPIC -p MQTTClient -all +pub +sub
```

3. El desarrollador de aplicaciones cliente MQTT crea un objeto `MqttConnectOptions` y establece `Username` y `Password` antes de conectarse al servidor.
4. El desarrollador de seguridad crea un `LoginModule` JAAS para autenticar el valor `Username` con el valor `Password` y lo incluye en el archivo de configuración JAAS.
5. El administrador de IBM MQ configura el canal MQTT para autenticar el valor `UserName` del cliente utilizando JAAS.

Windows Linux AIX Autenticación de cliente MQTT mediante TLS

Las conexiones entre el cliente MQTT y el gestor de colas las inicia siempre el cliente MQTT. El cliente MQTT es siempre el cliente SSL. Tanto la autenticación de cliente del servidor como la autenticación de servidor del cliente MQTT son opcionales.

Al proporcionar al cliente un certificado digital privado firmado, puede autenticar al cliente MQTT paraIBM MQ . El administrador de IBM MQ puede forzar a los clientes MQTT a autenticarse ellos mismos en el gestor de colas utilizando TLS. Sólo puede solicitar la autenticación de cliente como parte de una autenticación mutua.

Algunos tipos de redes privadas virtuales (VPN) como, por ejemplo, IPsec, autentican los puntos finales de una conexión TCP/IP, como alternativa a utilizar SSL. VPN cifra cada paquete IP que se transmite por la red. Una vez establecida la conexión VPN, se ha establecido la red de confianza. Puede conectar clientes MQTT a los canales de telemetría utilizando TCP/IP en la red VPN.

La autenticación de cliente mediante TLS depende de que el cliente tenga un secreto. Este dato secreto es la clave privada del cliente en el caso de un certificado autofirmado o una clave que proporciona una entidad emisora de certificados. La clave se utiliza para firmar el certificado digital del cliente. Todas las personas que tengan la correspondiente clave pública pueden verificar el certificado digital. Los certificados pueden ser de confianza o, si están en cadena, se les puede realizar un seguimiento a través de una cadena de certificados a un certificado raíz de confianza. La verificación de clientes envía todos los certificados de la cadena de certificados que proporciona el cliente al servidor. El servidor comprueba la cadena de certificados hasta que encuentra un certificado en el que confía. El certificado de confianza puede ser un certificado público generado a partir de un certificado autofirmado, o un certificado raíz que normalmente ha emitido una entidad emisora de certificados. Un último paso, que es opcional, es la comparación del certificado de confianza con una lista de revocación de certificados "activa".

El certificado de confianza puede haberlo emitido una entidad emisora de certificados y puede estar incluido en el almacén de certificados JRE. Puede ser un certificado autofirmado o cualquiera que se haya añadido al almacén de claves del canal de telemetría como un certificado de confianza.

Nota: El canal de telemetría tiene un almacén de claves/almacén de confianza combinado que contiene tanto las claves privadas de uno o varios canales de telemetría, y todos los certificados públicos necesarios para autenticar clientes. Puesto que un canal SSL debe tener un almacén de claves, y es el mismo archivo que el almacén de confianza, nunca se hace referencia al almacén de certificados JRE. La implicación es que si la autenticación de un cliente requiere un certificado raíz de CA, debe colocar el certificado raíz en el almacén de claves del canal, aunque el certificado raíz de CA ya esté en el almacén de certificados JRE. Nunca se hace referencia alguna al almacén de certificados JRE.

Considere las amenazas a las que la autenticación del cliente pretende hacer frente y qué papel juega el cliente y el servidor en esa situación. La autenticación del certificado de cliente sola no es suficiente para evitar accesos no autorizados al sistema. Si otra persona ha tomado posesión del dispositivo del cliente, éste no tiene necesariamente que estar actuando con la autoridad del titular del certificado. No confíe nunca en una única defensa frente a ataques no deseados. Utilice al menos dos factores para la autenticación, además de la posesión complementaria de un certificado con conocimiento de información privada. Por ejemplo, utilice JAAS y autentique el cliente utilizando una contraseña que emita el servidor.

La principal amenaza a la que se enfrenta un certificado de cliente es que caiga en las manos equivocadas. El certificado se encuentra en un almacén de claves protegido con contraseña en el cliente. ¿Cómo se coloca en el almacén de claves? ¿Cómo consigue el cliente MQTT la contraseña para el almacén de claves? ¿Qué nivel de seguridad tiene la protección con contraseña? Los dispositivos de telemetría a menudo son fáciles de eliminar y pueden ser pirateados en privado. ¿Debe el hardware del dispositivo

estar protegido contra manipulación no autorizada? La distribución y protección de certificados del lado del cliente se reconoce como difícil; se denomina el problema de la gestión de claves.

Una amenaza secundaria es que el dispositivo se utilice de forma incorrecta para acceder a servidores de forma imprevista. Por ejemplo, si la aplicación MQTT está amenazada, es posible utilizar algún punto débil de la configuración del servidor mediante la identidad de cliente autenticada.

Para autenticar un cliente MQTT mediante SSL, configure el canal de telemetría y el cliente.

Conceptos relacionados

“Configuración de canal de telemetría para la autenticación de cliente MQTT mediante TLS” en la página 356

ElIBM MQ El administrador configura los canales de telemetría en el servidor. Cada canal se configura para aceptar una conexión TCP/IP en un número de puerto diferente. Los canales TLS se configuran con un acceso protegido con frase de contraseña a archivos de claves. Si un canal TLS está definido sin frase de contraseña o archivo de claves, el canal no acepta conexiones SSL.

Configuración de cliente MQTT para la autenticación de cliente mediante TLS

Windows Linux AIX *Configuración de canal de telemetría para la autenticación de cliente MQTT mediante TLS*

ElIBM MQ El administrador configura los canales de telemetría en el servidor. Cada canal se configura para aceptar una conexión TCP/IP en un número de puerto diferente. Los canales TLS se configuran con un acceso protegido con frase de contraseña a archivos de claves. Si un canal TLS está definido sin frase de contraseña o archivo de claves, el canal no acepta conexiones SSL.

Establezca la propiedad `com.ibm.mq.MQTT.ClientAuth` de un canal de telemetría TLS en `REQUIRED` para obligar a todos los clientes que se conectan en dicho canal a proporcionar una prueba de que tienen certificados digitales verificados. Los certificados de cliente se autentican utilizando certificados de las entidades emisoras de certificados, dirigidos a un certificado raíz de confianza. Si el certificado de cliente es autofirmado o está firmado por un certificado que proviene de una entidad emisora de certificados, los certificados firmados públicamente del cliente, o autoridad de certificados, deben almacenarse de forma segura en el servidor.

Coloque el certificado de cliente firmado públicamente o el certificado de la entidad emisora de certificados en el almacén de claves del canal de telemetría. En el servidor, los certificados firmados públicamente se almacenan en el mismo archivo de claves que los certificados firmados en privado, en vez de en un almacén de confianza aparte.

El servidor verifica la firma de los certificados de cliente que se envía utilizando todos los certificados públicos y paquetes de cifrado que tenga. El servidor verifica la cadena de claves. El gestor de colas puede configurarse para probar el certificado respecto a la lista de revocación de certificados. La propiedad de lista de nombres de revocación de gestor de colas es `SSLCRLNL`.

Si alguno de los certificados que un cliente envía lo verifica un certificado del almacén de claves del servidor, el cliente se autentica.

El administrador de IBM MQ puede configurar el mismo canal de telemetría para utilizar JAAS en la comprobación del valor `Username` o `ClientIdentifier` del cliente con el valor `Password` del cliente.

Puede utilizar el mismo almacén de claves para varios canales de telemetría.

La verificación de al menos un certificado digital en el almacén de claves de cliente protegido con contraseña en el dispositivo autentica al cliente en el servidor. El certificado digital sólo se utiliza para la autenticación mediante IBM MQ. No se utiliza para verificar la dirección TCP/IP del cliente, o establecer la identidad del cliente para la autorización o la contabilidad. La identidad del cliente adoptada por el servidor es `Username` o `ClientIdentifier` del cliente, o una identidad creada por el administrador de IBM MQ.

También puede utilizar las suites de cifrado TLS para la autenticación de cliente. Si piensa utilizar suites de cifrado SHA-2, consulte “Requisitos del sistema para utilizar las suites de cifrado SHA-2 con canales de MQTT” en la página 361.

Conceptos relacionados

“Configuración del canal de telemetría para la autenticación de canal mediante TLS” en la página 357
ElIBM MQ El administrador configura los canales de telemetría en el servidor. Cada canal se configura para aceptar una conexión TCP/IP en un número de puerto diferente. Los canales TLS se configuran con un acceso protegido con frase de contraseña a archivos de claves. Si un canal TLS está definido sin frase de contraseña o archivo de claves, el canal no acepta conexiones SSL.

CipherSpecs y CipherSuites

Referencia relacionada

DEFINIR CANAL (MQTT)

ALTER CHANNEL (MQTT)

Windows Linux AIX Autenticación de canal de telemetría mediante TLS

Las conexiones entre el cliente MQTT y el gestor de colas las inicia siempre el cliente MQTT. El cliente MQTT es siempre el cliente SSL. Tanto la autenticación de cliente del servidor como la autenticación de servidor del cliente MQTT son opcionales.

El cliente intenta siempre autenticar el servidor, a menos que esté configurado para utilizar una CipherSpec que dé soporte a la conexión anónima. Si la autenticación falla, la conexión no se establece.

Algunos tipos de redes privadas virtuales (VPN) como, por ejemplo, IPsec, autentican los puntos finales de una conexión TCP/IP, como alternativa a utilizar SSL. VPN cifra cada paquete IP que se transmite por la red. Una vez establecida la conexión VPN, se ha establecido la red de confianza. Puede conectar clientes MQTT a los canales de telemetría utilizando TCP/IP en la red VPN.

La autenticación de servidor mediante SSL autentica el servidor al que va a enviar información confidencial. El cliente realiza las comprobaciones que hacen coincidir los certificados enviados desde el servidor con los certificados colocados en su almacén de confianza o en su JRE.`cacerts` almacenar.

El almacén de certificados JRE es un archivo JKS,`cacerts` . Está localizado enJRE `InstallPath\lib\security\` . Se instala con la contraseña predeterminada `changeit`. Puede almacenar los certificados en que confíe en el almacén de certificados de JRE, o en el almacén de confianza del cliente. No puede utilizar ambos almacenes. Utilice el almacén de confianza del cliente si desea mantener los certificados públicos en los que confía el cliente separados de otros certificados.Java uso de aplicaciones. Utilice el almacén de certificados JRE si desea utilizar un almacén de certificados común para todosJava aplicaciones que se ejecutan en el cliente. Si decide utilizar el almacén de certificados de JRE, revise los certificados que contenga, para asegurarse de que confía en ellos.

Puede modificar la configuración JSSE indicando un proveedor de confianza diferente. Puede personalizar un proveedor de confianza para que realice diferentes comprobaciones en un certificado. En algunos entornos OGSi que han utilizado el cliente MQTT, el entorno proporciona un proveedor de confianza diferente.

Para autenticar el canal de telemetría mediante TLS, configure el servidor y el cliente.

Windows Linux AIX Configuración del canal de telemetría para la autenticación de canal mediante TLS

ElIBM MQ El administrador configura los canales de telemetría en el servidor. Cada canal se configura para aceptar una conexión TCP/IP en un número de puerto diferente. Los canales TLS se configuran con un acceso protegido con frase de contraseña a archivos de claves. Si un canal TLS está definido sin frase de contraseña o archivo de claves, el canal no acepta conexiones SSL.

Almacene el certificado digital en el servidor, firmado con su clave privada, en el almacén de claves que el canal de telemetría vaya a utilizar en el servidor. Almacene cualquier certificado de su cadena de claves en el almacén de claves, si desea transmitir la cadena de claves al cliente. Configure el canal de telemetría mediante IBM MQ Explorer para utilizar TLS. Proporcione la vía de acceso al almacén de claves

y la frase de contraseña para acceder al mismo. Si no establece el número de puerto TCP/IP del canal, el número de puerto del canal de telemetría TLS toma, como valor predeterminado, el 8883.

También puede utilizar las suites de cifrado TLS para la autenticación de canal. Si piensa utilizar suites de cifrado SHA-2, consulte “Requisitos del sistema para utilizar las suites de cifrado SHA-2 con canales de MQTT” en la página 361.

Importante: V 9.4.0 V 9.4.0 A partir de IBM MQ 9.4.0, los repositorios de claves CMS y los archivos de ocultación no están soportados con los canales AMQP y MQTT que utilizan SSL/TLS. Utilice los repositorios de claves PKCS #12 y proteja las contraseñas del repositorio de claves utilizando en su lugar el sistema de protección de contraseñas IBM MQ . Puede crear un repositorio de claves PKCS #12 utilizando el mandato siguiente:

```
runmqakm -keydb -create -db filename.p12 -pw password -type pkcs12
```

Este mandato crea un archivo de repositorio de claves PKCS #12 denominado *filename*.p12 que está protegido con la contraseña especificada.

Conceptos relacionados

“Configuración de canal de telemetría para la autenticación de cliente MQTT mediante TLS” en la página 356

ElIBM MQ El administrador configura los canales de telemetría en el servidor. Cada canal se configura para aceptar una conexión TCP/IP en un número de puerto diferente. Los canales TLS se configuran con un acceso protegido con frase de contraseña a archivos de claves. Si un canal TLS está definido sin frase de contraseña o archivo de claves, el canal no acepta conexiones SSL.

CipherSpecs y CipherSuites

Referencia relacionada

DEFINE CHANNEL (MQTT)

ALTER CHANNEL (MQTT)

Ejemplo de configuración de canal MQTT utilizando la autenticación TLS

Este ejemplo le guía a través de un ejemplo de configuración de un canal MQTT que utiliza la autenticación TLS.

El ejemplo configura un canal entre MQTT y Mosquitto.

El ejemplo utiliza un contenedor Docker para IBM MQ en Red Hat Enterprise Linux y Mosquitto en CentOS, pero se aplica a cualquier tipo de servidor. (CentOS se ha utilizado para Mosquitto debido a titularidades de registro.)

Configurar el almacén de claves y el canal de IBM MQ para TLS unidireccional

Importante: V 9.4.0 V 9.4.0 A partir de IBM MQ 9.4.0, los repositorios de claves CMS y los archivos de ocultación no están soportados con los canales AMQP y MQTT que utilizan SSL/TLS. Utilice los repositorios de claves PKCS #12 y proteja las contraseñas del repositorio de claves utilizando en su lugar el sistema de protección de contraseñas IBM MQ .

Realice los pasos siguientes:

1. V 9.4.0 V 9.4.0 Cree el almacén de claves de IBM MQ :

   ```
   runmqakm -keydb -create -db mqtt.p12 -pw "passw0rd" -type p12
   ```

2. V 9.4.0 V 9.4.0 Cree un certificado personal:

   ```
   runmqakm -cert -create -db mqtt.p12 -pw "passw0rd" -size 2048 -dn "CN= mqm, OU=MQTest,
   O=MQSuppor, C=US" -sig_alg SHA256_WITH_RSA -label ibmwebspheremqmqm
   ```

   Puede utilizar el mandato siguiente para confirmar la creación del certificado:

```
runmqakm -cert -list -v -db mqtt.p12 -pw "passw0rd"
```

3. V 9.4.0 V 9.4.0 Cree el canal MQTT especificando el mandato siguiente en el indicador de runmqsc:

```
DEFINE CHANNEL(MQTTDEMO) CHLTYPE(MQTT) BACKLOG(4096) PORT(8883) MCAUSER('mqm')
PROTOCOL(MQTTV311,MQTTV3,HTTP) SSLCAUTH(OPTIONAL) SSLCIPH('SSL_RSA_WITH_AES_256_CBC_SHA256')
SSLKEYR('/var/mqm/mqtt/mqtt.p12') SSLKEYP('passw0rd') TRPTYPE(TCP)
```

   Tenga en cuenta que el canal utiliza correlaciones de cifrado Java , consulte TLS CipherSpecs y CipherSuites en IBM MQ classes for JMS.

4. Extraiga el certificado:

```
runmqakm -cert -extract -db mqtt.kdb -stashed -label ibmwebspheremqmqm -target serverCert.pem
```

Instalar Mosquitto en CentOS en un contenedor Docker

Realice los pasos siguientes para crear un contenedor Docker con Mosquitto en ejecución en CentOS:

1. `docker pull centos`
2. `docker run -it centos /bin/bash`
3. `yum -y install epel-release`
4. `yum -y install mosquitto`

Mover certificado de firmante a Mosquitto

Realice los pasos siguientes para mover el certificado que ha creado en IBM MQ a Mosquitto. Estos pasos se ejecutan en la máquina host de Docker .

1. Vea los ID de contenedor en Docker:

```
docker container ls
```

2. Copie el archivo del contenedor de docker en el docker del sistema local

```
cp MQ_Container_ID:/var/mqm/mqtt/serverCert.pem serverCert.pem
```

3. Copie el archivo de la máquina local en el directorio raíz de la máquina centOS :

```
docker cp serverCert.pem CentOS_ContainerID:/serverCert.pem
```

Publicar con Mosquitto

Publique un mensaje de prueba en Mosquitto utilizando el mandato siguiente:

```
mosquitto_pub -h 172.17.0.2 --cafile serverCert.pem --insecure -p 8883 -i mosquittoClient -t
test -m 'test message' -d
```

Los argumentos del mandato tienen los significados siguientes:

-h
La dirección IP del host de Red Hat Enterprise Linux (se puede encontrar utilizando **nslookup**).

-- cafile
El archivo que contiene el certificado de firmante.

-- inseguro
Esta opción se especifica porque el ejemplo utiliza un certificado autofirmado. No utilice esta opción cuando utilice certificados CA reales.

-p
Número de puerto.

-i
ID de cliente.

-t
El tema en el que se está publicando.

-m
El mensaje que se está publicando.

-d
Habilitar mensajes de depuración.

Configurar el canal MQTT para la autenticación TLS mutua

Especifique el mandato siguiente para volver a configurar el canal MQTT como SSLCAUTH (REQUIRED).

```
ALTER CHANNEL(MQTTDEMO) CHLTYPE(MQTT) SSLCAUTH(REQUIRED)
```

Cree un par de clave/certificado en el servidor Mosquitto y añádalo a IBM MQ

Especifique los mandatos siguientes para crear el par clave/certificado en Mosquitto:

1. Utilice **openSSL** para crear el par clave/certificado para Mosquitto:

   ```
   openssl req -x509 -newkey rsa:4096 -keyout mosquittoKey.pem -out mosquittoCert.pem -subj "/
   CN=Mosquitto"
   ```

2. Liste los ID de contenedor para los contenedores:

   ```
   docker container ls
   ```

3. Copie el certificado Mosquitto en el docker del sistema local:

   ```
   docker cp CentOS_ContainerID:mosquittoCert.pem .
   ```

4. Copie el certificado Mosquitto en IBM MQ:

   ```
   docker cp mosquittoCert.pem MQ_Container_ID:/var/mqm/mqtt
   ```

5. Añada el certificado al almacén de claves de IBM MQ :

   ```
   runmqakm -cert -add -db mqtt.kdb -stashed -file mosquittoCert.pem
   ```

6. Reinicie el canal MQTT .

Publicar con Mosquitto y autenticación mutua

Realice los pasos siguientes para publicar con Mosquitto utilizando la autenticación mutua.

1. El mandato siguiente debe publicar correctamente un mensaje de prueba:

   ```
   mosquitto_pub -h 172.17.0.2 --cafile serverCert.pem --insecure -p 8883 -i mosquittoClient -t
   test -m 'test message' -d --cert mosquittoCert.pem --key mosquittoKey.pem
   ```

2. El mandato siguiente no puede publicar un mensaje de prueba y generar un mensaje de error porque no envía un certificado personal desde Mosquitto:

   ```
   mosquitto_pub -h 172.17.0.2 --cafile serverCert.pem --insecure -p 8883 -i mosquittoClient -t
   test -m 'test message' -d /var/mqm/qmgrs/mqttDemoQM/errors/ mqxr_0.log
   ```

Información relacionada
Gestión de claves y certificados

Windows Linux AIX Requisitos del sistema para utilizar las suites de cifrado SHA-2 con canales de MQTT

Si utiliza una versión de Java que sea compatible con las suites de cifrado SHA-2, puede utilizar estas suites para proteger los canales de MQTT (telemetría) y las aplicaciones cliente.

Para IBM MQ 8.0, que incluye el servicio de telemetría (MQXR), la versión mínima de Java es Java 7 de IBM, SR6. Las suites de cifrado SHA-2 están soportadas de forma predeterminada en Java 7 de IBM, SR4 y versiones posteriores. Por lo tanto, puede utilizar las suites de cifrado SHA-2 con el servicio de telemetría (MQXR) para proteger sus canales MQTT (telemetría).

Si está ejecutando un cliente de MQTT con un JRE diferente, asegúrese de que también sea compatible con las suites de cifrado SHA-2.

Conceptos relacionados

Servicio de telemetría (MQXR)

"Configuración del canal de telemetría para la autenticación de canal mediante TLS" en la página 357

ElIBM MQ El administrador configura los canales de telemetría en el servidor. Cada canal se configura para aceptar una conexión TCP/IP en un número de puerto diferente. Los canales TLS se configuran con un acceso protegido con frase de contraseña a archivos de claves. Si un canal TLS está definido sin frase de contraseña o archivo de claves, el canal no acepta conexiones SSL.

Referencia relacionada

DEFINIR CANAL (MQTT)

ALTER CHANNEL (MQTT)

Windows Linux AIX Privacidad de las publicaciones en los canales de telemetría

La privacidad de las publicaciones MQTT enviadas en cualquier dirección en los canales de telemetría está protegida mediante TLS, para cifrar las transmisiones a través de la conexión.

Los clientes MQTT que se conectan a los canales de telemetría utilizan TLS para proteger la privacidad de las publicaciones transmitidas en el canal, mediante la criptografía de claves simétricas. Puesto que los puntos finales no se autentican, no se puede confiar en un canal de cifrado solo. Combine la protección de privacidad con la autenticación del servidor o mutua.

Algunos tipos de redes privadas virtuales (VPN) como, por ejemplo, IPsec, autentican los puntos finales de una conexión TCP/IP, como alternativa a utilizar SSL. VPN cifra cada paquete IP que se transmite por la red. Una vez establecida la conexión VPN, se ha establecido la red de confianza. Puede conectar clientes MQTT a los canales de telemetría utilizando TCP/IP en la red VPN.

Para obtener información sobre una configuración típica, que cifre el canal y autentique el servidor, consulte "Autenticación de canal de telemetría mediante TLS" en la página 357.

El cifrado de las conexiones TLS sin autenticar el servidor expone la conexión frente a ataques de terceros. Aunque la información que se intercambia está protegida contra escuchas no autorizadas, no sabe con quién se está intercambiando. A menos que controle la red, estará expuesto a que alguien intercepte las transmisiones IP, haciéndose pasar por el punto final.

Puede crear una conexión TLS cifrada, sin autenticar el servidor, mediante un intercambio de claves Diffie-Hellman CipherSpec que dé soporte a TLS anónima. El secreto maestro, compartido entre el cliente y el servidor, y que se utiliza para cifrar transmisiones TLS, se establece sin intercambiar ningún certificado de servidor firmado en privado.

Puesto que las conexiones anónimas son inseguras, la mayoría de las implementaciones TLS no toman como valor predeterminado la utilización de las CipherSpecs anónimas. Si un canal de telemetría acepta una conexión TLS que solicita un cliente, el canal deberá tener un almacén de claves protegido mediante una frase de contraseña. De forma predeterminada, puesto que las implementaciones TLS no utilizan las CipherSpecs anónimas, el almacén de claves debe contener un certificado de firma privada que demuestre que el cliente puede autenticarse.

Si utiliza las CipherSpecs anónima, el almacén de claves del servidor debe existir, pero no es necesario que contenga ningún certificado firmado en privado.

Otra forma de establecer una conexión cifrada es sustituir el proveedor de confianza en el cliente por su propia implementación. El proveedor de confianza no ha podría autenticar el certificado de servidor, pero la conexión se cifraría.

Atención: Al utilizar TLS con MQTT puede utilizar mensajes grandes, sin embargo, puede haber un posible impacto en el rendimiento al hacerlo. MQTT está optimizado para procesar mensajes pequeños (normalmente entre 1KB y 1MB de tamaño).

Windows Linux AIX Configuración TLS de clientes y canales de telemetría de MQTT Java

Configure TLS para autenticar el canal de telemetría y el cliente de MQTT Java y cifrar la transferencia de mensajes entre ellos. MQTT Los clientes de Java utilizan JSSE (Java Secure Socket Extension) para conectar canales de telemetría utilizando TLS. Algunos tipos de redes privadas virtuales (VPN) como, por ejemplo, IPsec, autentican los puntos finales de una conexión TCP/IP, como alternativa a utilizar SSL. VPN cifra cada paquete IP que se transmite por la red. Una vez establecida la conexión VPN, se ha establecido la red de confianza. Puede conectar clientes MQTT a los canales de telemetría utilizando TCP/IP en la red VPN.

Puede configurar la conexión entre un cliente Java MQTT y un canal de telemetría para utilizar el protocolo TLS sobre TCP/IP. Lo que se asegura depende de cómo configure TLS para que utilice JSSE. Empezando con la configuración más segura, puede configurar tres niveles de seguridad diferentes:

1. Permita que se conecten sólo los clientes MQTT de confianza. Conecte un cliente MQTT sólo a un canal de telemetría de confianza. Cifre los mensajes entre el cliente y el gestor de colas; consulte “Autenticación de cliente MQTT mediante TLS” en la página 355
2. Conecte un cliente MQTT sólo a un canal de telemetría de confianza. Cifre los mensajes entre el cliente y el gestor de colas; consulte “Autenticación de canal de telemetría mediante TLS” en la página 357.
3. Cifre los mensajes entre el cliente y el gestor de colas; consulte “Privacidad de las publicaciones en los canales de telemetría” en la página 361.

Parámetros de configuración JSSE

Modifique los parámetros JSSE para cambiar la forma en la que se configura una conexión TLS. Los parámetros de configuración de JSSE se organizan en tres conjuntos:

1. Canal de MQ Telemetry
2. Cliente deMQTT Java
3. JRE

Configure los parámetros del canal de telemetría mediante IBM MQ Explorer. Establezca los parámetros de cliente de MQTT Java en el atributo `MqttConnectionOptions.SSLProperties` . Modifique los parámetros de seguridad JRE editando los archivos en el directorio de seguridad JRE en el cliente y en el servidor.

Canal de MQ Telemetry

Establezca todos los parámetros TLS del canal de telemetría utilizando IBM MQ Explorer.

`ChannelName`

`ChannelName` es un parámetro necesario en todos los canales.

El nombre del canal identifica el canal asociado a un número de puerto particular. Asignar un nombre a los canales le ayudará a administrar conjuntos de clientes MQTT.

PortNumber

PortNumber es un parámetro opcional en todos los canales. El valor predeterminado es 1883 para canales TCP, y 8883 para los canales TLS.

El número de puerto TCP/IP asociado a este canal. Los clientes MQTT se conectan a un canal especificando el puerto definido para el canal. Si el canal tiene propiedades TLS, el cliente debe conectarse utilizando el protocolo TLS; por ejemplo:

```
MQTTClient mqttClient = new MqttClient( "ssl://www.example.org:8884", "clientId1");
mqttClient.connect();
```

KeyFileName

KeyFileName es un parámetro necesario para los canales TLS. Con los canales TCP, debe omitirse.

KeyFileName es la vía de acceso al almacén de claves Java que contiene los certificados digitales que proporciona el usuario. Utilice JKS, JCEKS o PKCS12 como el tipo de almacén de claves en el servidor.

Identifique el tipo de almacén de claves utilizando una de las extensiones de archivo siguientes:

```
.jks
.jceks
.p12
.pkcs12
```

Un almacén de claves que tenga cualquier otra extensión de archivo se supone que es un almacén de claves JKS.

Puede combinar un tipo de almacén de claves en el servidor con otros tipos de almacén de claves en el cliente.

Coloque el certificado privado del servidor en el almacén de claves. El certificado se conoce como el certificado del servidor. El certificado puede estar autofirmado o formar parte de una cadena de certificados firmado por una autoridad de firmas.

Si está utilizando una cadena de certificados, coloque los certificados asociados en el almacén de claves del servidor.

El certificado del servidor y cualquier certificado de su cadena de certificados se envían a los clientes para autenticar la identidad del servidor.

Si ha establecido ClientAuth en Required, el almacén de claves debe contener los certificados necesarios para autenticar el cliente. El cliente envía un certificado autofirmado, o una cadena de certificados, y el cliente se autentica mediante la primera verificación de este material contra un certificado del almacén de claves. Con el uso de una cadena de certificados, un certificado puede verificar muchos clientes, incluso si se emiten con certificados de cliente distintos.

PassPhrase

PassPhrase es un parámetro necesario para los canales TLS. Con los canales TCP, debe omitirse.

La frase de contraseña se utiliza para proteger el almacén de claves.

ClientAuth

ClientAuth es un parámetro TLS opcional. Toma como valor predeterminado que no se autentique ningún cliente. Con los canales TCP, debe omitirse.

Establezca ClientAuth si desea que el servicio de telemetría (MQXR) autentique el cliente, antes de permitir que el cliente se conecte al canal de telemetría.

Si establece ClientAuth, el cliente debe conectarse al servidor utilizando TLS y autenticar el servidor. Como respuesta a la definición de ClientAuth, el cliente envía su certificado digital al servidor y cualquier otro certificado de su almacén de claves. El certificado digital se conoce

como el certificado de cliente. Estos certificados se autentican con los certificados retenidos en el almacén de claves del canal y en el almacén de JRE `cacerts`.

`CipherSuite`

`CipherSuite` es un parámetro TLS opcional. El valor predeterminado es intentar todas las CipherSpecs habilitadas. Con los canales TCP, debe omitirse.

Si desea utilizar una CipherSpec particular, establezca `CipherSuite` en el mismo nombre que la CipherSpec que debe utilizarse para establecer una conexión TLS.

El servicio de telemetría y el cliente MQTT negocian una CipherSpec común entre todas las CipherSpecs que están habilitadas en cada extremo. Si se especifica una CipherSpec determinada en uno de los extremos de la conexión, o en ambos, debe coincidir con la CipherSpec del otro extremo.

Instale otros cifrados añadiendo otros proveedores adicionales a JSSE.

`Federal Information Processing Standards (FIPS)`

FIPS es un parámetro opcional. De forma predeterminada, no está definido.

Bien en el panel de propiedades del gestor de colas, o bien mediante **`runmqsc`**, establezca SSLFIPS. SSLFIPS especifica si sólo se deben utilizar algoritmos certificados por FIPS.

`Revocation namelist`

Revocation namelist es un valor opcional. De forma predeterminada, no está definido.

Bien en el panel de propiedades del gestor de colas, o bien mediante **`runmqsc`**, establezca SSLCRLNL. SSLCRLNL especifica una lista de nombres de objetos de información de autenticación que se utilizan para proporcionar ubicaciones de revocación de certificados.

No se utiliza ningún otro parámetro de gestor de colas que defina propiedades TLS.

Cliente de MQTT Java

Establezca propiedades TLS para el cliente Java en `MqttConnectionOptions.SSLProperties`; por ejemplo:

```
java.util.Properties sslClientProperties = new Properties();
sslClientProperties.setProperty("com.ibm.ssl.keyStoreType", "JKS");
com.ibm.micro.client.mqttv3.MqttConnectOptions conOptions = new MqttConnectOptions();
conOptions.setSSLProperties(sslClientProperties);
```

Los nombres y valores de propiedades específicas se describen en la clase `MqttConnectOptions`. Para obtener enlaces a la documentación de la API de cliente para las bibliotecas de cliente MQTT, consulte Referencia de programación de cliente MQTT.

`Protocol`

`Protocol` es opcional.

El protocolo se selecciona bajo negociación con el servidor de telemetría. Si necesita un protocolo específico, puede seleccionar uno. Si el servidor de telemetría no da soporte al protocolo, la conexión falla.

`ContextProvider`

`ContextProvider` es opcional.

`KeyStore`

`KeyStore` es opcional. Configúrelo si ha establecido `ClientAuth` en el servidor para forzar la autenticación del cliente.

Coloque el certificado digital del cliente, firmado mediante su clave privada, en el almacén de claves. Especifique la vía de acceso del almacén de claves y la contraseña. El tipo y el proveedor son opcionales. JKS es el tipo predeterminado, e IBMJCE el proveedor predeterminado.

Especifique un proveedor de almacén de claves diferente que haga referencia a una clase que añada un nuevo proveedor de almacén de claves. Pase el nombre del algoritmo utilizado por el proveedor del almacén de claves para crear una instancia de `KeyManagerFactory` estableciendo el nombre del gestor de claves.

`TrustStore`

`TrustStore` es opcional. Puede colocar todos los certificados de confianza en el almacén de JRE`cacerts`.

Configure el almacén de confianza si desea tener almacén de confianza diferente para el cliente. Es posible que no tenga que configurar el almacén de confianza si el servidor está utilizando un certificado emitido por una CA bien conocida que ya tiene su certificado raíz almacenado en `cacerts`.

Añada el certificado firmado públicamente del servidor o el certificado raíz al almacén de confianza, y especifique la vía de acceso de almacén de confianza y la contraseña. JKS es el tipo predeterminado, e IBMJCE el proveedor predeterminado.

Especifique un proveedor de almacén de confianza diferente que haga referencia a una clase que añada un nuevo proveedor de almacén de confianza. Pase el nombre del algoritmo utilizado por el proveedor del almacén de confianza para crear una instancia de `TrustManagerFactory` estableciendo el nombre del gestor de confianza.

JRE

En el JRE se configuran otros aspectos sobre la seguridad Java que afectan al comportamiento del TLS en el cliente y en el servidor. Los archivos de configuración en Windows están en *`Java Installation Directory`*`\jre\lib\security`. Si va a utilizar el JRE que se proporciona con IBM MQ la vía de acceso es la que se indica en la tabla siguiente:

Tabla 18. Vías de acceso del archivo por plataforma para los archivos de configuración de JRE TLS

Plataforma	**Vía de acceso de archivo**
Windows	*`WMQ Installation Directory`*`\java\jre\lib\security`
Plataformas AIX and Linux	*`WMQ Installation Directory`*`/java/jre64/jre/lib/security`

Entidades emisoras de certificados conocidas

El archivo `cacerts` contiene los certificados raíz de autoridades de certificación conocidas. `cacerts` se utiliza de forma predeterminada, a menos que especifique un almacén de confianza. Si utiliza el almacén de `cacerts` o no proporciona un almacén de confianza, debe revisar y editar la lista de firmantes en `cacerts` para cumplir los requisitos de seguridad.

V 9.4.0 V 9.4.0 Puede utilizar el mandato **`runmqktool`** para gestionar el archivo de certificados `cacerts` . `cacerts` es un archivo JKS. Especifique el parámetro **`-storetype jks`** cuando se utilice el mandato **`runmqktool`** para gestionar el archivo de certificados.

V 9.4.0 V 9.4.0 La contraseña predeterminada para el archivo `cacerts` es `changeit`. Cambie la contraseña utilizando el mandato **`runmqktool -storepasswd`** para proteger el archivo.

Configuración de clases de seguridad

Utilice el archivo `java.security` para registrar proveedores de seguridad adicionales y otras propiedades de seguridad predeterminadas.

Permisos

Utilice el archivo `java.policy` para modificar los permisos otorgados a los recursos. `javaws.policy` otorga permisos a `javaws.jar`

Fuerza de cifrado

Algunos JRE se entregan con poca fuerza de cifrado. Si no puede importar claves a los almacenes de claves, la poca fuerza de cifrado puede ser la causa. Si es necesario, descargue los archivos de jurisdicción fuertes, pero limitados, de los kits de desarrollador deIBM , Información de seguridad.

Importante: Es posible que su país tenga restricciones sobre la importación, posesión, utilización y nueva exportación a otro país de software cifrado. Antes de descargar o utilizar archivos de políticas sin restricciones, debe comprobar las leyes existentes en su país. Compruebe sus regulaciones y políticas sobre importación, posesión, utilización y nueva exportación de software cifrado para ver si están permitidas estas acciones.

Modificación del proveedor de confianza para permitir al cliente para conectarse a cualquier servidor

En el siguiente ejemplo se muestra cómo añadir un proveedor de confianza y cómo hacer referencia al mismo desde el código de cliente MQTT. En el ejemplo no se autentican el cliente ni el servidor. La conexión TLS resultante se cifra sin que se haya autenticado.

El fragmento de código de la Figura 16 en la página 366 establece el proveedor de confianza `AcceptAllProviders` y el gestor de confianza para el cliente MQTT.

```
java.security.Security.addProvider(new AcceptAllProvider());
java.util.Properties sslClientProperties = new Properties();
sslClientProperties.setProperty("com.ibm.ssl.trustManager","TrustAllCertificates");
sslClientProperties.setProperty("com.ibm.ssl.trustStoreProvider","AcceptAllProvider");
conOptions.setSSLProperties(sslClientProperties);
```

Figura 16. Fragmento de código de Cliente MQTT

```
package com.ibm.mq.id;
public class AcceptAllProvider extends java.security.Provider {
private static final long serialVersionUID = 1L;
public AcceptAllProvider() {
super("AcceptAllProvider", 1.0, "Trust all X509 certificates");
put("TrustManagerFactory.TrustAllCertificates",
AcceptAllTrustManagerFactory.class.getName());
}
```

Figura 17. AcceptAllProvider.java

```
protected static class AcceptAllTrustManagerFactory extends
javax.net.ssl.TrustManagerFactorySpi {
public AcceptAllTrustManagerFactory() {}
protected void engineInit(java.security.KeyStore keystore) {}
protected void engineInit(
javax.net.ssl.ManagerFactoryParameters parameters) {}
protected javax.net.ssl.TrustManager[] engineGetTrustManagers() {
return new javax.net.ssl.TrustManager[] { new AcceptAllX509TrustManager() };
}
```

Figura 18. AcceptAllTrustManagerFactory.java

```
protected static class AcceptAllX509TrustManager implements
javax.net.ssl.X509TrustManager {
public void checkClientTrusted(
java.security.cert.X509Certificate[] certificateChain,
String authType) throws java.security.cert.CertificateException {
report("Client authtype=" + authType);
for (java.security.cert.X509Certificate certificate : certificateChain) {
report("Accepting:" + certificate);
}
}
public void checkServerTrusted(
java.security.cert.X509Certificate[] certificateChain,
String authType) throws java.security.cert.CertificateException {
report("Server authtype=" + authType);
for (java.security.cert.X509Certificate certificate : certificateChain) {
report("Accepting:" + certificate);
}
}
public java.security.cert.X509Certificate[] getAcceptedIssuers() {
return new java.security.cert.X509Certificate[0];
}
private static void report(String string) {
System.out.println(string);
}
}
```

Figura 19. AcceptAllX509TrustManager.java

Windows Linux AIX Configuración JAAS del canal de telemetría

Configure JAAS para autenticar el valor de `Username` que envía el cliente.

El administrador de IBM MQ configura qué canales MQTT necesitan autenticación de cliente mediante JAAS. Especifique el nombre de una configuración JAAS para cada canal que vaya a realizar la autenticación JAAS. Los canales pueden utilizar todos la misma configuración JAAS o utilizar configuraciones JAAS diferentes. Las configuraciones se definen en *WMQData directory*`\qmgrs\`*qMgrName*`\mqxr\jaas.config`.

El archivo `jaas.config` está organizado por el nombre de configuración JAAS. Bajo cada nombre de configuración aparece una lista de las configuraciones de inicio de sesión; consulte la “Ejemplo de archivo jaas.config” en la página 368.

JAAS proporciona cuatro módulos de inicio de sesión estándar. Los módulos de inicio de sesión estándar de NT y UNIX son de valor limitado.

JndiLoginModule
Autentica con un servicio de directorio configurado en JNDI (Java Naming and Directory Interface).

Krb5LoginModule
Autentica utilizando protocolos Kerberos.

NTLoginModule
Autentica utilizando la información de seguridad de NT para el usuario actual.

UnixLoginModule
Autentica utilizando la información de seguridad de UNIX para el usuario actual.

El problema al utilizar `NTLoginModule` o `UnixLoginModule` es que el servicio de telemetría (MQXR) se ejecuta con la identidad `mqm`, y no con la identidad del canal MQTT. `mqm` es la identidad que se pasa a `NTLoginModule` o a `UnixLoginModule` para la autenticación, y no la identidad del cliente.

Para solucionar este problema, escriba un módulo de inicio de sesión propio o utilice otros módulos de inicio de sesión estándar. Se proporciona un `JAASLoginModule.java` de ejemplo con MQ Telemetry. Es una implementación de la interfaz `javax.security.auth.spi.LoginModule` . Utilícelo para desarrollar un método de autenticación propio.

Las clases nuevas de LoginModule que proporcione deben encontrarse en la vía de acceso de clase del servicio de telemetría (MQXR). No coloque sus clases en directorios de IBM MQ que estén en la vía de

acceso de clases. Cree sus propios directorios y defina la vía de acceso de clase completa del servicio de telemetría (MQXR).

Puede aumentar la vía de acceso de clases que utiliza el servicio de telemetría (MQXR) estableciendo la vía de acceso de clases en el archivo `service.env`. CLASSPATH debe estar en mayúsculas y la sentencia de vía de acceso de clase sólo puede contener literales. No puede utilizar variables en la CLASSPATH; por ejemplo, CLASSPATH=%CLASSPATH% es incorrecto. El servicio de telemetría (MQXR) establece su propia vía de acceso de clases. Se le añade CLASSPATH definida en `service.env`.

El servicio de telemetría (MQXR) proporciona dos devoluciones de llamada que devuelven el valor de `Username` y de `Password` del cliente conectado al canal MQTT. El `Nombre de usuario` y la `Contraseña` se establecen en el objeto `MqttConnectOptions` . Consulte “Método JAASLoginModule.Login() de ejemplo” en la página 368 para obtener un ejemplo de cómo acceder al `Nombre de usuario` y la `Contraseña`.

Ejemplo de archivo `jaas.config`

Ejemplo de un archivo de configuración JAAS con una configuración con nombre, `MQXRConfig`.

```
MQXRConfig {
samples.JAASLoginModule required debug=true;
//com.ibm.security.auth.module.NTLoginModule required;
//com.ibm.security.auth.module.Krb5LoginModule required
//         principal=principal@your_realm
//         useDefaultCcache=TRUE
//         renewTGT=true;
//com.sun.security.auth.module.NTLoginModule required;
//com.sun.security.auth.module.UnixLoginModule required;
//com.sun.security.auth.module.Krb5LoginModule required
//         useTicketCache="true"
//         ticketCache="${user.home}${/}tickets";
};
```

Método `JAASLoginModule.Login()` de ejemplo

Ejemplo de un módulo de inicio de sesión JAAS codificado para recibir el valor de `Username` y de `Password` que proporciona un cliente MQTT.

```
public boolean login()
throws javax.security.auth.login.LoginException {
javax.security.auth.callback.Callback[] callbacks =
new javax.security.auth.callback.Callback[2];
callbacks[0] = new javax.security.auth.callback.NameCallback("NameCallback");
callbacks[1] = new javax.security.auth.callback.PasswordCallback(
"PasswordCallback", false);
try {
callbackHandler.handle(callbacks);
String username = ((javax.security.auth.callback.NameCallback) callbacks[0])
.getName();
char[] password = ((javax.security.auth.callback.PasswordCallback) callbacks[1])
.getPassword();
 // Accept everything.
if (true) {
loggedIn = true;
} else
throw new javax.security.auth.login.FailedLoginException("Login failed");

principal= new JAASPrincipal(username);

} catch (java.io.IOException exception) {
throw new javax.security.auth.login.LoginException(exception.toString());
} catch (javax.security.auth.callback.UnsupportedCallbackException exception) {
throw new javax.security.auth.login.LoginException(exception.toString());
}

return loggedIn;
}
```

Tareas relacionadas

Resolución del problema: El servicio de telemetría no ha llamado al módulo de inicio de sesión JAAS

Referencia relacionada

Clase AuthCallback MQX

Administración de un cliente AMQP

Puede administrar un cliente AMQP utilizando IBM MQ Explorer o en una línea de mandatos. Utilice el Explorador para configurar canales y supervisar clientes AMQP conectados a IBM MQ. Configure la seguridad de los clientes AMQP utilizando TLS y JAAS.

Antes de empezar

Para obtener información sobre la instalación de AMQP en la plataforma, consulte Elección de qué instalar.

Administración utilizando IBM MQ Explorer

Utilice el Explorador para configurar canales AMQP y supervisar los clientes AMQP que están conectados a IBM MQ. Puede configurar la seguridad de los clientes AMQP utilizando TLS y JAAS.

Administración utilizando la línea de mandatos

Puede administrar un cliente AMQP en la línea de mandatos utilizando mandatos MQSC.

Windows V 9.4.0 Linux V 9.4.0 AIX El servicio AMQP no se inicia automáticamente al iniciar el gestor de colas

A partir de IBM MQ 9.4.0, el comportamiento predeterminado del valor del atributo **CONTROL** para iniciar el servicio AMQP ha cambiado. Al crear e iniciar un nuevo gestor de colas, el servicio AMQP no se inicia automáticamente como parte del proceso de inicio del gestor de colas.

Entre IBM MQ 9.0.4 y IBM MQ 9.4.0, el comportamiento predeterminado del valor del atributo **CONTROL** para iniciar el servicio AMQP es QMGR.

Si se ha instalado el componente AMQP, el servicio AMQP se inicia automáticamente, aunque no se utilice. Para evitar el inicio predeterminado de la máquina virtual Java (JVM) AMQP, tenía dos opciones:

- No instalar el componente AMQP, o
- Cambio del atributo **CONTROL** del servicio AMQP a MANUAL después de iniciar el gestor de colas.

A partir de IBM MQ 9.4.0, los gestores de colas recién creados han revertido el valor del atributo **CONTROL** de SYSTEM.AMQP.SERVICE a MANUAL, que era el valor predeterminado anterior a IBM MQ 9.0.4.

Los gestores de colas migrados, si utilizan AMQP, continúan iniciando automáticamente el servicio durante el inicio del gestor de colas. Para determinar si se ha utilizado AMQP, se comprueba lo siguiente:

- Canales AMQP existentes
- El canal ha iniciado los mensajes en las anotaciones de error de AMQP.

Atención:

- Esto sólo ocurre una vez; la primera vez que se inicia el gestor de colas después de una actualización.
- Durante la migración, si el atributo **CONTROL** se cambia de QMGR a MANUAL, se registra un mensaje informativo en el registro de errores de IBM MQ para indicar el cambio. Consulte Ubicación de registros de AMQP, registros de errores y archivos de configuración para obtener más información.

Si desea que el servicio AMQP se inicie automáticamente, cambie el atributo **CONTROL** del servicio a QMGR y reinicie el gestor de colas. Los reinicios posteriores del gestor de colas inician el servicio AMQP.

Visualización de objetos de IBM MQ en uso por clientes AMQP

Puede ver los distintos recursos de IBM MQ utilizados por los clientes AMQP, por ejemplo, conexiones y suscripciones.

Conexiones

Cuando se inicia el servicio AMQP se crean nuevos Hconns y se conectan al gestor de colas. Esta agrupación de Hconns se utiliza cuando los clientes AMQP publican mensajes. Puede ver los Hconns utilizando el mandato **DISPLAY CONN** . Por ejemplo:

```
DISPLAY CONN(*) TYPE(CONN) WHERE (APPLDESC LK 'IBM MQ Advanced Message Queuing Protocol*')
```

Este mandato también muestra Hconns específicos del cliente. Los Hconns que tienen un atributo de ID de cliente en blanco son los Hconns que se utilizan en la agrupación

Cuando un cliente AMQP se conecta a un canal AMQP, se conecta un nuevo Hconn al gestor de colas. Este Hconn se utiliza para consumir mensajes de forma asíncrona para las suscripciones que el cliente AMQP ha creado. Puede ver el Hconn utilizado por un cliente AMQP determinado utilizando el mandato **DISPLAY CONN** . Por ejemplo:

```
DISPLAY CONN(*) TYPE(CONN) WHERE (CLIENTID EQ 'recv_abcd1234')
```

Suscripciones creadas por clientes

Cuando un cliente AMQP se suscribe a un tema, se crea una nueva suscripción de IBM MQ. El nombre de suscripción incluye la siguiente información:

- El nombre del cliente. Si el cliente se ha unido a una suscripción compartida, se utiliza el nombre de la unidad compartida.
- El patrón de tema al que se ha suscrito el cliente
- Un prefijo. El prefijo es `private` si el cliente ha creado una suscripción no compartida o `share` si el cliente se ha unido a una suscripción compartida

Para ver las suscripciones en uso por un cliente AMQP determinado, ejecute el mandato **DISPLAY SUB** y filtre por el prefijo `private` :

```
DISPLAY SUB(':private:*')
```

Para ver las suscripciones compartidas que clientes utilizan, ejecute el mandato **DISPLAY SUB** y filtre por el prefijo `share`:

```
DISPLAY SUB(':share:*')
```

Dado que varios clientes AMQP pueden utilizar suscripciones compartidas, es posible que desee ver los clientes que consumen actualmente mensajes de la suscripción compartida. Puede hacerlo listando los Hconns que actualmente tienen un manejador abierto en la cola de suscripción. Para ver los clientes que actualmente utilizan una unidad compartida, realice los siguientes pasos:

1. Busque el nombre de cola que utiliza como destino la suscripción compartida. Por ejemplo:

```
DISPLAY SUB(':private:recv_e298452:public') DEST
   5 : DISPLAY SUB(':private:recv_e298452:public') DEST
AMQ8096: WebSphere MQ subscription inquired.
   SUBID(414D5120514D312020202020202020707E0A565C2D0020)
   SUB(:private:recv_e298452:public)
   DEST(SYSTEM.MANAGED.DURABLE.560A7E7020002D5B)
```

2. Ejecute el mandato **DISPLAY CONN** para buscar los manejadores abiertos en dicha cola:

```
DISPLAY CONN(*) TYPE(HANDLE) WHERE  (OBJNAME
EQ SYSTEM.MANAGED.DURABLE.560A7E7020002D5B)
   21 : DISPLAY CONN(*) TYPE(HANDLE) WHERE(OBJNAME EQ
SYSTEM.MANAGED.DURABLE.560A7E7020002D5B)

AMQ8276: Display Connection details.
   CONN(707E0A56642B0020)
   EXTCONN(414D5143514D31202020202020202020)
   TYPE(HANDLE)

   OBJNAME(SYSTEM.BASE.TOPIC)      OBJTYPE(TOPIC)

   OBJNAME(SYSTEM.MANAGED.DURABLE.560A7E7020002961)
   OBJTYPE(QUEUE)
```

3. Para cada uno de los manejadores, visualice el ID de cliente AMQP que tiene el descriptor de contexto abierto:

```
DISPLAY CONN(707E0A56642B0020) CLIENTID
   23 : DISPLAY CONN(707E0A56642B0020) CLIENTID

AMQ8276: Display Connection details.
   CONN(707E0A56642B0020)
   EXTCONN(414D5143514D31202020202020202020)
   TYPE(CONN)
   CLIENTID(recv_8f02c9d)
DISPLAY CONN(707E0A565F290020) CLIENTID
   24 : DISPLAY CONN(707E0A565F290020) CLIENTID
AMQ8276: Display Connection details.
   CONN(707E0A565F290020)
   EXTCONN(414D5143514D31202020202020202020)
   TYPE(CONN)
   CLIENTID(recv_86d8888)
```

Identificación, autorización y autenticación del cliente AMQP

Como otras aplicaciones cliente IBM MQ, puede proteger las conexiones AMQP de varias formas.

Puede utilizar las siguientes características de seguridad para proteger las conexiones AMQP para IBM MQ:

- Registros de autenticación de canal
- Autenticación de conexión
- Configuración de usuario de MCA de canal
- Definiciones de autoridad de IBM MQ
- Conectividad de TLS

Desde una perspectiva de seguridad, establecer una conexión consta de los dos pasos siguientes:

- Decidir si la conexión debe continuar
- Decidir qué identidad de IBM MQ asume la aplicación para comprobaciones de autorización posteriores

La siguiente información describe distintas configuraciones de IBM MQ y los pasos que se siguen cuando un cliente AMQP intenta realizar una conexión. No todas las configuraciones de IBM MQ utilizan todos los pasos descritos. Por ejemplo, algunas configuraciones no utilizan TLS para conexiones dentro del cortafuegos de la empresa y algunas configuraciones utilizan TLS pero no utilizan certificados de cliente para la autenticación. Muchos entornos no utilizan módulos personalizados o JAAS personalizados.

Establecer una conexión

Los siguientes pasos describen lo que sucede cuando se establece una conexión por un cliente AMQP. Los pasos determinan si la conexión continua y qué identidad de IBM MQ asume la aplicación para las comprobaciones de autorización:

1. Si el cliente abre una conexión TLS a IBM MQ y proporciona un certificado, el gestor de colas intenta validar el certificado de cliente.
2. Si el cliente proporciona el nombre de usuario y las credenciales de usuario, el gestor de colas recibe una trama AMQP SASL y se comprueba la configuración de MQ CONNAUTH.
3. Se comprueban las reglas de autenticación de canal MQ (por ejemplo, si la dirección IP y el DNS de certificado TLS son válidos)
4. El canal MCAUSER se confirma, a menos que las reglas de autenticación de canal determinen lo contrario.
5. Si se ha configurado un módulo JAAS, este se invoca
6. La comprobación de autorización de MQ CONNECT aplicada al ID de usuario MQ resultante.
7. Conexión establecida con una identidad de IBM MQ asumida.

Publicar un mensaje

Los siguientes pasos describen lo que sucede cuando se publica un mensaje por un cliente AMQP. Los pasos determinan si la conexión continua y qué identidad de IBM MQ asume la aplicación para las comprobaciones de autorización:

1. La trama para adjuntar enlaces AMQP llega al gestor de colas. La autorización de publicación de IBM MQ para la serie de tema especificada se comprueba para la identidad de usuario MQ establecida durante la conexión.
2. El mensaje se publica en una serie de tema indicada.

Suscribirse a un patrón de tema

Los siguientes pasos describen lo que sucede cuando un cliente AMQP se suscribe a un patrón de tema. Los pasos determinan si la conexión continua y qué identidad de IBM MQ asume la aplicación para las comprobaciones de autorización:

1. La trama para adjuntar enlaces AMQP llega al gestor de colas. La autorización de suscripción de IBM MQ para el patrón de tema especificado se comprueba para la identidad de usuario MQ establecida durante la conexión.
2. La suscripción se crea.

Autorización e identidad del cliente AMQP

Utilice el ID de cliente AMQP, el nombre de usuario AMQP o una identidad de cliente común definida en el canal o en una regla de autenticación de canal, para obtener autorización para acceder a objetos de IBM MQ.

El administrador elige cuándo definir o modificar el canal AMQP, configurando el valor CONNAUTH del gestor de colas, o definiendo las reglas de autenticación de canal. La identidad se utiliza para autorizar el acceso a temas de IBM MQ. La elección se realiza basándose en lo siguiente:

1. El atributo USECLNTID del canal.
2. El atributo ADOPTCTX de la regla CONNAUTH del gestor de colas.
3. El atributo MCAUSER definido en el canal.
4. El atributo USERSRC de una regla de autenticación de canal coincidente.

Evitar problemas: Posteriormente, se hará referencia a la identidad que elija el proceso, por ejemplo mediante el mandato DISPLAY CHSTATUS (AMQP), como el MCAUSER del cliente. Tenga en cuenta que esta no es necesariamente la misma identidad que el MCAUSER del canal al que se hace referencia en la opción (2).

Utilice el mandato de IBM MQ **setmqaut** para seleccionar qué objetos, y qué acciones, están autorizados para que los utilice la identidad asociada al canal AMQP. Por ejemplo, los mandatos siguientes autorizan una identidad de canal `AMQPClient`, proporcionada por el administrador del gestor de colas `QM1`:

```
setmqaut -m QM1 -t topic -n SYSTEM.BASE.TOPIC -p AMQPClient -all +pub +sub
```

y

```
setmqaut -m QM1 -t qmgr -p AMQPClient -all +connect
```

Autenticación de cliente AMQP utilizando una contraseña

Autentique el nombre de usuario del cliente AMQP utilizando la contraseña del cliente. Puede autenticar el cliente utilizando una identidad diferente a la utilizada para autorizar al cliente a publicar temas y a suscribirse a los mismos.

El servicio AMQP puede utilizar MQ CONNAUTH o JAAS para autenticar el nombre de usuario del cliente. Si uno de ellos está configurado, la contraseña proporcionada por el cliente la verifica la configuración CONNAUTH MQ o el módulo JAAS.

El procedimiento siguiente describe los pasos de ejemplo para autenticar usuarios individuales con los usuarios y contraseñas del sistema operativo local y, si tiene éxito, adoptar la identidad común `AMQPUser`:

1. El administrador de IBM MQ establece la identidad MCAUSER del canal AMQP en cualquier nombre, como por ejemplo `AMQPUser`, utilizando IBM MQ Explorer.
2. El administrador de IBM MQ autoriza a `AMQPUser` a publicar y a suscribirse a cualquier tema:

   ```
   setmqaut -m QM1 -t topic -n SYSTEM.BASE.TOPIC -p AMQPUser -all +pub +sub +connect
   ```

3. El administrador de IBM MQ configura una regla IDPWOS CONNAUTH para comprobar el nombre de usuario y la contraseña indicados por el cliente. La regla CONNAUTH debe establecer CHCKCLNT(REQUIRED) y ADOPTCTX(NO).

Nota: Se recomienda utilizar reglas de autenticación de canal y establecer el atributo de canal MCAUSER en un usuario que no tenga privilegios, para permitir un mayor control sobre las conexiones con el gestor de colas.

Privacidad de las publicaciones en los canales

La privacidad de publicaciones AMQP enviadas a cualquier dirección mediante los canales AMQP se protege utilizando TLS para cifrar transmisiones a través de la conexión.

Los clientes AMQP que se conecten a los canales de AMQP utilizan TLS para proteger la privacidad de las publicaciones transmitidas en el canal, mediante el cifrado de claves simétricas. Puesto que los puntos finales no se autentican, no se puede confiar en un canal de cifrado solo. Combine la protección de privacidad con la autenticación del servidor o mutua.

Algunos tipos de redes privadas virtuales (VPN) como, por ejemplo, IPsec, autentican los puntos finales de una conexión TCP/IP, como alternativa a utilizar TLS. VPN cifra cada paquete IP que se transmite por la red. Una vez establecida la conexión VPN, se ha establecido la red de confianza. Puede conectar clientes AMQP a canales AMQP utilizando TCP/IP en la red VPN.

El cifrado de las conexiones TLS sin autenticar el servidor expone la conexión frente a ataques de terceros. Aunque la información que se intercambia está protegida contra escuchas no autorizadas, no sabe con quién se está intercambiando. A menos que controle la red, estará expuesto a que alguien intercepte las transmisiones IP, haciéndose pasar por el punto final.

Puede crear una conexión TLS cifrada, sin autenticar el servidor, mediante un intercambio de claves Diffie-Hellman CipherSpec que dé soporte a TLS anónima. El secreto maestro, compartido entre el

cliente y el servidor, y que se utiliza para cifrar transmisiones TLS, se establece sin intercambiar ningún certificado de servidor firmado en privado.

Puesto que las conexiones anónimas son inseguras, la mayoría de las implementaciones TLS no toman como valor predeterminado la utilización de las CipherSpecs anónimas. Si un canal de AMQP acepta una conexión TLS que solicita un cliente, el canal deberá tener un almacén de claves protegido mediante una frase de contraseña. De forma predeterminada, puesto que las implementaciones TLS no utilizan las CipherSpecs anónimas, el almacén de claves debe contener un certificado de firma privada que demuestre que el cliente puede autenticarse.

Si utiliza las CipherSpecs anónima, el almacén de claves del servidor debe existir, pero no es necesario que contenga ningún certificado firmado en privado.

Otra forma de establecer una conexión cifrada es sustituir el proveedor de confianza en el cliente por su propia implementación. El proveedor de confianza no ha podría autenticar el certificado de servidor, pero la conexión se cifraría.

Configuración de clientes AMQP con TLS

Puede configurar clientes AMQP para que utilicen TLS para proteger los datos que fluyen a través de la red y para autenticar la identidad del gestor de colas al que se conecta el cliente.

Para utilizar TLS para la conexión desde un cliente AMQP a un canal AMQP, debe asegurarse de que el gestor de colas se ha configurado para TLS. Configuración de TLS en los gestores de colas describe cómo configurar el almacén de claves del que un gestor de colas lee los certificados TLS.

Cuando el gestor de colas se ha configurado con un almacén de claves, debe configurar los atributos TLS en el canal AMQP al que se conectarán los clientes. Los canales AMQP tienen siguientes cuatro atributos relacionados con la configuración de TLS:

SSLCAUTH
El atributo SSLCAUTH se utiliza para especificar si el gestor de colas debe requerir que un cliente AMQP presente un certificado de cliente para verificar su identidad.

SSLCIPH
El atributo SSLCIPH especifica el cifrado que el canal debe utilizar para codificar datos en el flujo TLS.

V 9.4.0 A partir de IBM MQ 9.4.0, los canales AMQP dan soporte a CipherSpecsgenéricas ANY*. Para obtener más información sobre CipherSpecs, consulte Habilitación de CipherSpecs.

SSLPEER
El atributo SSLPEER se utiliza para especificar el nombre distinguido (DN) con el que debe coincidir un certificado de cliente si una conexión va a estar permitida.

CERTLABL
CERTLABL especifica el certificado que el gestor de colas debe presentar al cliente. El almacén de claves del gestor de colas puede contener varios certificados. Este atributo permite especificar el certificado que debe utilizarse para las conexiones con este canal. Si no se ha especificado ningún CERTLABL, se utiliza el certificado en el repositorio de claves del gestor de colas con la etiqueta que corresponde al atributo CERTLABL del gestor de colas.

Cuando haya configurado el canal AMQP con los atributos TLS, debe reiniciar el servicio AMQP utilizando el siguiente mandato:

```
STOP SERVICE(SYSTEM.AMQP.SERVICE) START SERVICE(SYSTEM.AMQP.SERVICE)
```

Cuando un cliente AMQP se conecta a un canal AMQP protegido por TLS, el cliente verifica la identidad del certificado presentado por el gestor de colas. Para ello, debe configurar el cliente AMQP con un almacén de confianza que contenga el certificado del gestor de colas. Los pasos para hacerlo varían en función del cliente AMQP que esté utilizando. Para obtener información sobre los distintos clientes y API de AMQP, consulte la documentación del cliente AMQP correspondiente.

Referencia relacionada
DEFINE CHANNEL (definir un nuevo canal)

STOP SERVICE (detener un servicio) en Multiplatforms
START SERVICE (iniciar un servicio) en Multiplatforms

Desconexión de clientes AMQP del gestor de colas

Si desea desconectar los clientes AMQP del gestor de colas, ejecute el mandato PURGE CHANNEL o detenga la conexión con el cliente AMQP.

- Ejecute el mandato **PURGE CHANNEL**. Por ejemplo:

```
PURGE CHANNEL(MYAMQP) CLIENTID('recv_28dbb7e')
```

- De forma alternativa, detenga la conexión que el cliente AMQP está utilizando para desconectar el cliente realizando los pasos siguientes:

  1. Busque la conexión que utiliza el cliente ejecutando el mandato **DISPLAY CONN** . Por ejemplo:

     ```
     DISPLAY CONN(*) TYPE(CONN) WHERE (CLIENTID EQ 'recv_28dbb7e')
     ```

     La salida del mandato es la siguiente:

     ```
     DISPLAY CONN(*)  TYPE(CONN) WHERE(CLIENTID EQ 'recv_28dbb7e')
       40 : DISPLAY CONN(*) TYPE(CONN) WHERE(CLIENTID EQ 'recv_28dbb7e')
     AMQ8276: Display  Connection details.
       CONN(707E0A565F2D0020)
       EXTCONN(414D5143514D31202020202020202020)
       TYPE(CONN)
       CLIENTID(recv_28dbb7e)
     ```

  2. Detenga la conexión. Por ejemplo:

     ```
     STOP CONN(707E0A565F2D0020)
     ```

Administración de multidifusión

Utilice esta información para conocer las tareas de administración de IBM MQ Multicast, tales como reducir el tamaño de los mensajes de multidifusión y habilitar la conversión de datos.

Iniciación a la multidifusión

Utilice esta información para empezar a trabajar con temas y objetos de información de comunicación de IBM MQ Multicast.

Acerca de esta tarea

La mensajería de IBM MQ Multicast utiliza la red para entregar mensajes mediante la correlación de temas con direcciones de grupo. Las tareas siguientes son una forma rápida de probar si la dirección IP y puerto necesarios están configurados correctamente para la mensajería de multidifusión.

Creación de un objeto COMMINFO para multidifusión

El objeto de información de comunicación (COMMINFO) contiene los atributos asociados con la transmisión de multidifusión. Para obtener más información sobre los parámetros de objeto COMMINFO, consulte DEFINE COMMINFO.

Utilice el siguiente ejemplo de línea de mandatos para definir un objeto COMMINFO para multidifusión:

```
DEFINE COMMINFO(MC1) GRPADDR(group address) PORT(port number)
```

donde *MC1* es el nombre del objeto COMMINFO, *dirección de grupo* es la dirección IP o el nombre DNS de multidifusión de grupo y *número de puerto*es el puerto en el que se va a transmitir (el valor predeterminado es 1414).

Se crea un nuevo objeto COMMINFO denominado *MC1*; este nombre es el que debe especificar al definir un objeto TOPIC en el ejemplo siguiente.

Creación de un objeto TOPIC para multidifusión

Un tema es el asunto de la información que se publica en un mensaje de publicación/suscripción y un tema se define creando un objeto TOPIC. Los objetos TOPIC tienen dos parámetros que definen si pueden utilizarse con multidifusión o no. Estos parámetros son: **COMMINFO** y **MCAST**.

- **COMMINFO** Este parámetro especifica el nombre del objeto de información de comunicación de multidifusión. Para obtener más información sobre los parámetros de objeto COMMINFO, consulte DEFINE COMMINFO.
- **MCAST** Este parámetro especifica si la multidifusión está permitida en esta posición del árbol de temas.

Utilice el siguiente ejemplo de línea de mandatos para definir un objeto TOPIC para multidifusión:

```
DEFINE TOPIC(ALLSPORTS) TOPICSTR('Sports') COMMINFO(MC1) MCAST(ENABLED)
```

Se crea un nuevo objeto TOPIC denominado *ALLSPORTS*. Tiene una serie de tema *Sports*, su objeto de información de comunicación relacionado se denomina *MC1*(que es el nombre que ha especificado al definir un objeto COMMINFO en el ejemplo anterior), y la multidifusión está habilitada.

Prueba de la publicación/suscripción de multidifusión

Después de que se hayan creado los objetos TOPIC y COMMINFO, se pueden probar utilizando el ejemplo `amqspubc` y el ejemplo `amqssubc`. Para obtener más información sobre estos ejemplos, consulte Los programas de ejemplo de publicación/suscripción.

1. Abra dos ventanas de línea de mandatos; la primera línea de mandatos es para el ejemplo de publicación `amqspubc` y la segunda línea de mandatos es para el ejemplo de suscripción `amqssubc`.
2. Entre el siguiente mandato en la línea de mandatos 1:

   ```
   amqspubc Sports QM1
   ```

   donde *Sports* es la serie de tema del objeto TOPIC definido en un ejemplo anterior, y *QM1* es el nombre del gestor de colas.
3. Entre el siguiente mandato en la línea de mandatos 2:

   ```
   amqssubc Sports QM1
   ```

   donde *Sports* y *QM1* son los mismos que se utilizaron en el paso “2” en la página 376.
4. Especifique `Hello world` en la línea de mandatos 1. Si el puerto y la dirección IP especificados en el objeto COMMINFO se han configurado correctamente; el ejemplo `amqssubc` , que está a la escucha en el puerto de publicaciones de la dirección especificada, genera `Hello world` en la línea de mandatos 2.

Topología de temas de IBM MQ Multicast

Utilice este ejemplo para entender la topología de temas de IBM MQ Multicast.

El soporte de IBM MQ Multicast requiere que cada subárbol tenga su propio grupo y corriente de datos de multidifusión dentro de la jerarquía total.

El esquema de direccionamiento IP *classful network* ha designado espacio de direcciones para la dirección de multidifusión. El rango completo de multidifusión de la dirección IP es `224.0.0.0`

a 239.255.255.255, pero algunas de estas direcciones están reservadas. Para obtener una lista de direcciones reservadas, póngase en contacto con el administrador del sistema o consulte https://www.iana.org/assignments/multicast-addresses para obtener más información. Se recomienda utilizar la dirección de multidifusión con ámbito local en el rango de 239.0.0.0 a 239.255.255.255.

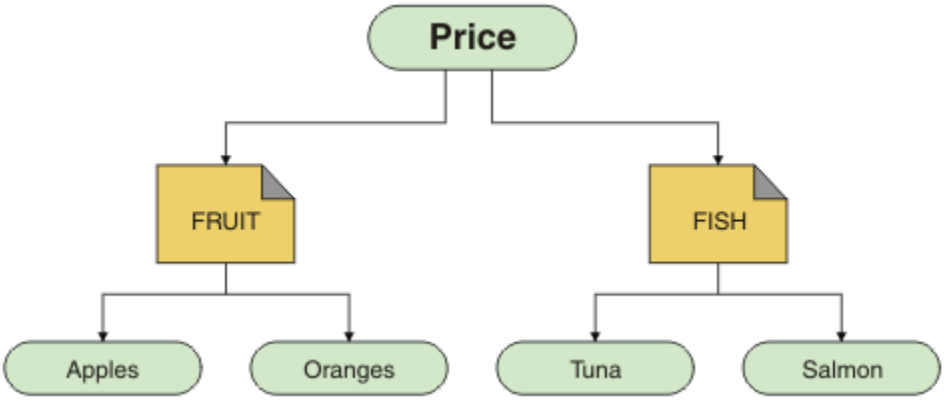
En el diagrama siguiente, hay dos posibles corrientes de datos de multidifusión:

```
DEF COMMINFO(MC1) GRPADDR(239.XXX.XXX.XXX
)

DEF COMMINFO(MC2) GRPADDR(239.YYY.YYY.YYY)
```

donde *239.XXX.XXX.XXX* y *239.YYY.YYY.YYY* son direcciones de multidifusión válidas.

Estas definiciones de temas se utilizan para crear un árbol de temas tal como se muestra en el diagrama siguiente:

```
DEFINE TOPIC(FRUIT) TOPICSTRING('Price/FRUIT') MCAST(ENABLED) COMMINFO(MC1)

DEFINE TOPIC(FISH) TOPICSTRING('Price/FISH') MCAST(ENABLED) COMMINFO(MC2)
```



Cada objeto de información de comunicación de multidifusión (COMMINFO) representa una corriente de datos diferente debido a que las direcciones de grupo son diferentes. En este ejemplo, el tema FRUIT se ha definido para utilizar el objeto COMMINFO MC1, el tema FISH se ha definido para utilizar el objeto COMMINFO MC2 y el nodo Price no tiene definiciones de multidifusión.

IBM MQ Multicast tiene un límite de 255 caracteres para las series de tema. Esta limitación significa que hay que tener cuidado con los nombres de nodos y los nodos hoja dentro del árbol; si los nombres de nodos y los nodos hoja son demasiado largos, la serie de tema puede superar los 255 caracteres y devolver el código de razón de 2425 (0979) (RC2425): MQRC_TOPIC_STRING_ERROR. Es recomendable que las series de tema sean lo más cortas posible porque unas series de tema más largas pueden tener un efecto perjudicial en el rendimiento.

Control del tamaño de mensajes de multidifusión

Utilice esta información para conocer el formato de mensaje de IBM MQ, y para reducir el tamaño de los mensajes de IBM MQ.

Los mensajes de IBM MQ tienen varios atributos asociados con ellos que están contenidos en el descriptor de mensaje. Para los mensajes pequeños, estos atributos pueden representar la mayor parte del tráfico de datos y pueden tener un importante efecto perjudicial sobre la velocidad de transmisión. IBM MQ Multicast permite al usuario configurar cuáles de estos atributos, si los hay, se transmiten junto con el mensaje.

La presencia de atributos de mensaje, distintos de la serie de tema, depende de si el objeto COMMINFO indica que deben enviarse o no. Si un atributo no se transmite, la aplicación receptora aplica un valor

predeterminado. Los valores MQMD predeterminados no son necesariamente los mismos que el valor MQMD_DEFAULT y se describen en Tabla 19 en la página 378.

El objeto COMMINFO contiene el atributo MCPROP, que controla cuántos campos MQMD y propiedades de usuario fluyen con el mensaje. Al establecer el valor de este atributo en un nivel apropiado, puede controlar el tamaño de los mensajes de IBM MQ Multicast:

MCPROP
Las propiedades multidifusión controla cuántas de las propiedades MQMD y de las propiedades de usuario fluyen con el mensaje.

TODOS
Todas las propiedades de usuario y todos los campos de MQMD se transmiten.

REPLY
Sólo se transmiten las propiedades de usuario y los campos MQMD que están relacionados con la respuesta a los mensajes. Estas propiedades son:

- MsgType
- MessageId
- CorrelId
- ReplyToQ
- ReplyToQmgr

USER
Sólo se transmiten las propiedades de usuario

NINGUNO
No se transmiten las propiedades de usuario ni los campos MQMD

COMPAT
Este valor hace que la transmisión del mensaje se realice en una modalidad compatible para RMM, que permite determinada interoperación con las aplicaciones XMS actuales y las aplicaciones IBM Integration Bus RMM.

Removed XMS .NETMensajería de multidifusión (usandoRMM) quedó obsoleto desdeIBM MQ 9.2 y eliminado enIBM MQ 9.3 .

Atributos de los mensajes de multidifusión

Los atributos de mensaje puede proceder de varios lugares, como MQMD, los campos de MQRFH2 y propiedades de mensaje.

La tabla siguiente muestra lo que sucede cuando se envían mensajes sujetos al valor de MCPROP (se describe antes en esta sección), y el valor predeterminado utilizado cuando un atributo no se envía.

Tabla 19. Atributos de mensajería y cómo se relacionan con la multidifusión

Atributo	Acción cuando se utiliza multidifusión	Valor predeterminado si no se transmite
TopicString	Siempre incluido	No aplicable
MQMQ StrucId	No transmitido	No aplicable
MQMD Version	No transmitido	No aplicable
Informe	Incluido si no es predeterminado	0
MsgType	Incluido si no es predeterminado	MQMT_DATAGRAM
Caducidad	Incluido si no es predeterminado	0
Comentarios	Incluido si no es predeterminado	0

Tabla 19. Atributos de mensajería y cómo se relacionan con la multidifusión (continuación)		
Atributo	**Acción cuando se utiliza multidifusión**	**Valor predeterminado si no se transmite**
Encoding	Incluido si no es predeterminado	MQENC_NORMAL(equiv)
CodedCharSetId	Incluido si no es predeterminado	1208
Formato	Incluido si no es predeterminado	MQRFH2
Prioridad	Incluido si no es predeterminado	4
Persistence	Incluido si no es predeterminado	MQPER_NOT_PERSISTENT
MsgId	Incluido si no es predeterminado	Null
CorrelId	Incluido si no es predeterminado	Null
BackoutCount	Incluido si no es predeterminado	0
ReplyToQ	Incluido si no es predeterminado	Espacio en blanco
ReplyToQMgr	Incluido si no es predeterminado	Espacio en blanco
UserIdentifier	Incluido si no es predeterminado	Espacio en blanco
AccountingToken	Incluido si no es predeterminado	Null
PutAppIType	Incluido si no es predeterminado	MQAT_JAVA
PutAppIName	Incluido si no es predeterminado	Espacio en blanco
PutDate	Incluido si no es predeterminado	Espacio en blanco
PutTime	Incluido si no es predeterminado	Espacio en blanco
ApplOriginData	Incluido si no es predeterminado	Espacio en blanco
GroupID	Excluido	No aplicable
MsgSeqNumber	Excluido	No aplicable
Desplazamiento	Excluido	No aplicable
MsgFlags	Excluido	No aplicable
OriginalLength	Excluido	No aplicable
UserProperties	Incluido	No aplicable

Referencia relacionada

Multi ALTER COMMINFO

DEFINE COMMINFO

Habilitación de la conversión de datos para la mensajería de Multicast

Utilice esta información para entender cómo funciona la conversión de datos para la mensajería de IBM MQ Multicast.

IBM MQ Multicast es un protocolo sin conexión compartido y, por tanto, no es posible que cada cliente realice solicitudes específicas para la conversión de datos. Todos los clientes suscritos a la misma secuencia de multidifusión reciben los mismos datos binarios; por lo tanto, si es necesaria la conversión de datos de IBM MQ, esta se realiza localmente en cada cliente.

En una instalación de plataforma mixta, es posible que la mayoría de los clientes requieran los datos en un formato que no sea el formato nativo de la aplicación transmisora. En esta situación, los valores de

CCSID y **ENCODING** del objeto COMMINFO de multidifusión se pueden utilizar para definir la codificación de la transmisión de mensajes para conseguir eficiencia.

IBM MQ Multicast da soporte a la conversión de datos de la carga útil de mensaje para los siguientes formatos incorporados:

- MQADMIN
- MQEVENT
- MQPCF
- MQRFH
- MQRFH2
- MQSTR

Además de estos formatos, también puede definir sus propios formatos y utilizar una salida de conversión de datos de MQDXP: Parámetro de salida de conversión de datos.

Si desea información sobre la programación de conversiones de datos, consulte Conversión de datos en MQI para la mensajería de multidifusión.

Para obtener más información sobre la conversión de datos, consulte Conversión de datos.

Si desea más información sobre las salidas de la conversión de datos y `ClientExitPath`, consulte Stanza ClientExitPath del archivo de configuración de cliente.

Supervisión de aplicaciones de multidifusión

Utilice esta información para aprender sobre la administración y la supervisión de IBM MQ Multicast.

El estado de los publicadores y suscriptores actuales para el tráfico de multidifusión (por ejemplo, el número de mensajes enviados y recibidos, o el número de mensajes perdidos) se transmite periódicamente al servidor desde el cliente. Cuando se recibe el estado, el atributo COMMEV del objeto COMMINFO especifica si el gestor de colas transmite o no un mensaje de suceso en SYSTEM.ADMIN.PUBSUB.EVENT. El mensaje de suceso contiene la información de estado recibida. Esta información es una inestimable ayuda de diagnóstico para buscar el origen de un problema.

Utilice el mandato MQSC **DISPLAY CONN** para visualizar información de conexión sobre las aplicaciones conectadas al gestor de colas. Para obtener más información sobre el mandato **DISPLAY CONN**, consulte DISPLAY CONN.

Utilice el mandato MQSC **DISPLAY TPSTATUS** para visualizar el estado de los publicadores y suscriptores. Para obtener más información sobre el mandato **DISPLAY TPSTATUS**, consulte DISPLAY TPSTATUS.

COMMEV y el indicador de la fiabilidad de mensajes de multidifusión

El *indicador de confiabilidad*, utilizado junto con el atributo **COMMEV** del objeto COMMINFO, es un elemento clave en la supervisión de publicadores y suscriptores de IBM MQ Multicast. El indicador de fiabilidad (el campo **MSGREL** que se devuelve en los mandatos de estado de Publicación o Suscripción) es un indicador de IBM MQ que muestra el porcentaje de transmisiones que no tienen errores. A veces se tienen que retransmitir mensajes debido a un error de transmisión, lo que se refleja en el valor de **MSGREL**. Las causas potenciales de errores de transmisión incluyen los suscriptores lentos, redes ocupadas e interrupciones de la red. **COMMEV** controla si se generan mensajes de suceso para manejadores de multidifusión que se crean mediante el objeto COMMINFO y se establece en uno de tres valores posibles:

DISABLED
Los mensajes de suceso no se graban.

ENABLED
Los mensajes de suceso se graban siempre, con una frecuencia definida en el parámetro COMMINFO **MONINT**.

EXCEPTION
Los mensajes de suceso se graban si la fiabilidad de los mensajes es inferior al umbral de fiabilidad. Un nivel de fiabilidad de mensaje de 90% o menos indica que puede existir un problema con la configuración de red, o que una o varias de las aplicaciones de publicación/suscripción se ejecuta demasiado lentamente:

- El valor **MSGREL(100,100)** indica que no ha habido problemas en el período de tiempo de corto plazo o largo plazo.
- El valor **MSGREL(80,60)** indica que el 20% de los mensajes tienen problemas actualmente, pero también una mejora del valor a largo plazo de 60.

Los clientes pueden continuar transmitiendo y recibiendo tráfico de multidifusión incluso cuando se interrumpe la conexión de difusión única para el gestor de colas, por lo tanto, los datos pueden estar desfasados.

Fiabilidad de los mensajes de multidifusión

Utilice esta información para aprender a establecer el historial de suscripciones y de mensajes de IBM MQ Multicast.

Un elemento clave para superar una anomalía de transmisión con la multidifusión es el almacenamiento en búfer de los datos transmitidos (un historial de mensajes que se mantiene en el extremo transmisor del enlace) por parte de IBM MQ. Este proceso significa que no es necesario ningún almacenamiento en búfer de mensajes en el proceso de la aplicación transmisora, ya que IBM MQ proporciona la fiabilidad. El tamaño de este historial se configura mediante el objeto de información de comunicación (COMMINFO), tal como se describe en la siguiente información. Un almacenamiento intermedio de transmisión mayor significa que hay más historial de transmisión que retransmitir si es necesario, pero debido a la naturaleza de la multidifusión, no se puede dar soporte a una garantía del 100% de entrega.

El historial de mensajes de IBM MQ Multicast se controla en el objeto de información de comunicación (COMMINFO) mediante el atributo **MSGHIST**:

MSGHIST
Este valor es la cantidad de historial de mensajes en kilobytes que mantiene el sistema para manejar las retransmisiones en el caso de acuses de recibo negativos (NACK).

El valor `0` ofrece el nivel mínimo de fiabilidad. El valor predeterminado es `100` KB.

El historial de nuevas suscripciones de IBM MQ Multicast se controla en el objeto de información de comunicación (COMMINFO) mediante el atributo **NSUBHIST**:

NSUBHIST
El historial de nuevas suscripciones controla si un suscriptor que se suscribe a una corriente de datos de publicación recibe todos los datos disponibles en este momento, o bien recibe únicamente publicaciones desde el momento de la suscripción.

NINGUNO
El valor `NONE`hace que el transmisor transmita sólo las publicaciones realizadas desde el momento de la suscripción. `NONE` es el valor predeterminado.

TODOS
Un valor `ALL` hace que el transmisor retransmita todo el historial del tema que sea conocido. En algunas circunstancias, esta situación puede proporcionar un comportamiento similar a las publicaciones retenidas.

Nota: La utilización del valor `ALL` puede tener un efecto perjudicial en el rendimiento si hay un historial de tema extenso, ya que se retransmite todo el historial del tema.

Referencia relacionada
DEFINE COMMINFO
Multi ALTER COMMINFO

Tareas avanzadas de multidifusión

Utilice esta información para obtener información sobre las tareas avanzadas de administración de multidifusión de IBM MQ, como la configuración de archivos `.ini` y la interoperatividad con IBM MQ LLM.

Si desea ver consideraciones de seguridad en una instalación de multidifusión, consulte Seguridad de multidifusión.

Puente entre dominios de publicación/suscripción de multidifusión y no de multidifusión

Utilice esta información para entender lo que sucede cuando un publicador no de multidifusión publica en un tema habilitado para IBM MQ Multicast.

Si un publicador no de multidifusión publica en un tema definido como habilitado para **MCAST** y para **BRIDGE**, el gestor de colas transmite el mensaje a través de multidifusión directamente a los suscriptores que puedan estar escuchando. Un publicador de multidifusión no puede publicar en los temas que no están habilitados para la multidifusión.

Los temas existentes pueden estar habilitados para multidifusión si se establecen los parámetros **MCAST**y **COMMINFO** de un objeto de tema. Consulte Conceptos de multidifusión iniciales para obtener más información sobre estos parámetros.

El atributo **BRIDGE** del objeto COMMINFO controla las publicaciones de publicaciones que no utilizan multidifusión. Si **BRIDGE** se establece en ENABLED y el parámetro **MCAST** del tema también se establece en ENABLED, las publicaciones de las aplicaciones que no están utilizando multidifusión se usan como puente con las aplicaciones que sí. Para obtener más información sobre el parámetro **BRIDGE**, consulte DEFINE COMMINFO.

Configuración de los archivos .ini para Multicast

Utilice esta información para entender los campos de multidifusión de IBM MQ en los archivos `.ini`.

La configuración adicional de IBM MQ Multicast se puede realizar en un archivo `ini`. El archivo `ini` específico que debe utilizar depende del tipo de aplicaciones:

- Cliente: Configure el archivo *MQ_DATA_PATH* `/mqclient.ini`.
- Gestor de colas: Configure el archivo *MQ_DATA_PATH* `/qmgrs/`*QMNAME*`/qm.ini`.

donde *MQ_DATA_PATH* es la ubicación del directorio de datos IBM MQ (`/var/mqm/mqclient.ini`), y *QMNAME* es el nombre del gestor de colas al que se aplica el archivo `.ini`.

El archivo `.ini` contiene campos utilizados para ajustar el comportamiento de multidifusión de IBM MQ:

```
Multicast:
Protocol       = IP | UDP
IPVersion       = IPv4 | IPv6 | ANY | BOTH
LimitTransRate    = DISABLED | STATIC | DYNAMIC
TransRateLimit    = 100000
SocketTTL       = 1
Batch         = NO
Loop         = 1
Interface       = <IPaddress>
FeedbackMode     = ACK | NACK | WAIT1
HeartbeatTimeout   = 20000
HeartbeatInterval   = 2000
```

Protocolo

UDP
En esta modalidad, los paquetes se envían utilizando el protocolo UDP. Los elementos de la red no pueden proporcionar asistencia en la distribución de multidifusión como lo hacen en la modalidad IP. El formato de paquete sigue siendo compatible con PGM. Éste es el valor predeterminado.

IP
En esta modalidad, el transmisor envía paquetes IP sin formato. Los elementos de red con soporte PGM ayudan en la distribución de paquetes de multidifusión fiables. Esta modalidad es totalmente compatible con el estándar PGM.

IPVersion

IPv4
Comunicarse utilizando sólo el protocolo IPv4. Éste es el valor predeterminado.

IPv6
Comunicarse utilizando sólo el protocolo IPv6.

CUALQUIERA
Comunicarse utilizando IPv4, IPv6, o ambos, dependiendo de qué protocolo está disponible.

BOTH
Admite la comunicación utilizando IPv4 e IPv6.

LimitTransRate

DISABLED
No hay ningún control de velocidad de transmisión. Éste es el valor predeterminado.

STATIC
Implementa el control de velocidad de transmisión estático. El transmisor no podría transmitir a una velocidad que supere la especificada por el parámetro TransRateLimit.

DYNAMIC
El transmisor adapta su velocidad de transmisión de acuerdo con los comentarios que obtiene de los receptores. En este caso, el límite de la velocidad de transmisión no puede ser mayor que el valor especificado por el parámetro TransRateLimit. El transmisor intenta alcanzar una velocidad de transmisión óptima.

TransRateLimit
Límite de la velocidad de transmisión en Kbps.

SocketTTL
El valor de SocketTTL determina si el tráfico de multidifusión puede pasar a través de un direccionador, o el número de direccionadores por los que puede pasar a través.

Lote
Controla si los mensajes se envían por lotes o inmediatamente; hay 2 valores posibles:

- *NO* Los mensajes no se envían por lotes, se envían inmediatamente.
- *YES* Los mensajes se envían por lotes.

Loop
Establezca el valor en 1 para habilitar el bucle de multidifusión. El bucle de multidifusión define si los datos enviados recorren un bucle de retorno al host o no.

Interfaz
Dirección IP de la interfaz en la que fluye el tráfico de multidifusión. Para obtener más información y solucionar problemas, consulte: Prueba de aplicaciones de multidifusión en una red de no multidifusión y Establecimiento de la red adecuada para el tráfico de multidifusión

FeedbackMode

NACK
Comentarios por acuses de recibo negativos. Éste es el valor predeterminado.

ACK
Comentarios por acuses de recibo positivos.

WAIT1
Comentarios por acuses de recibo positivos cuando el transmisor sólo espera 1 ACK de cualquiera de los receptores.

HeartbeatTimeout
Tiempo de espera de pulsaciones en milisegundos. Un valor de 0 indica que los sucesos de tiempo de espera de pulsaciones no están planteadas por el receptor o los receptores del tema. El valor predeterminado es 20000.

HeartbeatInterval
Intervalo de pulsaciones en milisegundos. Un valor de 0 indica que no se envían pulsaciones. El intervalo de pulsaciones debe ser considerablemente más pequeño que el valor **HeartbeatTimeout** para evitar sucesos de tiempo de espera de pulsación falsos. El valor predeterminado es 2000.

Interoperatividad de multidifusión con IBM MQ Low Latency Messaging

Utilice esta información para entender la interoperatividad entre IBM MQ Multicast e IBM MQ Low Latency Messaging (LLM).

La transferencia de carga útil básica es posible para una aplicación que utilice LLM, con otra aplicación que utilice la multidifusión para intercambiar mensajes en ambas direcciones. Aunque la multidifusión utiliza la tecnología LLM, el producto LLM en sí no está incorporado. Por lo tanto, es posible instalar LLM e IBM MQ Multicast, y operar y dar servicio a los dos productos por separado.

Las aplicaciones LLM que se comunican con la multidifusión quizá tengan que enviar y recibir propiedades de mensaje. Las propiedades de mensaje de IBM MQ y los campos MQMD se transmiten como propiedades de mensaje LLM con códigos de propiedad de mensaje LLM específicos tal como se muestra en la siguiente tabla:

Tabla 20. Correlaciones de propiedades de mensaje de IBM MQ con propiedades de IBM MQ LLM

Propiedad de IBM MQ	Tipo de propiedad de IBM MQ LLM	Clase de propiedad de LLM	Código de propiedad de LLM
MQMD.Report	RMM_MSG_PROP_INT32	LLM_PROP_KIND_Int32	-1001
MQMD.MsgType	RMM_MSG_PROP_INT32	LLM_PROP_KIND_Int32	-1002
MQMD.Expiry	RMM_MSG_PROP_INT32	LLM_PROP_KIND_Int32	-1003
MQMD.Feedback	RMM_MSG_PROP_INT32	LLM_PROP_KIND_Int32	-1004
MQMD.Encoding	RMM_MSG_PROP_INT32	LLM_PROP_KIND_Int32	-1005
MQMD.CodedCharSetId	RMM_MSG_PROP_INT32	LLM_PROP_KIND_Int32	-1006
MQMD.Format	RMM_MSG_PROP_BYTES	LLM_PROP_KIND_String	-1007
MQMD.Priority	RMM_MSG_PROP_INT32	LLM_PROP_KIND_Int32	-1008
MQMD.Persistence	RMM_MSG_PROP_INT32	LLM_PROP_KIND_Int32	-1009
MQMD.MsgId	RMM_MSG_PROP_BYTES	LLM_PROP_KIND_ByteArray	-1010
MQMD.BackoutCount	RMM_MSG_PROP_INT32	LLM_PROP_KIND_Int32	-1012
MQMD.ReplyToQ	RMM_MSG_PROP_BYTES	LLM_PROP_KIND_String	-1013
MQMD.ReplyToQMger	RMM_MSG_PROP_BYTES	LLM_PROP_KIND_String	-1014
MQMD.PutDate	RMM_MSG_PROP_BYTES	LLM_PROP_KIND_String	-1020
MQMD.PutTime	RMM_MSG_PROP_BYTES	LLM_PROP_KIND_String	-1021
MQMD.ApplOriginData	RMM_MSG_PROP_BYTES	LLM_PROP_KIND_String	-1022
MQPubOptions	RMM_MSG_PROP_INT32	LLM_PROP_KIND_int32	-1053

Para obtener más información sobre LLM, consulte la documentación del producto LLM: IBM MQ Low Latency Messaging.

IBM i Administración de IBM MQ for IBM i

Los mandatos CL son el método preferido para administrar IBM MQ en IBM i. También puede utilizar mandatos MQSC, mandatos PCF, mandatos de control y administración remota.

Acerca de esta tarea

Las tareas de administración incluyen la creación, inicio, modificación, visualización, detención y supresión de clústeres, procesos y objetos de IBM MQ (gestores de colas, colas, listas de nombres, definiciones de proceso, canales, canales de conexión de clientes, escuchas, servicios y objetos de información de autenticación).

Consulte los enlaces siguientes para obtener detalles sobre cómo administrar IBM MQ for IBM i.

Conceptos relacionados

Explicación de los nombres de biblioteca de gestor de colas de IBM MQ for IBM i

Servicios instalables y componentes en IBM i

Tareas relacionadas

Cambio de la información de configuración de IBM MQ en Multiplatforms

Configuración de la seguridad en IBM i

“Invocación del manejador de cola de mensajes no entregados en IBM i” en la página 169
En IBM MQ for IBM i, invoque el manejador DLQ estableciendo el mandato **STRMQMDLQ** .

Determinación de problemas con aplicaciones IBM MQ for IBM i

Referencia relacionada

Objetos predeterminados y del sistema

IBM i Gestión de IBM MQ for IBM i utilizando mandatos CL

Utilice esta información para obtener una descripción de los mandatos de IBM MQ IBM i.

Se puede acceder a la mayoría de los grupos de mandatos de IBM MQ, incluidos los que están asociados a gestores de colas, colas, temas, canales, listas de nombres, definiciones de proceso y objetos de información de autenticación, mediante el mandato **WRK*** correspondiente.

El principal mandato del grupo es **WRKMQM**. Este mandato le permite, por ejemplo, visualizar una lista de todos los gestores de colas del sistema, junto con la información de estado. De forma alternativa, puede procesar todos los mandatos específicos del gestor de colas utilizando distintas opciones con respecto a cada entrada.

Desde el mandato **WRKMQM** puede seleccionar áreas específicas de cada gestor de colas, por ejemplo, trabajar con canales, temas o colas, y a partir de ahí seleccionar objetos individuales.

Registrar definiciones de aplicaciones de IBM MQ

Al crear o personalizar aplicaciones de IBM MQ, resulta útil mantener un registro de todas las definiciones de IBM MQ que se han creado. Este registro se puede utilizar:

- Fines de recuperación
- Mantenimiento
- Retrotracción de las aplicaciones de IBM MQ

Puede registrar las definiciones de aplicaciones de IBM MQ de una de estas dos maneras:

1. Creando programas CL para generar las definiciones de IBM MQ para el servidor.
2. Creando archivos de texto MQSC como miembros SRC para generar las definiciones de IBM MQ utilizando el lenguaje de mandatos de IBM MQ para múltiples plataformas.

Para obtener más detalles sobre la definición de objetos de cola, consulte "Administración de IBM MQ utilizando mandatos MQSC" en la página 12 y "Utilización de los formatos de mandato programable de IBM MQ" en la página 27.

Referencia relacionada

Referencia de mandatos CL de IBM MQ for IBM i

IBM i Antes de empezar a utilizar IBM MQ for IBM i utilizando mandatos CL

Utilice esta información para iniciar el subsistema de IBM MQ y crear un gestor de colas local.

Antes de empezar

Asegúrese de que el subsistema de IBM MQ está en ejecución (mediante el mandato STRSBS QMQM/QMQM) y que la cola de trabajos asociada a ese subsistema no está retenida. De forma predeterminada, el subsistema de IBM MQ y la cola de trabajos se llaman ambos QMQM en la biblioteca QMQM.

Acerca de esta tarea

Utilizar la línea de mandatos de IBM i para iniciar un gestor de colas

Procedimiento

1. Cree un gestor de colas local emitiendo el mandato CRTMQM desde una línea de mandatos de IBM i.

   Al crear un gestor de colas, tiene la opción de hacer que ese gestor de colas sea el gestor de colas predeterminado. El gestor de colas predeterminado (sólo puede haber uno) es aquel al que se aplica un mandato CL, si se omite el parámetro del nombre del gestor de colas (MQMNAME).
2. Inicie un gestor de colas local emitiendo el mandato STRMQM desde una línea de mandatos de IBM i.

   Si el gestor de colas tarda más de algunos segundos en reiniciarse, IBM MQ mostrará mensajes de estado de forma intermitente que detallan el progreso del inicio. Para obtener más información sobre estos mensajes, consulte Mensajes y códigos de razón.

Qué hacer a continuación

Puede detener un gestor de colas emitiendo el mandato ENDMQM desde la línea de mandatos de IBM i, y controlar un gestor de colas emitiendo otros mandatos de IBM MQ desde una línea de mandatos de IBM i.

Los gestores de colas remotos no se pueden iniciar de forma remota, sino que han de ser los operadores locales quienes los creen y los inicien en sus sistemas. Una excepción a esta regla es cuando existen recursos de operación remota (externos a IBM MQ for IBM i) que permiten habilitar estas operaciones.

El administrador de colas local no puede detener un gestor de colas remoto.

Nota: Como parte de la desactivación temporal de un sistema IBM MQ, tiene que desactivar temporalmente los gestores de colas activos. Este tema se describe en el apartado "Desactivación temporal de IBM MQ for IBM i" en la página 457.

IBM i Crear objetos de IBM MQ for IBM i

Utilice esta información para comprender los métodos para crear objetos de IBM MQ para IBM i.

Antes de empezar

Las siguientes tareas sugieren diversas maneras de utilizar IBM MQ for IBM i desde la línea de mandatos.

Acerca de esta tarea

Hay dos métodos en línea para crear objetos de IBM MQ, que son:

Procedimiento

1. Utilizando un mandato Create, por ejemplo: el mandato **Create MQM Queue** : **CRTMQMQ**
2. Utilizando un mandato Trabajar con objeto MQM, seguido de F6, por ejemplo: El mandato **Work with MQM Queues** : **WRKMQMQ**

Qué hacer a continuación

Para obtener una lista de todos los mandatos, consulte Mandatos CL de IBM MQ for IBM i.

Nota: Todos los mandatos de MQM se pueden emitir desde el menú Mandatos de gestor de colas de mensajes. Para visualizar este menú, escriba `GO CMDMQM` en la línea de mandatos y pulse la tecla `Enter` .

El sistema muestra automáticamente el panel de solicitud cuando se selecciona un mandato de este menú. Para visualizar el panel de solicitud para un mandato que ha escrito directamente en la línea de mandatos, pulse F4 antes de pulsar la tecla `Enter` .

Crear una cola local mediante el mandato CRTMQMQ

Procedimiento

1. Escriba `CHGMQM` en la línea de mandatos y pulse la tecla F4.
2. En el panel **Crear cola MQM**, escriba el nombre de la cola que desea crear en el campo `Queue name` . Si desea especificar un nombre con una combinación de mayúsculas/minúsculas, encierre el nombre entre comillas simples.
3. Escriba `*LCL` en el campo `Queue type` .
4. Especifique un nombre de gestor de colas, a menos que esté utilizando el gestor de colas predeterminado, y pulse la tecla `Enter` . Puede sobrescribir cualquiera de los valores con un nuevo valor. Desplácese hacia delante para ver más campos. Las opciones usadas para los clústeres están al final de la lista de opciones.
5. Cuando haya cambiado algún valor, pulse la tecla `Enter` para crear la cola.

Crear una cola local mediante el mandato WRKMQMQ

Procedimiento

1. Escriba `WRKMQMQ` en la línea de mandatos.
2. Entre el nombre de un gestor de colas.
3. Si desea visualizar el panel de solicitud, pulse F4. El panel de solicitud es útil para reducir el número de colas visualizadas, especificando para ello un nombre de cola o un tipo de cola genérico.
4. Pulse `Enter` y se mostrará el **panel Trabajar con colas MQM** . Puede sobrescribir cualquiera de los valores con un nuevo valor. Desplácese hacia delante para ver más campos. Las opciones usadas para los clústeres están al final de la lista de opciones.
5. Pulse F6 para crear una cola nueva; aparecerá el panel **CRTMQMQ**. Consulte “Crear una cola local mediante el mandato CRTMQMQ” en la página 387 para obtener instrucciones sobre cómo crear la cola. Cuando haya creado la cola, vuelve a aparecer el panel **Trabajar con colas MQM**. La nueva cola se añade a la lista cuando pulsa `F5=Refresh`.

Modificación de los atributos del gestor de colas

Acerca de esta tarea

Para modificar los atributos del gestor de colas especificado en el mandato **CHGMQM**, indique los atributos y los valores que desea cambiar. Por ejemplo, para modificar los atributos de `jupiter.queue.manager`, se utilizan estas opciones:

Procedimiento

Escriba **CHGMQM** en la línea de mandatos y pulse la tecla F4.

Resultados

El mandato cambia la cola de mensajes no entregados utilizada y habilita la función de inhibir sucesos.

IBM i Utilización de colas locales en IBM i

Esta sección contiene ejemplos de algunos de los mandatos que se pueden usar para gestionar colas locales. Todos los mandatos que se muestran también están disponibles utilizando las opciones del **panel de mandatos WRKMQMQ**.

Definición de una cola local

Para una aplicación, el gestor de colas local es el gestor de colas al que está conectada la aplicación. Las colas gestionadas por el gestor de colas local son locales con respecto a ese gestor de colas.

Utilice el mandato **CRTMQMQ QTYPE *LCL** para crear una definición de una cola local y también para crear la estructura de datos llamada cola. Además, puede modificar las características de la cola para que sean distintas de las de la cola local predeterminada.

En este ejemplo, la cola que definimos, `orange.local.queue`, se ha especificado para que tenga estas características:

- Está habilitada para las operaciones de obtención, inhabilitada para las operaciones de transferencia y funciona de acuerdo a primero en entrar, primero en salir (FIFO).
- Es una cola *corriente*, es decir, no es una cola de inicio ni una cola de transmisión, ni tampoco genera mensajes desencadenantes.
- La profundidad máxima de la cola es de 1000 mensajes; la longitud máxima de mensaje es de 2000 bytes.

Esto se logra con el siguiente mandato en el gestor de colas predeterminado:

```
CRTMQMQ QNAME('orange.local.queue') QTYPE(*LCL)
TEXT('Queue for messages from other systems')
PUTENBL(*NO)
GETENBL(*YES)
TRGENBL(*NO)
MSGDLYSEQ(*FIFO)
MAXDEPTH(1000)
MAXMSGLEN(2000)
USAGE(*NORMAL)
```

Nota:

1. USAGE *NORMAL indica que esta cola no es una cola de transmisión.
2. Si ya hay una cola local con el nombre `orange.local.queue` en el mismo gestor de colas, este mandato fallará. Utilice el atributo REPLACE *YES si desea escribir encima de la definición ya existente de una cola, pero consulte también el apartado "Modificación de atributos de cola local" en la página 389.

Definición de una cola de mensajes no entregados

Cada gestor de colas debe tener una cola local que se utilizará como cola de mensajes no entregados, con el fin de que los mensajes que no se puedan entregar en su destino correcto se almacenen para recuperación posterior. La cola de mensajes no entregados se debe indicar explícitamente en el gestor de colas. Para ello, especifique una cola de mensajes no entregados en el mandato **CRTMQM** o bien utilice el mandato **CHGMQM** para especificar una más adelante. Para poder utilizar la cola de mensajes no entregados, primero es preciso definirla.

Junto con el producto se suministra una cola de mensajes no entregados de ejemplo denominada SYSTEM.DEAD.LETTER.QUEUE. Esta cola se crea automáticamente cuando se crea el gestor de colas. Si es necesario, puede modificar esta definición. No es preciso cambiar su nombre, aunque puede hacerlo si lo desea.

Una cola de mensajes no entregados no tiene ningún requisito especial excepto que:

- Debe ser una cola local.
- Su atributo MAXMSGL (longitud máxima de mensajes) debe permitir que la cola pueda alojar los mensajes más grandes que el gestor de colas tenga que manejar **más** el tamaño de la cabecera de mensaje no entregado (MQDLH).

IBM MQ proporciona un manejador de la cola de mensajes no entregados que le permite especificar cómo se han de procesar o eliminar los mensajes encontrados en una cola de mensajes no entregados. Para obtener más información, consulte "Invocación del manejador de cola de mensajes no entregados en IBM i" en la página 169.

Visualización de los atributos de objeto predeterminados

Cuando define un objeto de IBM MQ, éste toma del objeto predeterminado todos los atributos que no especifique. Por ejemplo, cuando define una cola local, la cola hereda todos los atributos que omita en la definición de la cola local predeterminada, que tiene el nombre SYSTEM.DEFAULT.LOCAL.QUEUE. Si desea saber cuáles son exactamente esos atributos, utilice este mandato:

```
DSPMQMQ QNAME(SYSTEM.DEFAULT.LOCAL.QUEUE) MQMNAME(MYQUEUEMANAGER)
```

Copiar una definición de cola local

Cuando desee copiar una definición de cola local, utilice el mandato CPYMQMQ. Por ejemplo:

```
CPYMQMQ FROMQ('orange.local.queue') TOQ('magenta.queue') MQMNAME(MYQUEUEMANAGER)
```

Este mandato crea una cola que tiene los mismos atributos que nuestra cola `orange.local.queue`, en vez de tener los atributos de la cola local predeterminada del sistema.

También puede utilizar el mandato **CPYMQMQ** para copiar una definición de cola, sustituyendo uno o varios cambios realizados en los atributos de la original. Por ejemplo:

```
CPYMQMQ FROMQ('orange.local.queue') TOQ('third.queue') MQMNAME(MYQUEUEMANAGER)
MAXMSGLEN(1024)
```

Este mandato copia los atributos de la cola `orange.local.queue` en la cola `third.queue`, pero especifica que la longitud máxima de mensaje de la nueva cola ha de ser 1024 bytes, en lugar de 2000.

Nota: Cuando se utiliza el mandato **CPYMQMQ**, sólo se copian los atributos de la cola, no los mensajes de la cola.

Modificación de atributos de cola local

Los atributos de las colas se pueden cambiar de dos formas: mediante el mandato **CHGMQMQ** o bien el mandato **CPYMQMQ** con el atributo REPLACE *YES. En "Definición de una cola local" en la página 388, ha definido la cola `orange.local.queue`. Si, por ejemplo, tiene que aumentar la longitud máxima de mensajes en esta cola a 10.000 bytes.

- Utilizando el mandato **CHGMQMQ**:

```
CHGMQMQ QNAME('orange.local.queue') MQMNAME(MYQUEUEMANAGER) MAXMSGLEN(10000)
```

Este mandato modifica un solo atributo, el de la longitud máxima del mensaje; todos los demás atributos no cambian.

- Si se utiliza el mandato **CRTMQMQ** con la opción REPLACE *YES, por ejemplo:

```
CRTMQMQ QNAME('orange.local.queue') QTYPE(*LCL) MQMNAME(MYQUEUEMANAGER)
MAXMSGLEN(10000) REPLACE(*YES)
```

Este mandato no sólo cambia la longitud máxima de mensaje sino también los demás atributos, a los que se asignan sus valores predeterminados. La cola está ahora habilitada para operaciones de transferencia mientras que anteriormente estaba inhibida para dichas operaciones. El valor predeterminado es la transferencia habilita, según lo especificado por la cola SYSTEM.DEFAULT.LOCAL.QUEUE, a menos que se haya cambiado ese valor.

Si ***reduce*** la longitud máxima del mensaje en una cola existente, los mensajes existentes no se ven afectados. Sin embargo, todos los mensajes nuevos deben cumplir los nuevos criterios.

Vaciar una cola local

Para suprimir todos los mensajes de una cola local denominada magenta.queue, utilice el mandato siguiente:

```
CLRMQMQ QNAME('magenta.queue') MQMNAME(MYQUEUEMANAGER)
```

Una cola no se puede vaciar si:

- Hay mensajes no confirmados que se han transferido a la cola bajo punto de sincronización.
- Hay una aplicación que tiene abierta actualmente la cola.

Suprimir una cola local

Utilice el mandato **DLTMQMQ** para suprimir una cola local.

No se puede suprimir una cola si en ella hay mensajes no confirmados, o si se está utilizando.

Habilitar colas grandes

IBM MQ da soporte a colas de más de 2 GB. Consulte la documentación del sistema operativo para obtener información sobre cómo permitir que IBM i dé soporte a archivos grandes.

La información del producto IBM i se puede encontrar en IBM Documentation.

Es posible que algunos programas de utilidad no puedan manejar archivos de más de 2 GB. Antes de habilitar el soporte para archivos grandes, consulte la documentación de su sistema operativo para obtener información sobre las limitaciones de este soporte.

IBM i Utilización de colas de alias en IBM i

Esta sección contiene ejemplos de algunos de los mandatos que se pueden usar para gestionar colas alias. Todos los mandatos que se muestran también están disponibles utilizando las opciones del **panel de mandatos WRKMQMQ**.

Una cola alias (a veces conocida como alias de cola) proporciona un método para redirigir las llamadas MQI. La cola alias no es una cola real, sino una definición que se resuelve en una cola real. La definición de cola alias contiene el nombre de una cola destino, que se especifica mediante el atributo TGTQNAME.

Cuando una aplicación especifica una cola alias en una llamada MQI, el gestor de colas resuelve el nombre de cola real en el momento de la ejecución.

Por ejemplo, supongamos que se ha desarrollado una aplicación para transferir los mensajes a una cola llamada my.alias.queue. La aplicación especifica el nombre de esta cola cuando realiza una solicitud **MQOPEN** e, indirectamente, si transfiere un mensaje a esta cola. La aplicación no sabe que la cola es una

cola alias. Para cada llamada MQI que utilice este alias, el gestor de colas determina el nombre de cola real, que podría ser una cola local o una cola remota definida en este gestor de colas.

Si cambia el valor del atributo TGTQNAME, puede redirigir las llamadas MQI a otra cola, posiblemente situada en otro gestor de colas. Esto resulta de gran utilidad para el mantenimiento, la migración y el equilibrio de la carga.

Definición de una cola alias

El mandato siguiente crea una cola alias:

```
CRTMQMQ QNAME('my.alias.queue') QTYPE(*ALS) TGTQNAME('yellow.queue')
MQMNAME(MYQUEUEMANAGER)
```

Este mandato redirige las llamadas MQI que especifican `my.alias.queue` a la cola `yellow.queue`. El mandato no crea la cola de destino; las llamadas MQI fallan si la cola `yellow.queue` no existe en el momento de la ejecución.

Si cambia la definición del alias, puede redirigir las llamadas MQI a otra cola. Por ejemplo:

```
CHGMQMQ QNAME('my.alias.queue') TGTQNAME('magenta.queue') MQMNAME(MYQUEUEMANAGER)
```

Este mandato redirige las llamadas MQI a otra cola, `magenta.queue`.

También puede utilizar colas alias para que hacer que una sola cola (la cola destino) parezca tener atributos distintos para aplicaciones distintas. Esto se hace definiendo dos alias, uno para cada aplicación. Suponga que tiene dos aplicaciones:

- La aplicación ALPHA puede transferir mensajes a `yellow.queue`, pero no puede obtener mensajes de ella.
- La aplicación BETA puede obtener mensajes de `yellow.queue`, pero no tiene autorización para transferir mensajes a ella.

Para lograrlo, utilice estos mandatos:

```
/* This alias is put enabled and get disabled for application ALPHA */

CRTMQMQ QNAME('alphas.alias.queue') QTYPE(*ALS)     TGTQNAME('yellow.queue')
PUTENBL(*YES) GETENBL(*NO) MQMNAME(MYQUEUEMANAGER)

/* This alias is put disabled and get enabled for application BETA */

CRTMQMQ QNAME('betas.alias.queue') QTYPE(*ALS) TGTQNAME('yellow.queue')
PUTENBL(*NO) GETENBL(*YES) MQMNAME(MYQUEUEMANAGER)
```

ALPHA utiliza el nombre de la cola `alphas.alias.queue` en las llamadas MQI ; BETA utiliza el nombre de la cola `betas.alias.queue`. Ambas aplicaciones acceden a la misma cola, pero de forma diferente.

Cuando define las colas alias, puede utilizar el atributo REPLACE *YES, de la misma manera que cuando lo utiliza con las colas locales.

Utilización de otros mandatos con colas alias

Puede utilizar los correspondientes mandatos para visualizar o cambiar los atributos de colas alias. Por ejemplo:

```
* Display the alias queue's attributes */

DSPMQMQ QNAME('alphas.alias.queue') MQMNAME(MYQUEUEMANAGER)

/* ALTER the base queue name, to which the alias resolves. */
/* FORCE = Force the change even if the queue is open. */
```

```
CHQMQMQ QNAME('alphas.alias.queue') TGTQNAME('orange.local.queue') FORCE(*YES)
MQMNAME(MYQUEUEMANAGER)
```

IBM i Trabajar con colas de modelo en IBM i

Esta sección contiene ejemplos de algunos de los mandatos que se pueden usar para gestionar colas modelo. Todos los mandatos que se muestran también están disponibles utilizando las opciones del **panel de mandatos WRKMQMQ**.

Un gestor de colas crea una cola dinámica si recibe una llamada MQI de una aplicación que especifica un nombre de cola que se ha definido como cola modelo. El nombre de la nueva cola dinámica lo genera el gestor de colas cuando se crea la cola. Una cola modelo es una plantilla que especifica los atributos de las colas dinámicas que se creen a partir de ella.

Las colas modelo proporcionan a las aplicaciones un método práctico de crear colas cuando se necesitan.

Definición de una cola modelo

Las colas modelo se definen con un conjunto de atributos del mismo modo en que se define una cola local. Las colas modelo y las colas locales tienen el mismo conjunto de atributos, excepto que en las colas modelo se puede especificar si las colas dinámicas creadas son temporales o persistentes. (Las colas permanentes se conservan tras reiniciar el gestor de colas, las colas temporales no). Por ejemplo:

```
CRTMQMQ QNAME('green.model.queue') QTYPE(*MDL) DFNTYPE(*PERMDYN)
```

Este mandato crea una definición de cola modelo. Desde el atributo DFNTYPE, las colas reales creadas a partir de esta plantilla son colas dinámicas permanentes. Los atributos no especificados se copian automáticamente de la cola predeterminada SYSYTEM.DEFAULT.MODEL.QUEUE.

Cuando defina las colas modelo, puede utilizar el atributo REPLACE *YES, de la misma manera que cuando lo utiliza con las colas locales.

Utilización de otros mandatos con colas modelo

Puede utilizar los correspondientes mandatos para visualizar o alterar los atributos de una cola modelo. Por ejemplo:

```
/* Display the model queue's attributes */

DSPMQMQ MQMNAME(MYQUEUEMANAGER) QNAME('green.model.queue')

/* ALTER the model queue to enable puts on any */
/* dynamic queue created from this model. */

CHGMQMQ MQMNAME(MYQUEUEMANAGER) QNAME('blue.model.queue') PUTENBL(*YES)
```

IBM i Utilización del desencadenamiento en IBM i

Utilice esta información para aprender sobre el desencadenamiento y las definiciones de proceso.

IBM MQ proporciona un recurso para iniciar una aplicación automáticamente cuando se cumplen ciertas condiciones en una cola. Un ejemplo de estas condiciones es cuando el número de mensajes en una cola alcanza un número especificado. Este recurso se denomina *desencadenamiento* y se describe detalladamente en Desencadenamiento de canales.

¿Qué son los desencadenantes?

El gestor de colas define ciertas condiciones como sucesos desencadenantes constituyentes. Si se habilita el desencadenamiento para una cola y se produce un suceso desencadenante, el gestor de colas envía un mensaje desencadenante a una cola llamada cola de inicio. La presencia del mensaje desencadenante en la cola de inicio indica que se ha producido un suceso desencadenante.

Los mensajes desencadenantes generados por el gestor de colas no son persistentes. Como consecuencia, se reducen las anotaciones cronológicas (lo que mejora el rendimiento), y se minimizan los duplicados durante el reinicio, de manera que se mejora el tiempo de reinicio.

¿Qué es el supervisor de desencadenantes?

El programa que procesa la cola de inicio se denomina aplicación supervisora desencadenante y su función consiste en leer el mensaje desencadenante y realizar las acciones adecuadas, basándose en la información contenida en el mensaje desencadenante. Normalmente esta acción consistiría en iniciar alguna otra aplicación para procesar la cola que ha causado la generación del mensaje desencadenante. Desde el punto de vista del gestor de colas, la aplicación supervisora desencadenante no tiene ninguna particularidad especial, se trata de otra aplicación que lee mensajes de una cola (la cola de inicio).

Modificación de los atributos de sometimiento de trabajos del supervisor de desencadenantes

El supervisor desencadenante que se suministra como mandato **STRMQMTRM** somete un trabajo para cada mensaje desencadenante utilizando la descripción de trabajo predeterminada del sistema, QDFTJOBD. Esto tiene algunas limitaciones, ya que los trabajos sometidos siempre se llaman QDFTJOBD y tienen los atributos de la descripción de trabajo predeterminada, incluida la lista de bibliotecas *SYSVAL. IBM MQ proporciona un método para alterar temporalmente estos atributos. Por ejemplo, es posible personalizar los trabajos sometidos para que tengan nombres más significativos, tal como se indica a continuación:

1. En la descripción del trabajo, especifique la descripción que desee, por ejemplo los valores de las anotaciones cronológicas.
2. Especifique los Datos de entorno de la definición de proceso utilizada en el proceso desencadenante:

   ```
   CHGMQMPRC PRCNAME(MY_PROCESS) MQMNAME(MHA3) ENVDATA ('JOBD(MYLIB/TRIGJOBD)')
   ```

   El Supervisor desencadenante realiza un SBMJOB utilizando la descripción especificada.

Es posible alterar temporalmente otros atributos del SBMJOB especificando la palabra clave y el valor adecuados en los Datos de entorno de la definición de proceso. La única excepción a esta regla es la palabra clave CMD, porque en este caso el supervisor de desencadenantes completa el atributo. A continuación se muestra un ejemplo del mandato para especificar los Datos de entorno de la definición de proceso, en el que se van a modificar el nombre de trabajo y la descripción:

```
CHGMQMPRC PRCNAME(MY_PROCESS) MQMNAME(MHA3) ENVDATA ('JOBD(MYLIB/TRIGJOB)
JOB(TRIGGER)')
```

Definición de una cola de aplicación para el desencadenamiento

Una cola de aplicación es una cola local que las aplicaciones utilizan para el envío de mensajes, mediante la MQI. El desencadenamiento requiere definir varios atributos de cola en la cola de aplicación. El propio desencadenamiento está habilitado mediante el atributo TRGENBL.

En este ejemplo, se ha de generar un suceso desencadenante cuando haya 100 mensajes de prioridad 5 o superior en la cola local `motor.insurance.queue`, tal como se indica a continuación:

```
CRTMQMQ MQMNAME(MYQUEUEMANAGER) QNAME('motor.insurance.queue') QTYPE(*LCL)
PRCNAME('motor.insurance.quote.process') MAXMSGLEN(2000)
DFTMSGPST(*YES) INITQNAME('motor.ins.init.queue')
TRGENBL(*YES) TRGTYPE(*DEPTH) TRGDEPTH(100) TRGMSGPTY(5)
```

donde los parámetros son:

MQMNAME(MYQUEUEMANAGER)
Nombre del gestor de colas.

QNAME('motor.insurance.queue')
El nombre de la cola de aplicación que se está definiendo.

PRCNAME('motor.insurance.quote.process')
El nombre de la aplicación que el programa supervisor desencadenante debe iniciar.

MAXMSGLEN(2000)
La longitud máxima de los mensajes de la cola.

DFTMSGPST(*YES)
Los mensajes de esta cola son persistentes de forma predeterminada.

INITQNAME('motor.ins.init.queue')
El nombre de la cola de inicio en la que el gestor de colas va a poner el mensaje desencadenante.

TRGENBL(*YES)
El valor del atributo de desencadenamiento.

TRGTYPE(*DEPTH)
Se genera un suceso desencadenante cuando el número de mensajes de la prioridad requerida (**TRGMSGPTY**) alcanza el número especificado en **TRGDEPTH**.

TRGDEPTH(100)
El número de mensajes necesarios para generar un suceso desencadenante.

TRGMSGPTY(5)
La prioridad de los mensajes que el gestor de colas va a contar para decidir si debe generar un suceso desencadenante. Sólo se incluyen en el recuento los mensajes con prioridad 5 o superior.

Definición de una cola de inicio

Cuando se produce un suceso desencadenante, el gestor de colas coloca un mensaje desencadenante en la cola de inicio especificada en la definición de la cola de aplicación. Las colas de inicio no tienen ningún valor especial, pero la siguiente definición de la cola local `motor.ins.init.queue` le puede servir de orientación:

```
CRTMQMQ MQMNAME(MYQUEUEMANAGER) QNAME('motor.ins.init.queue') QTYPE(*LCL)
GETENBL(*YES) SHARE(*NO) TRGTYPE(*NONE)
MAXMSGL(2000)
MAXDEPTH(1000)
```

Crear una definición de proceso

Para crear una definición de proceso, utilice el mandato **CRTMQMPRC**. Una definición de proceso asocia una cola de aplicación a la aplicación que va a procesar los mensajes de la cola. Esto se lleva a cabo mediante el atributo PRCNAME en la cola de la aplicación `motor.insurance.queue`. El mandato siguiente crea el proceso necesario, `motor.insurance.quote.process`, identificado en este ejemplo:

```
CRTMQMPRC MQMNAME(MYQUEUEMANAGER) PRCNAME('motor.insurance.quote.process')
TEXT('Insurance request message processing')
APPTYPE(*OS400) APPID(MQTEST/TESTPROG)
USRDATA('open, close, 235')
```

donde los parámetros son:

MQMNAME(MYQUEUEMANAGER)
Nombre del gestor de colas.

PRCNAME('motor.insurance.quote.process')
El nombre de la definición de proceso.

TEXT('Insurance request message processing')
Una descripción del programa de aplicación relacionado con esta definición. Este texto se visualiza al utilizar el mandato **DSPMQMPRC**. sirve para ayudarle a identificar lo que hace el proceso. Si utiliza espacios en la serie, debe colocarla entre comillas simples.

APPTYPE(*OS400)
El tipo de aplicación que se ha de iniciar.

APPID(MQTEST/TESTPROG)
El nombre del archivo ejecutable de la aplicación, especificado como nombre de archivo totalmente calificado.

USRDATA('open, close, 235')
Datos definidos por el usuario, que la aplicación puede utilizar.

Visualizar la definición de proceso

El mandato **DSPMQMPRC** le permite examinar los resultados de la definición. Por ejemplo:

```
MQMNAME(MYQUEUEMANAGER) DSPMQMPRC('motor.insurance.quote.process')
```

También puede utilizar el mandato **CHGMQMPRC** para modificar una definición de proceso existente, y el mandato **DLTMQMPRC** para suprimir una definición de proceso.

IBM i Comunicación entre dos sistemas IBM MQ en IBM i

Este ejemplo de código muestra cómo configurar dos sistemas IBM MQ for IBM i, utilizando mandatos CL, para que puedan comunicarse entre sí.

Los sistemas se llaman SYSTEMA y SYSTEMB, y se utiliza el protocolo de comunicaciones TCP/IP.

Lleve a cabo el procedimiento siguiente:

1. Cree un gestor de colas en SYSTEMA, y llámelo QMGRA1.

```
CRTMQM   MQMNAME(QMGRA1) TEXT('System A - Queue +
Manager 1') UDLMSGQ(SYSTEM.DEAD.LETTER.QUEUE)
```

2. Inicie este gestor de colas.

```
STRMQM   MQMNAME(QMGRA1)
```

3. Defina los objetos IBM MQ en SYSTEMA que necesita para enviar mensajes a un gestor de colas en SYSTEMB.

```
/* Transmission queue */
CRTMQMQ  QNAME(XMITQ.TO.QMGRB1) QTYPE(*LCL) +
MQMNAME(QMGRA1) TEXT('Transmission Queue +
to QMGRB1') MAXDEPTH(5000) USAGE(*TMQ)

/* Remote queue that points to a queue called TARGETB        */
/* TARGETB belongs to queue manager QMGRB1 on SYSTEMB        */
CRTMQMQ  QNAME(TARGETB.ON.QMGRB1) QTYPE(*RMT) +
MQMNAME(QMGRA1) TEXT('Remote Q pointing +
at Q TARGETB on QMGRB1 on Remote System +
SYSTEMB') RMTQNAME(TARGETB) +
RMTMQMNAME(QMGRB1) TMQNAME(XMITQ.TO.QMGRB1)

/* TCP/IP sender channel to send messages to the queue manager on SYSTEMB*/
CRTMQMCHL CHLNAME(QMGRA1.TO.QMGRB1) CHLTYPE(*SDR) +
MQMNAME(QMGRA1) TRPTYPE(*TCP) +
TEXT('Sender Channel From QMGRA1 on +
SYSTEMA to QMGRB1 on SYSTEMB') +
CONNAME(SYSTEMB) TMQNAME(XMITQ.TO.QMGRB1)
```

4. Cree un gestor de colas en SYSTEMB, y llámelo QMGRB1.

```
CRTMQM   MQMNAME(QMGRB1) TEXT('System B - Queue +
Manager 1') UDLMSGQ(SYSTEM.DEAD.LETTER.QUEUE)
```

5. Inicie el gestor de colas en SYSTEMB.

```
STRMQM   MQMNAME(QMGRB1)
```

6. Defina los objetos IBM MQ que necesita para recibir mensajes del gestor de colas en SYSTEMA.

```
/* Local queue to receive messages on */
CRTMQMQ  QNAME(TARGETB) QTYPE(*LCL) MQMNAME(QMGRB1) +
TEXT('Sample Local Queue for QMGRB1')

/* Receiver channel of the same name as the sender channel on SYSTEMA */
CRTMQMCHL CHLNAME(QMGRA1.TO.QMGRB1) CHLTYPE(*RCVR) +
MQMNAME(QMGRB1) TRPTYPE(*TCP) +
TEXT('Receiver Channel from QMGRA1 to +
QMGRB1')
```

7. Por último, inicie un escucha TCP/IP en SYSTEMB para que se pueda iniciar el canal. En este ejemplo se utiliza el puerto predeterminado 1414.

```
STRMQMLSR MQMNAME(QMGRB1)
```

Ahora ya está preparado para enviar mensajes de prueba entre SYSTEMA y SYSTEMB. Utilizando uno de los ejemplos suministrados, ponga una serie de mensajes en la cola remota situada en SYSTEMA.

Inicie el canal en SYSTEMA, ya sea mediante el mandato **STRMQMCHL** o bien mediante el mandato **WRKMQMCHL** y entrando una solicitud de inicio (Opción 14) para el canal emisor.

El canal debe pasar al estado RUNNING (en ejecución) y los mensajes se envían a la cola TARGETB situada en SYSTEMB.

Para comprobar los mensajes, emita el mandato:

```
WRKMQMMSG QNAME(TARGETB) MQMNAME(QMGRB1).
```

IBM i Definiciones de recurso de ejemplo en IBM i

Este ejemplo contiene el programa CL de ejemplo de IBM i AMQSAMP4.

```
/*********************************************************************/
/*                                                                   */
/* Program name: AMQSAMP4                                            */
/*                                                                   */
/* Description: Sample CL program defining MQM queues                */
/*       to use with the sample programs                             */
/*       Can be run, with changes as needed, after                   */
/*       starting the MQM                                            */
/*                                                                   */
/* <N_OCO_COPYRIGHT>                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* 63H9336                                                           */
/* (c) Copyright IBM Corp. 1993, 2024. All Rights Reserved.          */
/*                                                                   */
/* US Government Users Restricted Rights - Use, duplication or       */
/* disclosure restricted by GSA ADP Schedule Contract with           */
/* IBM Corp.                                                         */
/* <NOC_COPYRIGHT>                                                 */
/*                                                                   */
/*********************************************************************/
/*                                                                   */
/* Function:                                                         */
/*                                                                   */
/*                                                                   */
/*  AMQSAMP4 is a sample CL program to create or reset the           */
/*  MQI resources to use with the sample programs.                   */
/*                                                                   */
/*  This program, or a similar one, can be run when the MQM          */
/*  is started - it creates the objects if missing, or resets        */
```

```
/*  their attributes to the prescribed values.                    */
/*                                                                */
/*                                                                */
/*                                                                */
/*                                                                */
/*  Exceptions signaled: none                                     */
/*  Exceptions monitored: none                                    */
/*                                                                */
/*  AMQSAMP4 takes a single parameter, the Queue Manager name     */
/*                                                                */
/******************************************************************/
QSYS/PGM PARM(&QMGRNAME)

/******************************************************************/
/*  Queue Manager Name Parameter                                  */
/******************************************************************/
QSYS/DCL VAR(&QMGRNAME) TYPE(*CHAR)

/******************************************************************/
/*    EXAMPLES OF DIFFERENT QUEUE TYPES                           */
/*                                                                */
/*  Create local, alias and remote queues                         */
/*                                                                */
/*  Uses system defaults for most attributes                      */
/*                                                                */
/******************************************************************/
/*  Create a local queue   */
CRTMQMQ   QNAME('SYSTEM.SAMPLE.LOCAL')        +
MQMNAME(&QMGRNAME)             +
QTYPE(*LCL) REPLACE(*YES)         +
+
TEXT('Sample local queue') /* description */+
SHARE(*YES)        /* Shareable  */+
DFTMSGPST(*YES) /* Persistent messages OK */

/*  Create an alias queue  */
CRTMQMQ   QNAME('SYSTEM.SAMPLE.ALIAS')        +
MQMNAME(&QMGRNAME)             +
QTYPE(*ALS) REPLACE(*YES)         +
+
TEXT('Sample alias queue')         +
DFTMSGPST(*YES) /* Persistent messages OK */+
TGTQNAME('SYSTEM.SAMPLE.LOCAL')

/*  Create a remote queue - in this case, an indirect reference  */
/*   is made to the sample local queue on OTHER queue manager  */
CRTMQMQ   QNAME('SYSTEM.SAMPLE.REMOTE')        +
MQMNAME(&QMGRNAME)             +
QTYPE(*RMT) REPLACE(*YES)         +
+
TEXT('Sample remote queue')/* description */+
DFTMSGPST(*YES) /* Persistent messages OK */+
RMTQNAME('SYSTEM.SAMPLE.LOCAL')       +
RMTMQMNAME(OTHER)  /* Queue is on OTHER  */

/*  Create a transmission queue for messages to queues at OTHER  */
/*  By default, use remote node name                */
CRTMQMQ   QNAME('OTHER') /* transmission queue name */+
MQMNAME(&QMGRNAME)             +
QTYPE(*LCL) REPLACE(*YES) +
TEXT('Transmision queue to OTHER') +
USAGE(*TMQ)  /* transmission queue   */

/******************************************************************/
/*    SPECIFIC QUEUES AND PROCESS USED BY SAMPLE PROGRAMS         */
/*                                                                */
/*  Create local queues used by sample programs                   */
/*  Create MQI process associated with sample initiation queue    */
/*                                                                */
/******************************************************************/
/*  General reply queue  */
CRTMQMQ   QNAME('SYSTEM.SAMPLE.REPLY')        +
MQMNAME(&QMGRNAME)             +
QTYPE(*LCL) REPLACE(*YES)         +
+
TEXT('General reply queue')         +
DFTMSGPST(*NO) /* Not Persistent     */

/*  Queue used by AMQSINQ4                     */
CRTMQMQ   QNAME('SYSTEM.SAMPLE.INQ')         +
MQMNAME(&QMGRNAME)             +
QTYPE(*LCL) REPLACE(*YES)         +
```

```
+
TEXT('Queue for AMQSINQ4')         +
SHARE(*YES)        /* Shareable  */+
DFTMSGPST(*NO) /* Not Persistent     */+
+
TRGENBL(*YES) /* Trigger control on   */+
TRGTYPE(*FIRST)/* Trigger on first message*/+
PRCNAME('SYSTEM.SAMPLE.INQPROCESS')     +
INITQNAME('SYSTEM.SAMPLE.TRIGGER')

/*  Queue used by AMQSSET4                    */
CRTMQMQ   QNAME('SYSTEM.SAMPLE.SET')         +
MQMNAME(&QMGRNAME)             +
QTYPE(*LCL) REPLACE(*YES)         +
+
TEXT('Queue for AMQSSET4')         +
SHARE(*YES)        /* Shareable */ +
DFTMSGPST(*NO)/* Not Persistent     */ +
+
TRGENBL(*YES) /* Trigger control on   */ +
TRGTYPE(*FIRST)/* Trigger on first message*/+
PRCNAME('SYSTEM.SAMPLE.SETPROCESS')     +
INITQNAME('SYSTEM.SAMPLE.TRIGGER')

/*  Queue used by AMQSECH4                     */
CRTMQMQ   QNAME('SYSTEM.SAMPLE.ECHO')         +
MQMNAME(&QMGRNAME)             +
QTYPE(*LCL) REPLACE(*YES)         +
+
TEXT('Queue for AMQSECH4')         +
SHARE(*YES)        /* Shareable */ +
DFTMSGPST(*NO)/* Not Persistent     */ +
+
TRGENBL(*YES) /* Trigger control on   */ +
TRGTYPE(*FIRST)/* Trigger on first message*/+
PRCNAME('SYSTEM.SAMPLE.ECHOPROCESS')    +
INITQNAME('SYSTEM.SAMPLE.TRIGGER')

/*  Initiation Queue used by AMQSTRG4, sample trigger process   */
CRTMQMQ   QNAME('SYSTEM.SAMPLE.TRIGGER') +
MQMNAME(&QMGRNAME)             +
QTYPE(*LCL) REPLACE(*YES)  +
TEXT('Trigger queue for sample programs')

/*  MQI Processes associated with triggered sample programs    */
/*                                */
/*****  Note - there are versions of the triggered samples ******/
/*****   in different languages - set APPID for these   ******/
/*****   process to the variation you want to trigger   ******/
/*                                */
CRTMQMPRC  PRCNAME('SYSTEM.SAMPLE.INQPROCESS')     +
MQMNAME(&QMGRNAME)             +
REPLACE(*YES)               +
+
TEXT('Trigger process for AMQSINQ4')    +
ENVDATA('JOBPTY(3)') /* Submit parameter */ +
/** Select the triggered program here       **/        +
APPID('QMQM/AMQSINQ4')  /* C       +
/*    APPID('QMQM/AMQ0INQ4')  /* COBOL */    +
/*    APPID('QMQM/AMQ3INQ4')  /* RPG - ILE */

CRTMQMPRC  PRCNAME('SYSTEM.SAMPLE.SETPROCESS')     +
MQMNAME(&QMGRNAME)             +
REPLACE(*YES)               +
+
TEXT('Trigger process for AMQSSET4')    +
ENVDATA('JOBPTY(3)') /* Submit parameter */ +
/** Select the triggered program here       **/        +
APPID('QMQM/AMQSSET4')  /* C   */    +
/*    APPID('QMQM/AMQ0SET4')  /* COBOL */    +
/*    APPID('QMQM/AMQ3SET4')  /* RPG - ILE */

CRTMQMPRC  PRCNAME('SYSTEM.SAMPLE.ECHOPROCESS')    +
MQMNAME(&QMGRNAME)             +
REPLACE(*YES)               +
+
TEXT('Trigger process for AMQSECH4')    +
ENVDATA('JOBPTY(3)') /* Submit parameter */ +
/** Select the triggered program here       **/        +
APPID('QMQM/AMQSECH4')  /* C   */    +
/*    APPID('QMQM/AMQ0ECH4')  /* COBOL */    +
/*    APPID('QMQM/AMQ3ECH4')  /* RPG - ILE */
```

```
/*************************************************************************/
/*                                                                       */
/*  Normal return.                                                       */
/*                                                                       */
/*************************************************************************/
SNDPGMMSG  MSG('AMQSAMP4 Completed creating sample +
objects for ' *CAT &QMGRNAME)
RETURN
ENDPGM

/*************************************************************************/
/*                                                                       */
/* END OF AMQSAMP4                                                       */
/*                                                                       */
/*************************************************************************/
```

IBM i Formas alternativas de administrar IBM MQ for IBM i

El uso de mandatos CL es el método preferido de administración de IBM MQ for IBM i. Sin embargo, puede utilizar otros métodos de administración, incluidos los mandatos MQSC, los mandatos PCF, los mandatos de control y la administración remota.

Acerca de esta tarea

Normalmente se utilizan mandatos CL de IBM i para administrar IBM MQ for IBM i. Para obtener una visión general de estos mandatos, consulte "Gestión de IBM MQ for IBM i utilizando mandatos CL" en la página 385.

También puede utilizar mandatos MQSC y mandatos PCF tal como se describe en los subtemas, y puede utilizar mandatos de control tal como se describe en "Administración de IBM MQ for Multiplatforms utilizando mandatos de control" en la página 10.

Puede utilizar los sucesos de instrumentación de IBM MQ para supervisar el funcionamiento de los gestores de colas. Consulte Sucesos de instrumentación para obtener información sobre los sucesos de instrumentación de IBM MQ y cómo utilizarlos.

Utilice cualquiera de los métodos de administración descritos en los subtemas siguientes como alternativa a la utilización de mandatos CL de IBM i :

IBM i Administración local y remota en IBM i

Los objetos de IBM MQ for IBM i se pueden administrar de forma local o remota.

Acerca de esta tarea

La *administración local* consiste en llevar a cabo las tareas de administración en cualquier gestor de colas definido en el sistema local. En IBM MQ, esto puede considerarse como administración local porque no hay ningún canal IBM MQ implicado, es decir, la comunicación la gestiona el sistema operativo. Para realizar este tipo de tarea, hay que iniciar la sesión en el sistema remoto y ejecutar los mandatos desde allí o crear un proceso que pueda ejecutarlos automáticamente.

IBM MQ da soporte a la administración desde un único punto mediante lo que se conoce como *administración remota*. La administración remota consiste en enviar mensajes de control PCF (formato de mandato programable) a la cola `SYSTEM.ADMIN.COMMAND.QUEUE` situada en el gestor de colas de destino.

Hay varias maneras de generar mensajes PCF. Estos se describen en los pasos siguientes.

Procedimiento

- Escribir un programa utilizando mensajes PCF. Consulte "Administración utilizando mandatos PCF en IBM i" en la página 401.

- Escriba un programa utilizando la MQAI, que envía mensajes PCF. Consulte “Utilizar la MQAI para simplificar el uso de los PCF” en la página 38.
- Utilice el Explorador de IBM MQ , disponible con IBM MQ for Windows, que le permite utilizar una interfaz gráfica de usuario (GUI) y genera los mensajes PCF correctos. Consulte “Utilización de IBM MQ Explorer con IBM MQ for IBM i” en la página 402
- Utilizar **STRMQMMQSC** para enviar mandatos indirectamente a un gestor de colas remoto. Consulte “Administración utilizando mandatos MQSC en IBM i” en la página 400.

  Por ejemplo, puede emitir un mandato remoto para cambiar una definición de cola en un gestor de colas remoto.

  Algunos mandatos no se pueden emitir de esta manera, especialmente los que crean o inician gestores de colas y los que inician servidores de mandatos. Para realizar este tipo de tarea, hay que iniciar la sesión en el sistema remoto y ejecutar los mandatos desde allí o crear un proceso que pueda ejecutarlos automáticamente.

IBM i Administración utilizando mandatos MQSC en IBM i

En IBM i, cree una lista de mandatos en un archivo de script y, a continuación, ejecute el archivo utilizando el mandato **STRMQMMQSC** . Utilice mandatos MQSC para gestionar objetos del gestor de colas, incluidos el propio gestor de colas, colas, definiciones de proceso, listas de nombres, canales, canales de conexión de cliente, escuchas, servicios, temas y objetos de información de autenticación.

Acerca de esta tarea

Los mandatos de script de IBM MQ (MQSC) se escriben en formato legible, en texto EBCDIC. Los mandatos MQSC se emiten a un gestor de colas utilizando el mandato CL **STRMQMMQSC** IBM MQ . Éste es un método sólo de proceso por lotes, que toma su entrada de un archivo físico fuente en el sistema de bibliotecas de servidor. El nombre predeterminado de este archivo físico fuente es QMQSC.

Atención: No utilice la biblioteca QTEMP como biblioteca de origen para STRMQMMQSC, ya que el uso de la biblioteca QTEMP es limitado. Debe utilizar otra biblioteca como un archivo de entrada para el mandato.

Por motivos de portabilidad entre los entornos IBM MQ, limite la longitud de línea en los archivos de mandatos MQSC a 72 caracteres. Utilice el signo más para indicar que el mandato continúa en la siguiente línea.

Los atributos de objeto especificados en MQSC se muestran en este tema en mayúsculas (por ejemplo, RQMNAME), aunque no distinguen entre mayúsculas y minúsculas.

Nota:

1. El formato de un archivo MQSC no depende de su ubicación en el sistema de archivos.
2. Los nombres de los atributos de MQSC pueden tener ocho caracteres como máximo.
3. Los mandatos MQSC están disponibles en todas las plataformas IBM MQ .

Si desea una descripción de cada mandato MQSC y su sintaxis, consulte Mandatos MQSC.

Procedimiento

1. Cree el archivo de origen QMQSC.

   IBM MQ for IBM i no proporciona un archivo de origen denominado QMQSC. Para procesar mandatos MQSC, debe crear el archivo de origen QMQSC en una biblioteca de su elección, emitiendo el mandato siguiente:

   ```
   CRTSRCPF FILE(MYLIB/QMQSC) RCDLEN(240) TEXT('IBM MQ - MQSC Source')
   ```

2. Trabajar con los miembros.

El código fuente de MQSC se guarda en los miembros de este archivo fuente. Para trabajar con los miembros, entre el siguiente mandato:

```
WRKMBRPDM MYLIB/QMQSC
```

Ahora puede añadir nuevos miembros y mantener los existentes.

```
.
.
DEFINE QLOCAL(ORANGE.LOCAL.QUEUE) REPLACE +
DESCR(' ') +
PUT(ENABLED) +
DEFPRTY(0) +
DEFPSIST(NO) +
GET(ENABLED) +
MAXDEPTH(5000) +
MAXMSGL(1024) +
DEFSOPT(SHARED) +
NOHARDENBO +
USAGE(NORMAL) +
NOTRIGGER;
.
.
```

Figura 20. Extraer de un archivo de mandatos MQSC, `myprog.in`, que muestra un mandato MQSC (DEFINE QLOCAL) con sus atributos.

Información relacionada

Administración de IBM MQ utilizando mandatos MQSC

IBM i Administración utilizando mandatos PCF en IBM i

La finalidad de los mandatos PCF (formato de mandato programable) de IBM MQ es permitir que las tareas de administración se puedan programar en un programa de administración. De este modo, puede crear colas y definiciones de proceso y cambiar gestores de colas desde un programa.

Los mandatos PCF abarcan el mismo tipo de funciones que las proporcionadas por los mandatos MQSC. Sin embargo, a diferencia de los mandatos MQSC, los mandatos PCF y sus respuestas no están en un formato de texto legible para el usuario.

Puede escribir un programa que emita mandatos PCF a cualquier gestor de colas de la red desde un solo nodo. De este modo, puede centralizar y automatizar las tareas de administración.

Cada mandato PCF es una estructura de datos que se incluye en la parte de datos de aplicación de un mensaje de IBM MQ. El mandato se envía al gestor de colas de destino utilizando la función MQPUT de MQI, igual que cualquier otro mensaje. El servidor de mandatos situado en el gestor de colas que recibe el mensaje lo interpreta como un mensaje de mandato y ejecuta el mandato. Para obtener las respuestas, la aplicación emite una llamada MQGET y los datos de respuesta se devuelven en otra estructura de datos. La aplicación puede entonces procesar la respuesta y actuar en conformidad.

En resumen, estos son algunos de los elementos que un programador de aplicaciones debe especificar para crear un mensaje de mandato PCF:

Descriptor de mensaje

Es un descriptor de mensaje estándar de IBM MQ, en el que:

- El tipo de mensaje (*MsgType*) es MQMT_REQUEST.
- El formato del mensaje (*Format*) es MQFMT_ADMIN.

Datos de la aplicación

Contienen el mensaje PCF, incluida la cabecera PCF, en el que:

- El tipo de mensaje PCF (*Type*) especifica MQCFT_COMMAND.
- El identificador de mandato especifica el mandato, por ejemplo, *Change Queue* (MQCMD_CHANGE_Q).

Los PCF de escape son mandatos PCF que contienen mandatos MQSC dentro del texto del mensaje. Los PCF se pueden utilizar para enviar mandatos a un gestor de colas remoto. Consulte el apartado "Utilizar la MQAI para simplificar el uso de los PCF" en la página 38 para obtener más información.

Para obtener una descripción completa de las estructuras de datos PCF y cómo implementarlas, consulte Estructuras para mandatos y respuestas.

IBM i Utilización de IBM MQ Explorer con IBM MQ for IBM i

Utilice esta información para administrar IBM MQ for IBM i con IBM MQ Explorer.

IBM MQ for Windows (plataforma x86) e IBM MQ para Linux (plataformas x86 y x86-64) proporcionan una interfaz de administración llamada IBM MQ Explorer para realizar tareas de administración, como alternativa al uso de mandatos CL, de control o MQSC.

IBM MQ Explorer le permite realizar la administración local o remota de la red desde un sistema que ejecute Windows (plataforma x86) o Linux (plataformas x86 y x86-64), apuntando IBM MQ Explorer a los gestores de colas y clústeres en los que esté interesado.

Con IBM MQ Explorer, puede hacer lo siguiente:

- Iniciar y detener un gestor de colas (sólo en la máquina local).
- Definir, visualizar y modificar las definiciones de objetos de IBM MQ, como colas, temas y canales.
- Examinar los mensajes de una cola.
- Iniciar y detener un canal.
- Ver información del estado de un canal.
- Ver los gestores de colas de un clúster.
- Comprobar qué aplicaciones, usuarios o canales tienen una determinada cola abierta.
- Crear un nuevo clúster de gestor de colas utilizando el asistente para **Crear un nuevo clúster**.
- Añadir un gestor de colas a un clúster utilizando el asistente para **Añadir un gestor de colas a un clúster**.
- Gestionar el objeto de información de autenticación, que se utiliza con la seguridad de canal de TLS (seguridad de la capa de transporte).

Utilizando la ayuda en línea, puede:

- Definir y controlar diversos recursos que incluyen gestores de colas, colas, canales, definiciones de proceso, canales de conexión de cliente, escuchas, servicios, listas de nombres y clústeres.
- Iniciar o detener un gestor de colas y los procesos asociados a él.
- Ver gestores de colas y sus objetos asociados en su estación de trabajo o desde otras estaciones de trabajo.
- Comprobar el estado de gestores de colas, clústeres y canales.

Asegúrese de haber satisfecho los siguientes requisitos antes de intentar utilizar IBM MQ Explorer para gestionar IBM MQ en una máquina servidor. Compruebe que:

1. Hay un servidor de mandatos en ejecución para cualquier gestor de colas que se esté administrando, iniciado en el servidor con el mandato CL **STRMQMCSVR**.
2. Existe un escucha TCP/IP adecuado para cada gestor de colas remoto. Este es el escucha de IBM MQ que se ha iniciado con el mandato **STRMQMLSR**.
3. El canal de conexión con el servidor, llamado `SYSTEM.ADMIN.SVRCONN`, existe en todos los gestores de colas remotos. Debe crear este canal usted mismo. Este canal es necesario para cada gestor de colas remoto que se esté administrando. Sin él, la administración remota no es posible.
4. Verifique que la cola `SYSTEM.MQEXPLORER.REPLY.MODEL` exista.

IBM i Gestión del servidor de mandatos para la administración remota en IBM i

Utilice esta información sobre obtener más información sobre la administración remota del servidor de mandatos de IBM MQ for IBM i.

Cada gestor de colas puede tener un servidor de mandatos asociado. El servidor de mandatos procesa todos los mandatos entrantes procedentes de gestores de colas remotos o los mandatos PCF procedentes de aplicaciones. Presenta los mandatos al gestor de colas para que los procese y devuelve un código de terminación o un mensaje de operador, dependiendo del origen del mandato.

Es obligatorio tener un servidor de mandatos para toda tarea de administración que implique los PCF, la MQAI y también para la administración remota.

Nota: En la administración remota, debe asegurarse de que el gestor de colas de destino está ejecutándose. De lo contrario, los mensajes que contienen mandatos no pueden salir del gestor de colas desde el que se han emitido. En vez de ello, estos mensajes se transfieren a la cola de transmisión local que sirve al gestor de colas remoto. Evite esta situación en la medida de lo posible.

Existen mandatos de control individuales para iniciar y detener el servidor de mandatos. Puede realizar las operaciones descritas en las secciones siguientes utilizando IBM MQ Explorer.

Iniciar y detener el servidor de mandatos

Para iniciar el servidor de mandatos, utilice este mandato CL:

```
STRMQMCSVR MQMNAME('saturn.queue.manager')
```

donde `saturn.queue.manager` es el gestor de colas para el que se inicia el servidor de mandatos.

Para detener el servidor de mandatos, utilice uno de los siguientes mandatos CL:

1. ```
   ENDMQMCSVR MQMNAME('saturn.queue.manager') OPTION(*CNTRLD)
   ```

   Realizar una detención controlada, donde `saturn.queue.manager` es el gestor de colas para el que el servidor de mandatos se está deteniendo. Esta es la opción predeterminada, lo que significa que `OPTION(*CNTRLD)` se puede omitir.

2. ```
   ENDMQMCSVR MQMNAME('saturn.queue.manager') OPTION(*IMMED)
   ```

   Realizar una detención inmediata, donde `saturn.queue.manager` es el gestor de colas para el que el servidor de mandatos se está deteniendo.

Visualización del estado del servidor de mandatos

Para la administración remota, debe asegurarse de que el servidor de mandatos del gestor de colas de destino está ejecutándose. Si no se está ejecutando, no se podrá procesar ningún mandato remoto. Todos los mensajes que contienen mandatos se ponen en la cola de mandatos del gestor de colas de destino `SYSTEM.ADMIN.COMMAND.QUEUE`.

Para visualizar el estado del servidor de mandatos de un gestor de colas, llamado `saturn.queue.manager` en este ejemplo, el mandato CL es:

```
DSPMQMCSVR MQMNAME('saturn.queue.manager')
```

Emita este mandato en la máquina destino. Si el servidor de mandatos está ejecutándose, aparece el panel de la Figura 21 en la página 404:

```
Display MQM Command Server (DSPMQMCSVR)


Queue manager name . . . . . > saturn.queue.manager

MQM Command Server Status. . . . > RUNNING




F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

Figura 21. Panel Visualizar servidor de mandatos de MQM

IBM i Ejecución de mandatos de consola web

Debe configurar el entorno tal como se describe en el texto siguiente, para que los mandatos de Qshell relacionados con la consola web se ejecuten correctamente en IBM MQ for IBM i.

Acerca de esta tarea

Cuando se inicia Qshell, initializa las tablas internas para procesar los mandatos basándose en el identificador de juego de caracteres codificados del trabajo. Para que los mandatos de Qshell relacionados con la consola web se ejecuten correctamente en IBM i, debe configurar el entorno.

Para establecer un entorno local, se establece la variable de entorno LANG en el nombre de vía de acceso de un objeto de entorno local. Por ejemplo, para establecer el entorno local para inglés de EE.UU., la variable de entorno LANG se establece de la siguiente manera:

```
LANG=/QSYS.LIB/EN_US.LOCALE
```

En el Qshell, puede comprobar el valor mediante el conjunto de mandatos de emisión para enumerar todas las variables de entorno. Normalmente, es LANG, que puede afectar al entorno local para el entorno de ejecución. También puede tener LC_ALL.

Para ejecutar mandatos de Qshell correctamente, el valor del entorno local debe ser coherente con el valor del trabajo.

Procedimiento

Utilice el mandato de CL DSPJOB JOB(JobNumber/USERProfile/JobName)

a) Seleccione la opción 2 para mostrar los atributos de definición de trabajo.

b) Los atributos siguientes deben ser coherentes con el valor de entorno LANG o LC_ALL

- Identificador de idioma
- Identificador de país o región
- Identificador de juego de caracteres codificado

Por ejemplo, si

```
LANG=/QSYS.LIB/FR_FR.LOCALE
```

los atributos de trabajo serían:

- ID de idioma . . . . . FRA
- ID de país o región . . . . . FR
- ID de juego de caracteres codificado. . . . 297

Qué hacer a continuación

Para obtener más información sobre el soporte multilingüístico, consulte el tema de IBM Documentation Consideraciones sobre el soporte multilingüístico.

IBM i Gestión de trabajo en IBM i

Esta información describe la forma en que IBM MQ maneja las solicitudes de trabajo y detalla las opciones disponibles para priorizar y controlar los trabajos asociados a IBM MQ.

Aviso

No modifique los objetos de gestión de trabajo de IBM MQ a menos que comprenda perfectamente los conceptos de gestión de trabajo de IBM i e IBM MQ.

Puede encontrar información adicional sobre los subsistemas y las descripciones de trabajo en Gestión de trabajo en la documentación del producto IBM i. Preste especial atención a las secciones que tratan sobre el inicio y direccionamiento de trabajos (Starting jobs y Batch jobs).

IBM MQ for IBM i incorpora el entorno IBM i UNIX y hebras IBM i. **No** haga ningún cambio en los objetos del Sistema de archivos integrado (IFS).

Durante el funcionamiento normal, un gestor de colas de IBM MQ inicia una serie de trabajos por lotes para realizar diferentes tareas. De forma predeterminada, estos trabajos por lotes se ejecutan en el subsistema QMQM que se crea cuando se instala IBM MQ.

La gestión de trabajo se refiere al proceso de adaptar las tareas de IBM MQ para obtener el rendimiento óptimo del sistema, o para simplificar la administración.

Por ejemplo, puede:

- Cambiar la prioridad de ejecución de los trabajos para hacer que un gestor de colas responda antes que otro.
- Redirigir la salida de una serie de trabajos a una cola de salida concreta.
- Hacer que todos los trabajos de un cierto tipo se ejecuten en un subsistema específico.
- Aislar los errores en un subsistema.

La gestión de trabajo se lleva a cabo creando o cambiando las descripciones de trabajo asociadas a los trabajos de IBM MQ. Puede configurar la gestión de trabajo para:

- Toda una instalación de IBM MQ.
- Gestores de colas individuales.
- Trabajos individuales de gestores de colas individuales.

IBM i Tareas IBM MQ para IBM i

Esta tabla contiene los trabajos de IBM MQ for IBM i y una breve descripción de cada uno.

Cuando un gestor de colas está en ejecución, puede ver algunos o todos los trabajos por lotes siguientes ejecutándose bajo el perfil de usuario QMQM en el subsistema de IBM MQ. Los trabajos se describen brevemente en la Tabla 21 en la página 405.

Puede ver todos los trabajos conectados a un gestor de colas utilizando la opción 22 del panel **Trabajar con gestor de colas** (WRKMQM). Para ver los escuchas, utilice el mandato WRKMQMLSR.

Tabla 21. Tareas de IBM MQ.

Nombre de trabajo	Función
AMQZMUC0	Gestor de programas de utilidad. Este trabajo ejecuta programas de utilidad del gestor de colas críticos, por ejemplo, el gestor de cadenas de diario.

Tabla 21. Tareas de IBM MQ. (continuación)	
Nombre de trabajo	**Función**
AMQZXMA0	El controlador de ejecución que es el primer trabajo que inicia el gestor de colas. Maneja las solicitudes MQCONN e inicia procesos de agente para que procesen las llamadas de API de IBM MQ.
AMQZFUMA	Gestor de autorizaciones sobre objetos (OAM).
AMQZLAA0	Agentes de gestor de colas que realizan la mayor parte del trabajo de las aplicaciones que se conectan al gestor de colas utilizando MQCNO_STANDARD_BINDING.
AMQZLSA0	Agente de gestor de colas.
AMQZMUF0	Gestor de programas de utilidad
AMQZMGR0	Controlador de procesos. Este trabajo se utiliza para iniciar y gestionar escuchas y servicios.
AMQZMUR0	Gestor de programas de utilidad. Este trabajo ejecuta programas de utilidad del gestor de colas críticos, por ejemplo, el gestor de cadenas de diario.
AMQFQPUB	Daemon de publicación/suscripción en cola.
AMQFCXBA	Trabajo de operador de intermediario.
RUNMQBRK	Trabajo de control de intermediario.
AMQRMPPA	Trabajo de agrupación de procesos de canal.
AMQCRSTA	Canal de respuesta invocado a través de TCP/IP.
AMQCRS6B	Canal receptor y conexión de cliente de LU62 (vea la nota).
AMQRRMFA	Gestor de depósitos para clústeres.
AMQCLMAA	Escucha TCP/IP sin hebras.
AMQPCSEA	Procesador de mandatos PCF que maneja las peticiones de administración remota y PCF.
RUNMQTRM	Supervisor desencadenante.
RUNMQDLQ	Manejador de cola de mensajes no entregados.
RUNMQCHI	El iniciador de canal.
RUNMQCHL	Trabajo de canal emisor que se inicia para cada canal emisor.
RUNMQLSR	Escucha TCP/IP con hebras.
AMQRCMLA	Procesador de mandatos MQSC y PCF de canal.

Nota: Se ejecuta el trabajo receptor LU62 en el subsistema de comunicaciones y adquiere sus propiedades de ejecución de las entradas de direccionamiento y de comunicaciones que se utilizan para iniciar el trabajo. Consulte Extremo iniciado (receptor) para obtener más información.

IBM i Objetos de gestión de trabajo en IBM i

Cuando se instala IBM MQ, se proporcionan diversos objetos en la biblioteca QMQM para ayudarle con la gestión del trabajo. Estos objetos son los necesarios para que los trabajos de IBM MQ se ejecuten en el subsistema que les corresponde.

Se proporcionan descripciones de trabajo de ejemplo para dos de los trabajos por lotes de IBM MQ. Si no se proporciona ninguna descripción de trabajo específica para un trabajo de IBM MQ, el trabajo se ejecuta con la descripción de trabajo predeterminada QMQMJOBD.

Los objetos de gestión de trabajo que se suministran cuando se instala IBM MQ se listan en la Tabla 22 en la página 407 y los objetos que se crean para un gestor de colas se listan en la Tabla 23 en la página 407

Nota: Los objetos de gestión de trabajo pueden encontrarse en la biblioteca QMQM y los objetos del gestor de colas pueden encontrarse en la biblioteca del gestor de colas.

Tabla 22. Objetos de gestión de trabajo

Nombre	Tipo	Descripción
AMQZLAA0	*JOBD	La descripción de trabajo que utilizan los procesos de agente de IBM MQ
AMQZLSA0	*JOBD	El agente del gestor de colas de enlaces aislados
AMQZXMA0	*JOBD	La descripción de trabajo que utilizan los controladores de ejecución de IBM MQ
QMQM	*SBSD	El subsistema en el que se ejecutan todos los trabajos de IBM MQ
QMQM	*JOBQ	La cola de trabajos conectada al subsistema suministrado
QMQMJOBD	*JOBD	La descripción de trabajo predeterminada de IBM MQ, que utiliza si no hay una descripción de trabajo específica para un trabajo
QMQMMSG	*MSGQ	La cola de mensajes predeterminada para los trabajos de IBM MQ
QMQMRUN20	*CLS	Una descripción de clase para los trabajos de IBM MQ de prioridad alta
QMQMRUN35	*CLS	Una descripción de clase para los trabajos de IBM MQ de prioridad media
QMQMRUN50	*CLS	Una descripción de clase para los trabajos de IBM MQ de prioridad baja

Tabla 23. Objetos de gestión de trabajo creados para un gestor de colas

Nombre	Tipo	Descripción
AMQA000000	*JRNRCV	Receptor de diario local
AMQAJRN	*JRN	Diario local
AMQJRNINF	*USRSPC	Espacio de usuario que se actualiza con los últimos receptores de diario necesarios para el arranque y la recuperación desde soporte de almacenamiento de un gestor de colas. Este espacio de usuario puede consultarlo una aplicación para determinar los receptores de diario que requieren archivado y los que se pueden suprimir de forma segura.
AMQAJRNMSG	*MSGQ	Cola de mensajes de diario local
AMQCRC6B	*PGM	Programa para iniciar la conexión LU6.2
AMQRFOLD	*XX_ENCODE _CASE_ONE archivo	Archivo de definiciones de canal del gestor de colas migrado
QMQMMSG	*MSGQ	Cola de mensajes del gestor de colas

IBM i Cómo IBM MQ usa los objetos de gestión de trabajo en IBM i

Esta información describe la forma en que IBM MQ utiliza los objetos de gestión de trabajo, y proporciona ejemplos de configuración.

Atención: No modifique los valores de entrada de la cola de trabajos en el subsistema QMQM para limitar el número de trabajos permitidos en el subsistema según la prioridad. Si intenta hacer esto, puede impedir la ejecución de trabajos de IBM MQ esenciales tras su envío y hacer que el inicio del gestor de colas falle.

Para entender cómo se configura la gestión de trabajo, es necesario entender primero cómo utiliza IBM MQ la descripciones de trabajo.

La descripción de trabajo empleada para iniciar el trabajo controla numerosos atributos del trabajo. Por ejemplo:

- La cola de trabajos en la que se pone el trabajo y el subsistema en el que se ejecuta el trabajo.
- Los datos de direccionamiento que se utilizan para iniciar el trabajo y la clase que el trabajo emplea para los parámetros en tiempo de ejecución.
- La cola de salida que el trabajo utiliza para los archivos de impresión.

El proceso de iniciar un trabajo de IBM MQ puede considerarse un proceso de tres pasos:

1. IBM MQ selecciona una descripción de trabajo.

   IBM MQ utiliza la siguiente técnica para determinar qué descripción de trabajo se ha de utilizar para un trabajo por lotes:

   a. Busca en la biblioteca del gestor de colas una descripción de trabajo que tenga el nombre del trabajo. Consulte Comprender los nombres de biblioteca del gestor de colas IBM MQ for IBM i para obtener más detalles sobre la biblioteca del gestor de colas.

   b. Busca en la biblioteca del gestor de colas la descripción de trabajo predeterminada, que es QMQMJOBD.

   c. Busca en la biblioteca QMQM una descripción de trabajo cuyo nombre sea el del trabajo.

   d. Utiliza la descripción de trabajo predeterminada, QMQMJOBD, de la biblioteca QMQM.

2. El trabajo se somete a la cola de trabajos.

   Las descripciones de trabajo que se suministran con IBM MQ se han configurado para que, de forma predeterminada, pongan los trabajos en la cola de trabajos QMQM de la biblioteca QMQM. La cola de trabajos QMQM está conectada al subsistema QMQM suministrado y, por eso, los trabajos empiezan a ejecutarse de forma predeterminada en el subsistema QMQM.

3. El trabajo entra en el subsistema y sigue los pasos de direccionamiento.

   Cuando el trabajo entra en el subsistema, se utilizan los datos de direccionamiento indicados en la descripción de trabajo para localizar las entradas de direccionamiento del trabajo.

   Los datos de direccionamiento deben corresponderse con una de las entradas de direccionamiento definidas en el subsistema QMQM, y ello define cuál de las clases suministradas (QMQMRUN20, QMQMRUN35 o QMQMRUN50) va a utilizar el trabajo.

Nota: Si los trabajos de IBM MQ no parecen estar iniciándose, asegúrese de que el subsistema está ejecutándose y que la cola de trabajos no está retenida.

Si ha modificado los objetos de gestión de trabajo de IBM MQ, asegúrese de que todo está asociado correctamente. Por ejemplo, si especifica una cola de trabajos distinta de QMQM/QMQM en la descripción de trabajo, asegúrese de que se realiza un ADDJOBQE para el subsistema, que es QMQM.

Puede crear una descripción de trabajo para cada trabajo documentado en la Tabla 21 en la página 405 utilizando la siguiente hoja de trabajo como ejemplo:

```
What is the queue manager library name? ___________
Does job description AMQZXMA0 exist in the queue manager library? Yes   No
```

```
Does job description QMQMJOBD exist in the queue manager library? Yes   No
Does job description AMQZXMA0 exist in the QMQM library?     Yes   No
Does job description QMQMJOBD exist in the QMQM library?     Yes   No
```

Si responde No a todas estas preguntas, cree una descripción de trabajo global QMQMJOBD en la biblioteca QMQM.

La cola de mensajes de IBM MQ

Se crea una cola de mensajes de IBM MQ, QMQMMSG, en cada biblioteca de gestor de colas. Se envían mensajes del sistema operativo a esta cola cuando finalizan trabajos del gestor de colas y IBM MQ envía mensajes a la cola. Por ejemplo, para notificar qué receptores de diario se necesitan al arrancar. Procure mantener el número de mensajes de esta cola de mensajes en un tamaño razonable para que sea más fácil de supervisar.

IBM i Ejemplos de funcionamiento predeterminado en IBM i

Estos ejemplos muestran cómo funciona una instalación de IBM MQ sin modificar cuando algunos de los trabajos estándar se someten durante el inicio del gestor de colas.

En primer lugar, se inicia el trabajo del controlador de ejecución AMQZXMA0.

1. Emita el mandato **STRMQM** para el gestor de colas TESTQM.
2. IBM MQ realiza una búsqueda en la biblioteca de gestor de colas QMTESTQM, primero de la descripción de trabajo AMQZXMA0 y luego de la descripción de trabajo QMQMJOBD.

   No existe ninguna de estas descripciones de trabajo, por lo que IBM MQ busca la descripción de trabajo AMQZXMA0 en la biblioteca del producto QMQM. Esta descripción de trabajo existe, por lo que se emplea para someter el trabajo.
3. La descripción de trabajo utiliza la cola de trabajos predeterminada de IBM MQ, por lo que el trabajo se somete a la cola de trabajos QMQM/QMQM.
4. Los datos de direccionamiento existentes en la descripción de trabajo AMQZXMA0 son QMQMRUN20, de modo que el sistema busca en las entradas de direccionamiento del subsistema una entrada que se corresponda con esos datos.

   De forma predeterminada, la entrada de direccionamiento que tiene el número de secuencia 9900 tiene datos de comparación que se corresponden con QMQMRUN20, por lo que el trabajo se inicia con la clase definida en esa entrada de direccionamiento, que también se llama QMQMRUN20.
5. La clase QMQM/QMQMRUN20 tiene la prioridad de ejecución establecida en 20; así que el trabajo AMQZXMA0 se ejecuta en el subsistema QMQM con la prioridad que tienen los trabajos más interactivos del sistema.

IBM i Configuración de ejemplos de gestión de trabajo en IBM i

Utilice esta información para obtener más información acerca de cómo puede cambiar y crear descripciones de trabajo de IBM MQ para modificar los atributos de tiempo de ejecución de los trabajos de IBM MQ.

La clave de la flexibilidad de la gestión de trabajo de IBM MQ consiste en que IBM MQ busca las descripciones de trabajo en dos niveles:

- Si crea o cambia las descripciones de trabajo en una biblioteca del gestor de colas, esos cambios alteran temporalmente las descripciones de trabajo globales existentes en QMQM, pero los cambios son locales y solo afectan a ese gestor de colas en concreto.
- Si crea o cambia descripciones de trabajo globales en la biblioteca QMQM, esas descripciones de trabajo afectan a todos los gestores de colas del sistema, a menos que se alteren temporalmente de forma local a nivel de gestores de colas individuales.

1. El ejemplo siguiente muestra cómo aumentar la prioridad de los trabajos de control de canal para un gestor de colas individual.

Para lograr que los trabajos de gestor de repositorios y de iniciador de canal, llamados AMQRRMFA y RUNMQCHI, se ejecuten lo antes posible para el gestor de colas TESTQM, siga estos pasos:

a. Cree duplicados locales de la descripción de trabajo QMQM/QMQMJOBD con los nombres de los procesos de IBM MQ que desea controlar en la biblioteca del gestor de colas. Por ejemplo:

```
CRTDUPOBJ OBJ(QMQMJOBD) FROMLIB(QMQM) OBJTYPE(*JOBD) TOLIB(QMTESTQM)
NEWOBJ(RUNMQCHI)
CRTDUPOBJ OBJ(QMQMJOBD) FROMLIB(QMQM) OBJTYPE(*JOBD) TOLIB(QMTESTQM)
NEWOBJ(AMQRRMFA)
```

b. Cambie el parámetro de datos de direccionamiento existente en la descripción de trabajo para asegurar que los trabajos utilicen la clase QMQMRUN20.

```
CHGJOBD JOBD(QMTESTQM/RUNMQCHI) RTGDTA('QMQMRUN20')
CHGJOBD JOBD(QMTESTQM/AMQRRMFA) RTGDTA('QMQMRUN20')
```

Ahora, los trabajos AMQRRMFA y RUNMQCHI del gestor de colas TESTQM van a:

- Utilizar las nuevas descripciones de trabajo locales que hay en la biblioteca del gestor de colas
- Ejecutarse con la prioridad 20, pues se emplea la clase QMQMRUN20 cuando los trabajos entran en el subsistema

2. El ejemplo siguiente define una nueva clase de prioridad de ejecución para el subsistema QMQM.

a. Cree una clase duplicada en la biblioteca QMQM para permitir que otros gestores de colas accedan a la clase, emitiendo el siguiente mandato:

```
CRTDUPOBJ OBJ(QMQMRUN20) FROMLIB(QMQM) OBJTYPE(*CLS) TOLIB(QMQM)
NEWOBJ(QMQMRUN10)
```

b. Cambie la clase para que tenga la nueva prioridad de ejecución, emitiendo el siguiente mandato:

```
CHGCLS CLS(QMQM/QMQMRUN10) RUNPTY(10)
```

c. Añada la nueva definición de clase en el subsistema, emitiendo el siguiente mandato:

```
ADDRTGE SBSD(QMQM/QMQM) SEQNBR(8999) CMPVAL('QMQMRUN10') PGM(QSYS/QCMD)
CLS(QMQM/QMQMRUN10)
```

Nota: Puede especificar cualquier valor numérico para el número de secuencia de direccionamiento, pero los valores deben estar en orden secuencial. Este número de secuencia indica al subsistema el orden en que debe buscarse una coincidencia de los datos de direccionamiento en las entradas de direccionamiento.

d. Cambie la descripción de trabajo local o global para que utilice la nueva clase de prioridad, emitiendo el siguiente mandato:

```
CHGJOBD JOBD(QMQMlibname/QMQMJOBD) RTGDTA('QMQMRUN10')
```

Ahora todos los trabajos del gestor de colas asociados a QMlibraryname utilizan una prioridad de ejecución de 10.

3. El ejemplo siguiente enseña cómo se ejecuta un gestor de colas en su propio subsistema.

Para hacer que todos los trabajos del gestor de colas TESTQM se ejecuten en el subsistema QBATCH, siga estos pasos:

a. Cree un duplicado local de la descripción de trabajo QMQM/QMQMJOBD en la biblioteca del gestor de colas, con este mandato

```
CRTDUPOBJ OBJ(QMQMJOBD) FROMLIB(QMQM) OBJTYPE(*JOBD) TOLIB(QMTESTQM)
```

b. Cambie el parámetro cola de trabajos, en la descripción de trabajo, para asegurar que los trabajos utilicen la cola de trabajos QBATCH.

```
CHGJOBD JOBD(QMTESTQM/QMQMJOBD) JOBQ(*LIBL/QBATCH)
```

Nota: La cola de trabajos está asociada a la descripción del subsistema. Si encuentra que los trabajos permanecen en la cola de trabajos, verifique que la definición de la cola de trabajos está definida en el SBSD. Utilice el mandato DSPSBSD para el subsistema y seleccione la opción 6, Entradas de la cola de trabajos.

Ahora, todos los trabajos del gestor de colas TESTQM van a:

- Utilizar la nueva descripción de trabajo local predeterminada que hay en la biblioteca del gestor de colas
- Enviarse a la cola de trabajos QBATCH

Para estar seguro de que los trabajos se direccionan correctamente y adquieren la prioridad deseada, puede:

- Crear entradas de direccionamiento para los trabajos de IBM MQ en el subsistema QBATCH, o bien
- Basarse en una entrada de direccionamiento global que invoque QCMD, sin tener en cuenta los datos de direccionamiento utilizados.

  Esta opción sólo funciona si se establece el valor *NOMAX en la opción de máximo de trabajos activos para la cola de trabajos QBATCH. El valor predeterminado del sistema es 1.

4. El siguiente ejemplo crea otro subsistema IBM MQ

   a. Cree un subsistema duplicado en la biblioteca QMQM, emitiendo el siguiente mandato:

```
CRTDUPOBJ OBJ(QMQM) FROMLIB(QMQM) OBJTYPE(*SBSD) TOLIB(QMQM) NEWOBJ(QMQM2)
```

   b. Elimine la cola de trabajos QMQM, emitiendo el siguiente mandato:

```
RMVJOBQE SBSD(QMQM/QMQM2) JOBQ(QMQM/QMQM)
```

   c. Cree una nueva cola de trabajos para el subsistema, emitiendo el siguiente mandato:

```
CRTJOBQ JOBQ(QMQM/QMQM2) TEXT('Job queue for IBM MQ Queue Manager')
```

   d. Añada una entrada de la cola de trabajos en el subsistema, emitiendo el siguiente mandato:

```
ADDJOBQE SBSD(QMQM/QMQM2) JOBQ(QMQM/QMQM2) MAXACT(*NOMAX)
```

   e. Cree un QMQMJOBD duplicado en la biblioteca del gestor de colas, emitiendo el siguiente mandato:

```
CRTDUPOBJ OBJ(QMQMJOBD) FROMLIB(QMQM) OBJTYPE(*JOBD) TOLIB(QMlibraryname)
```

   f. Cambie la descripción de trabajo para que utilice la nueva cola de trabajos, emitiendo el siguiente mandato:

```
CHGJOBD JOBD(QMlibraryname/QMQMJOBD) JOBQ(QMQM/QMQM2)
```

   g. Inicie el subsistema emitiendo el siguiente mandato:

```
STRSBS SBSD(QMQM/QMQM2)
```

Nota:

a. Puede especificar el subsistema en cualquier biblioteca. Si por algún motivo se vuelve a instalar el producto, o se sustituye la biblioteca QMQM, se eliminarán los cambios que haya realizado.

b. Ahora todos los trabajos del gestor de colas asociados a QMlibraryname se ejecutan bajo el subsistema QMQM2.

IBM i Disponibilidad, copia de seguridad, recuperación y reinicio en IBM i

Utilice esta información para entender cómo IBM MQ for IBM i utiliza el soporte de registro por diario de IBM i como ayuda para su estrategia de copia de seguridad y restauración.

Debe conocer los métodos de copia de seguridad y recuperación estándar de IBM i, así como con el uso de diarios y los receptores de diario asociados en IBM i antes de leer esta sección. Para obtener información sobre estos temas, consulte Copia de seguridad y recuperación.

Para entender la estrategia de copia de seguridad y recuperación, primero debe entender cómo IBM MQ for IBM i organiza los datos en el sistema de archivos de IBM i y en el sistema de archivos integrado (IFS).

IBM MQ for IBM i guarda los datos en una biblioteca individual para cada instancia del gestor de colas y en archivos continuos del sistema de archivos IFS.

Las bibliotecas específicas de gestor de colas contienen diarios, receptores de diario y los objetos necesarios para controlar la gestión de trabajo del gestor de colas. Los directorios y archivos del IFS contienen los archivos de configuración de IBM MQ, las descripciones de los objetos de IBM MQ y los datos contenidos en ellos.

Todo cambio que se realice en estos objetos (que sea recuperable en caso de anomalía del sistema) se registra en un diario *antes* de ser aplicado al correspondiente objeto. Así, estos cambios se pueden recuperar reproduciendo la información registrada en el diario.

Puede configurar IBM MQ for IBM i para que utilice varias instancias del gestor de colas en distintos servidores para aumentar la disponibilidad del gestor de colas y acelerar la recuperación en caso de anomalía de un servidor o del gestor de colas.

IBM i Diarios del gestor de colas en IBM i

Utilice esta información para entender cómo IBM MQ for IBM i utiliza los diarios en su operación para controlar las actualizaciones realizadas en objetos locales.

Cada biblioteca de gestor de colas contiene un diario para ese gestor de colas, y el diario tiene el nombre QM *GRLIB*/AMQ *A* JRN, donde QM *GRLIB* es el nombre de la biblioteca del gestor de colas, y *A* es una letra, A en el caso de un gestor de colas de instancia única, que es exclusiva de la instancia del gestor de colas.

QM *GRLIB* toma el nombre QM, seguido del nombre del gestor de colas en un formato exclusivo. Por ejemplo, un gestor de colas llamado TEST tiene una biblioteca de gestor de colas llamado QMTEST. La biblioteca del gestor de colas se puede especificar al crear un gestor de colas mediante el mandato **CRTMQM**.

Los diarios tienen receptores de diario asociados, que contienen la información que se registra por diario. Los receptores son objetos a los que sólo se puede añadir información y que, con el tiempo, se llenan.

Los receptores de diario utilizan un valioso espacio de memoria con información anticuada. Sin embargo, para minimizar este problema, siempre puede colocar la información en un almacenamiento permanente. En cualquier momento, hay un receptor de diario conectado al diario. El receptor de diario, cuando llega al tamaño umbral predeterminado, se desconecta y se sustituye por un nuevo receptor de diario. Puede especificar el umbral de receptores de diario al crear un gestor de colas utilizando **CRTMQM** y el parámetro **THRESHOLD**.

Los receptores de diario asociados al diario local de IBM MQ para IBM i existen en cada biblioteca de gestor de colas y adoptan el siguiente convenio de denominación:

```
AMQ Arnnnnn
```

donde

A
es una letra de la A-Z. La A es para gestores de cola con una sola instancia. Variará para las distintas instancias de un gestor de colas multiinstancia.

NNNNN
es 00000 to 99999 decimal que se incrementa en 1 para el siguiente diario de la secuencia.

r
es decimal 0 to 9, que se incrementa en 1 cada vez que se restaura un receptor.

La secuencia de los diarios se basa en la fecha. No obstante, la denominación del diario siguiente se basa en estas reglas:

1. A AMQArnnnnn le sigue AMQAr(nnnn+1) y el número nnnnn se recorta cuando llega a 99999. Por ejemplo, después de AMQA099999 viene AMQA000000 y después de AMQA999999 sigue AMQA900000.
2. Si ya existe un diario cuyo nombre se haya generado mediante la regla 1, se envía el mensaje CPI70E3 a la cola de mensajes QSYSOPR y se detiene la conmutación automática de receptores.

   Se sigue utilizando el receptor conectado en ese momento hasta que se investigue el problema y se conecte manualmente un nuevo receptor.
3. Si se terminan los nombres de la secuencia (es decir, en el sistema ya existen todos los nombres de diario posibles), deberá realizar las dos tareas siguientes:
   a. Suprimir los diarios que ya no se necesiten (consulte el apartado "Gestión de diarios en IBM i" en la página 418).
   b. Registrar los cambios de diario en el receptor de diario más reciente (utilizando **RCDMQMIMG**) y luego repetir el paso anterior. Así se podrán reutilizar los nombres de los receptores de diario antiguos.

El diario AMQAJRN utiliza la opción MNGRCV(*SYSTEM) para permitir que el sistema operativo cambie automáticamente los receptores de diario cuando se alcance el umbral. Para obtener más información sobre el modo en que el sistema maneja los receptores, consulte *IBM i Backup and Recovery*.

El valor umbral predeterminado del receptor de diario es de 100.000 KB. Al crear el gestor de colas, puede establecer este valor en uno más grande. El valor inicial del atributo LogReceiverSize se escribe en la stanza LogDefaults del archivo mqs.ini.

Cuando un receptor de diario se amplía más allá del umbral especificado, el receptor se separa y se crea un nuevo receptor de diario que hereda los atributos del receptor anterior. Los cambios en los atributos LogReceiverSize o LogASP después de que se haya creado un gestor de colas se ignoran cuando el sistema conecta automáticamente un nuevo receptor de diario

Consulte Modificación de la información de configuración de IBM MQ en Multiplatforms para obtener más detalles sobre la configuración del sistema.

Si necesita cambiar el tamaño de los receptores de diario una vez creado el gestor de colas, cree un nuevo receptor de diario y establezca que su propietario es QMQM, utilizando para ello los siguientes mandatos:

```
CRTJRNRCV JRNRCV(QM GRLIB/AMQ Arnnnnn) THRESHOLD(xxxxxx) +
TEXT('MQM LOCAL JOURNAL RECEIVER')
CHGOBJOWN OBJ(QM GRLIB/AMQ Arnnnnn) OBJTYPE(*JRNRCV) NEWOWN(QMQM)
```

donde

QMGRLIB
Es el nombre de la biblioteca del gestor de colas

A
Es el identificador de la instancia (normalmente A).

rnnnnn
Es el siguiente receptor de diario en la secuencia de denominación descrita anteriormente

xxxxxx
Es el nuevo umbral de receptor (en KB)

Nota: El tamaño máximo del receptor está regido por el sistema operativo. Para comprobar este valor, mire la palabra clave THRESHOLD en el mandato **CRTJRNRCV**.

Ahora conecte el nuevo receptor al diario AMQAJRN con el mandato:

```
CHGJRN JRN(QMGRLIB/AMQ A JRN) JRNRCV(QMGRLIB/AMQ Annnnnn)
```

En "Gestión de diarios en IBM i" en la página 418 hallará los detalles de cómo puede gestionar estos receptores de diario.

IBM i *Uso del diario del gestor de colas en IBM i*

Utilice esta información para entender cómo IBM MQ for IBM i utiliza los diarios en su operación para controlar las actualizaciones realizadas en objetos locales.

Las actualizaciones persistentes realizadas en las colas de mensajes se producen en dos etapas. Primero, los registros que representan la actualización se graban en el diario y después se actualiza el archivo de cola.

Por lo tanto, puede suceder que los receptores de diario estén más actualizados que los archivos de las colas. Para asegurar que el proceso de reinicio empiece desde un punto coherente, IBM MQ utiliza puntos de comprobación.

Un punto de comprobación es un momento específico en el que el registro descrito en el diario es igual que el registro de la cola. El punto de comprobación propiamente dicho consta de una serie de registros de diario necesarios para reiniciar el gestor de colas. Por ejemplo, el estado de todas las transacciones (es decir, las unidades de trabajo) activas en el momento del punto de comprobación.

Los puntos de comprobación los genera automáticamente IBM MQ. Estos se toman cuando el gestor de colas se inicia y se cierra, y después de que se anoten un determinado número de operaciones.

Puede forzar que un gestor de colas tome un punto de comprobación emitiendo el mandato RCDMQMIMG en todos los objetos de un gestor de colas y visualizando los resultados, como se indica a continuación:

```
RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) MQMNAME(Q_MGR_NAME) DSPJRNDTA(*YES)
```

A medida que las colas manejan nuevos mensajes, el registro de punto de comprobación deja de ser coherente con el estado actual de las colas.

Cuando se reinicia IBM MQ, éste localiza el registro de punto de comprobación más reciente en el registro. Esta información se guarda en el archivo de punto de comprobación que se actualiza al final de cada punto de comprobación. El registro de punto de comprobación representa el punto más reciente de coherencia entre el archivo de registro y los datos. Los datos de este punto de comprobación se utilizan para reconstruir las colas tal como eran en el momento del punto de comprobación. Al reconstruir las colas, se reproduce el registro para devolver a las colas el estado que tenían antes de la anomalía o el cierre del sistema.

Para entender cómo IBM MQ utiliza el diario, considere el caso de una cola local llamada TESTQ en el gestor de colas TEST. Esto se representa mediante el archivo de IFS:

```
/QIBM/UserData/mqm/qmgrs/TEST/queues
```

Si un mensaje especificado se transfiere a esta cola y luego se recupera de la cola, las acciones que tienen lugar se muestran en la Figura 22 en la página 415.

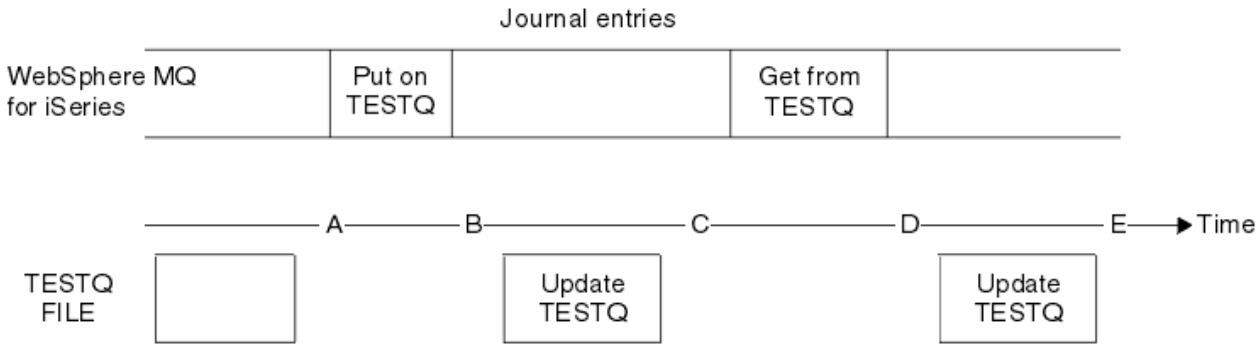


Figura 22. Secuencia de sucesos que se producen al actualizar objetos de MQM

Los cinco puntos, del A al E, que se muestran en el diagrama representan momentos que definen los siguientes estados:

A
La representación del archivo de IFS de la cola es coherente con la información contenida en el diario.

B
Se graba una entrada de diario en el diario que define una operación `Put` en la cola.

C
Se realiza la correspondiente actualización en la cola.

D
Se graba una entrada de diario en el diario que define una operación `Get` de la cola.

E
Se realiza la correspondiente actualización en la cola.

La clave de las posibilidades de recuperación de IBM MQ for IBM i es que el usuario puede guardar la representación del archivo IFS de TESTQ como es en el momento A y, posteriormente, recuperar la representación del archivo IFS de TESTQ como es en el momento E, restaurando el objeto guardado y reproduciendo las entradas del diario a partir del momento A.

IBM MQ for IBM i utiliza esta estrategia para recuperar los mensajes persistentes después de una anomalía del sistema. IBM MQ recuerda una entrada determinada de los receptores de diario y garantiza que en el momento del inicio reproducirá las entradas de los diarios a partir de ese punto. Esta entrada de inicio se recalcula periódicamente para que IBM MQ sólo tenga que realizar la reproducción mínima necesaria en el próximo inicio.

IBM MQ proporciona recuperación individual de objetos. Toda la información persistente relacionada con un objeto se registra en los diarios locales de IBM MQ for IBM i. Cualquier objeto de IBM MQ que quede dañado o deteriorado se puede reconstruir completamente a partir de la información guardada en el diario.

Para obtener más información sobre el modo en que el sistema maneja los receptores, consulte “Disponibilidad, copia de seguridad, recuperación y reinicio en IBM i” en la página 412.

IBM i *Imágenes de soporte en IBM i*

En IBM i, una imagen de soporte es una copia completa de un objeto IBM MQ que se registra en el diario. Algunos objetos corrompidos o dañados se pueden recuperar de forma automática a partir de su imagen de soporte.

Un objeto de IBM MQ de larga duración puede representar un gran número de entradas de diario, que se remontan al momento de su creación. Para evitarlo, existe en IBM MQ for IBM i el concepto de imagen de soporte de un objeto.

Esta imagen de soporte es una copia completa del objeto de IBM MQ registrado en el diario. Si se toma una imagen de un objeto, el objeto se puede reconstruir reproduciendo las entradas de diario desde la imagen en adelante. La entrada de diario que representa el punto de reproducción de cada objeto de IBM MQ se conoce como entrada de recuperación de soporte. IBM MQ hace un seguimiento de lo siguiente:

- La entrada de recuperación desde soporte para cada objeto del gestor de colas.
- La entrada más antigua de este conjunto (consulte el mensaje de error AMQ7462 de "Gestión de diarios en IBM i" en la página 418 para obtener más detalles).

Periódicamente se toman imágenes del objeto *CTLG y del objeto *MQM, ya que estos objetos son esenciales para el reinicio del gestor de colas.

Las imágenes de los demás objetos se toman cuando sea conveniente. De forma predeterminada, se toman imágenes de todos los objetos cuando se cierra un gestor de colas usando el mandato **ENDMQM** con el parámetro ENDCCTJOB(*YES). Esta operación puede requerir una cantidad de tiempo considerable si se trata de gestores de colas muy grandes. Si necesita cerrarlo rápidamente, especifique el parámetro RCDMQMIMG(*NO) con ENDCCTJOB(*YES). En estos casos, es recomendable registrar una imagen de soporte completa en los diarios después de que se haya reiniciado el gestor de colas, mediante el siguiente mandato:

```
RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) MQMNAME(Q_MGR_NAME)
```

IBM MQ registra automáticamente una imagen de un objeto si encuentra un punto adecuado en el que un objeto se pueda describir de forma compacta realizando una pequeña entrada en el diario. Sin embargo, puede que esto no suceda nunca para algunos objetos, como son las colas que contienen siempre grandes cantidades de mensajes.

En lugar de permitir que la fecha de la entrada de recuperación de soporte más antigua continúe durante un periodo innecesariamente largo, utilice el mandato IBM MQ RCDMQMIMG, que le permite tomar una imagen de los objetos seleccionados manualmente.

Recuperación a partir de imágenes de soporte

IBM MQ recupera automáticamente algunos objetos a partir de su imagen de soporte si se detecta que los objetos están dañados o deteriorados. En particular, esto se aplica a los objetos especiales *MQM y *CTLG como parte del arranque normal del gestor de colas. Si en el momento del último cierre del gestor de colas quedó incompleta alguna transacción de punto de sincronismo, también se recupera automáticamente cualquier cola afectada, con el fin de completar la operación de arranque.

Debe recuperar otros objetos manualmente, utilizando el mandato RCRMQMOBJde IBM MQ . Este mandato reproduce las entradas del diario para volver a crear el objeto de IBM MQ. Si un objeto de IBM MQ resulta dañado, las únicas acciones válidas son suprimirlo o crearlo de nuevo con este método. Sin embargo, tenga en cuenta que los mensajes no persistentes no se pueden recuperar de este modo.

IBM i *Puntos de comprobación en IBM MQ for IBM i*

Los puntos de comprobación se toman a diferentes horas para proporcionar un punto de inicio coherente y conocido para la recuperación.

La hebra de punto de comprobación en el proceso AMQZMUC0 se encarga de tomar el punto de comprobación en los puntos siguientes:

- Arranque del gestor de colas (STRMQM).
- Cierre del gestor de colas (ENDMQM).
- Cuando haya transcurrido una cantidad de tiempo desde el último punto de comprobación (el periodo predeterminado son 30 minutos) y se haya grabado un número mínimo de registros de anotaciones desde el último punto de comprobación (el valor predeterminado es 100).
- Después de que se haya grabado un número de registros de anotaciones. El valor predeterminado es 10 000.
- Después de que se haya sobrepasado el tamaño umbral de diario y se haya creado un receptor de nuevo diario.
- Cuando se toma una imagen de soporte con:

```
RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) MQMNAME(Q_MGR_NAME) DSPJRNDTA(*YES)
```

IBM i Copias de seguridad de datos de IBM MQ for IBM i

Utilice esta información para conocer los dos tipos de copias de seguridad de IBM MQ para cada gestor de colas.

Para cada gestor de colas, hay dos tipos de copia de seguridad de IBM MQ a tener en cuenta:

- Copia de seguridad de datos y de diario.

  Para asegurar que ambos tipos de datos sean compatibles, haga esta copia de seguridad únicamente después de cerrar el gestor de colas.
- Copia de seguridad de diario.

  Puede hacerse mientras el gestor de colas está activo.

Para ambos métodos, debe encontrar los nombres del directorio IFS del gestor de colas y de la biblioteca del gestor de colas. Puede encontrarlos en el archivo de configuración de IBM MQ (mqs.ini). Para obtener más información, consulte Modificación de la información de configuración de IBM MQ en Multiplatforms.

Utilice los procedimientos siguientes para realizar ambos tipos de copia de seguridad:

Copia de seguridad de datos y de diario de un determinado gestor de colas

Nota: No utilice una solicitud de guardar-mientras-está-activo cuando el gestor de colas esté ejecutándose. No se puede completar una solicitud de este tipo a menos que se hayan confirmado o retrotraído todas las definiciones de compromiso que tengan cambios pendientes. Si se utiliza este mandato cuando el gestor de colas está activo, las conexiones de los canales podrían no finalizar con normalidad. Utilice siempre el procedimiento siguiente.

1. Cree un receptor de diario vacío, utilizando el siguiente mandato:

   ```
   CHGJRN JRN(QMTEST/AMQAJRN) JRNRCV(*GEN)
   ```
2. Utilice el mandato **RCDMQMIMG** para registrar una imagen MQM para todos los objetos IBM MQ y, a continuación, fuerce un punto de comprobación utilizando el mandato:

   ```
   RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) DSPJRNDTA(*YES) MQMNAME(TEST)
   ```
3. Finalice los canales y asegúrese de que el gestor de colas no está ejecutándose. Si el gestor de colas se está ejecutando, deténgalo con el mandato **ENDMQM**.
4. Haga copia de seguridad de la biblioteca del gestor de colas emitiendo el siguiente mandato:

   ```
   SAVLIB LIB(QMTEST)
   ```
5. Haga copia de seguridad de los directorios de IFS del gestor de colas emitiendo el siguiente mandato:

   ```
   SAV DEV(...) OBJ(('/QIBM/UserData/mqm/qmgrs/test'))
   ```

Copia de seguridad de diario de un determinado gestor de colas

En el supuesto de que en algún momento haya hecho una operación de guardar completa y debido a que toda la información relevante se guarda en los diarios, puede realizar copias de seguridad parciales guardando los receptores de diario. Estas registran todos los cambios realizados a partir del momento de la copia de seguridad completa y, para ello, se emiten los siguientes mandatos:

1. Cree un receptor de diario vacío, utilizando el siguiente mandato:

```
CHGJRN JRN(QMTEST/AMQAJRN) JRNRCV(*GEN)
```

2. Utilice el mandato **RCDMQMIMG** para registrar una imagen MQM para todos los objetos IBM MQ y, a continuación, fuerce un punto de comprobación utilizando el mandato:

```
RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) DSPJRNDTA(*YES) MQMNAME(TEST)
```

3. Guarde los receptores de diario utilizando el siguiente mandato:

```
SAVOBJ OBJ(AMQ*) LIB(QMTEST) OBJTYPE(*JRNRCV) .........
```

Una estrategia de copia de seguridad sencilla consiste en realizar todas las semanas una copia de seguridad completa de las bibliotecas de IBM MQ y hacer todos los días una copia de seguridad de los diarios. Esto, como es natural, depende de cómo haya configurado la estrategia de copia de seguridad de la empresa.

IBM i *Gestión de diarios en IBM i*

Como parte de la estrategia de copia de seguridad, conviene que se ocupe de los receptores de diario. Es conveniente eliminar los receptores de diario de las bibliotecas de IBM MQ por varias razones:

- Para liberar espacio; esto es válido para todos los receptores de diario
- Para mejorar el rendimiento al arrancar (STRMQM)
- Para mejorar el rendimiento al volver a crear objetos (RCRMQMOBJ)

Antes de suprimir un receptor de diario, asegúrese de que dispone de una copia y de que ya no necesita el receptor de diario.

Los receptores de diario se pueden eliminar de la biblioteca del gestor de colas *después* de desconectarlos de los diarios y de guardarlos, siempre que estén disponibles para restauración si se necesitan en una operación de recuperación.

El concepto de la gestión de diarios se muestra en la Figura 23 en la página 419.

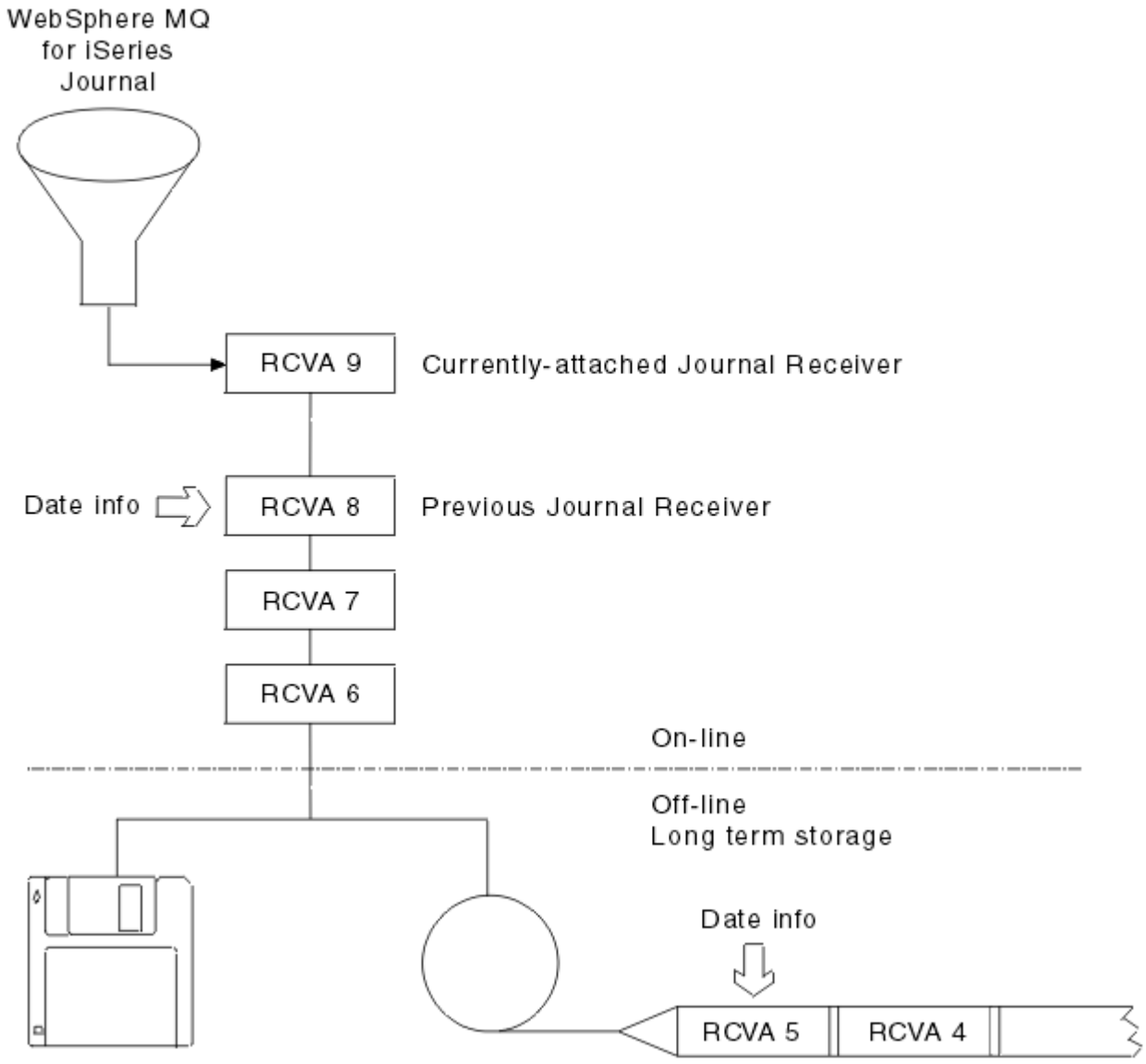


Figura 23. Registro por diario en IBM i

Conviene saber a qué punto anterior de los diarios es probable que tenga que retroceder IBM MQ para poder determinar cuándo se puede eliminar un receptor de diario de la biblioteca del gestor de colas cuya copia de seguridad se haya realizado y cuándo se puede descartar la propia copia de seguridad.

IBM MQ emite dos mensajes a la cola de mensajes del gestor de colas (QMQMMSG en la biblioteca del gestor de colas) para ayudar a determinar ese momento. Estos mensajes se emiten cuando se inicia, cuando cambia un receptor de diario local y se utiliza RCDMQIMG para forzar un punto de comprobación. Los dos mensajes son:

AMQ7460
Punto de recuperación de arranque. Este mensaje define la fecha y hora de la entrada de arranque a partir de la cual IBM MQ reproduce el diario en el caso de que se produzca un paso de recuperación de arranque. Si el receptor de diario que contiene este registro está disponible en las bibliotecas de IBM MQ, este mensaje también incluye el nombre del receptor de diario que contiene el registro.

AMQ7462
Entrada más antigua de recuperación desde soporte. Este mensaje define la fecha y hora de la entrada más antigua que se utilizará para volver a crear un objeto a partir de su imagen de soporte.

El receptor de diario identificado es el más antiguo que se necesita. Los receptores de diario de IBM MQ cuyas fechas de creación sean todavía más antiguas ya no se necesitan. Si sólo se visualizan

asteriscos, es preciso restaurar las copias de seguridad a partir de la fecha indicada para determinar cuál es el receptor de diario más antiguo.

Cuando se anotan estos mensajes, IBM MQ también graba un objeto de espacio de usuario en la biblioteca del gestor de colas que contiene una sola entrada: el nombre del receptor de diario más antiguo que debe conservarse en el sistema. Este espacio de usuario se denomina AMQJRNINF y los datos se graban en el siguiente formato:

```
JJJJJJJJJJLLLLLLLLLLYYYYMMDDHHMMSSmmm
```

donde:

JJJJJJJJJJ
Es el nombre del receptor más antiguo que IBM MQ todavía necesita.

LLLLLLLLLL
Es el nombre de la biblioteca de receptores de diario.

YYYY
Es el año de la entrada de diario más antigua que IBM MQ necesita.

MM
Es el mes de la entrada de diario más antigua que IBM MQ necesita.

DD
Es el día de la entrada de diario más antigua que IBM MQ necesita.

HH
Es la hora de la entrada de diario más antigua que IBM MQ necesita.

SS
Son los segundos de la entrada de diario más antigua que IBM MQ necesita.

mmm
Son los milisegundos de la entrada de diario más antigua que IBM MQ necesita.

Una vez que se ha eliminado del sistema el receptor de diario más antiguo, este espacio de usuario contiene asteriscos (*) para el nombre del receptor de diario.

Nota: La realización periódica de `RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) DSPJRNDTA(*YES)` puede ahorrar tiempo de inicio para IBM MQ y reducir el número de receptores de diario locales que necesita salvar y restaurar para la recuperación.

IBM MQ for IBM i no hace referencia a los receptores de diario a menos que esté realizando un paso de recuperación, ya sea para el inicio o para volver a crear un objeto. Si detecta que un diario necesario no está presente, emite el mensaje AMQ7432 a la cola de mensajes del gestor de colas (QMQMMSG) para notificar la fecha y hora de la entrada de diario que necesita para completar el paso de recuperación.

Si esto ocurriera, restaure a partir de la copia de seguridad todos los receptores de diario desconectados después de esa fecha, para permitir que sea satisfactorio el paso de recuperación.

Mantenga el receptor de diario que contiene la entrada de arranque, y todos los receptores de diario posteriores, disponibles en la biblioteca del gestor de colas.

Mantenga el receptor de diario que contiene el `Media Recovery Entry`más antiguo, y cualquier receptor de diario posterior, disponible en todo momento, y presente en la biblioteca del gestor de colas o de la que se haya realizado una copia de seguridad.

Cuando se fuerza un punto de comprobación:

- Si el receptor de diario especificado en AMQ7460 no está avanzado, indica que hay una unidad de trabajo incompleta que debe confirmarse o restituirse.
- Si el receptor de diario especificado en AMQ7462 no está avanzado, indica hay uno o más objetos dañados.

IBM i *Restauración de un gestor de colas completo (datos y diarios) en IBM i*

Utilice esta información para restaurar uno o más gestores de colas de una copia de seguridad o desde una máquina remota.

Si tiene que recuperar uno o varios gestores de colas de IBM MQ a partir de una copia de seguridad, siga estos pasos:

1. Desactive temporalmente los gestores de colas de IBM MQ.
2. Localice el juego de copias de seguridad más reciente, que consta de la última copia de seguridad completa y de los receptores de diario de los que posteriormente se ha hecho copia de seguridad.
3. Realice una operación RSTLIB, a partir de la copia de seguridad completa, para restaurar las bibliotecas de datos de IBM MQ al estado que tenían en el momento de la copia de seguridad completa; para ello, emita estos mandatos:

   ```
   RSTLIB LIB(QMQRLIB1) .........
   RSTLIB LIB(QMQRLIB2) .........
   ```

   Si un receptor de diario se ha guardado parcialmente en una copia de seguridad de diario y luego se ha guardado completamente en una copia de seguridad posterior, restaure sólo el que se ha guardado completamente. Restaure los diarios de forma individual y por orden cronológico.
4. Realice una operación RST para restaurar los directorios IFS de IBM MQ en el sistema de archivos IFS; para ello, emita el siguiente mandato:

   ```
   RST DEV(...) OBJ(('/QIBM/UserData/mqm/qmgrs/testqm')) ...
   ```

5. Inicie el gestor de colas de mensajes. Se reproducen todos los registros de diario grabados desde la copia de seguridad completa y se restauran todos los objetos de IBM MQ al estado coherente que tenían en el momento de la copia de seguridad del diario.

Si desea restaurar un gestor de colas completo en una máquina distinta, utilice el procedimiento siguiente para restaurarlo todo desde la biblioteca del gestor de colas. (TEST es el nombre que se utiliza para el gestor de colas de ejemplo.)

1. `CRTMQM TEST`
2. `DLTLIB LIB(QMTEST)`
3. `RSTLIB SAVLIB(QMTEST) DEV(*SAVF) SAVF(QMGRLIBSAV)`
4. Suprima los siguientes archivos IFS:

   ```
   /QIBM/UserData/mqm/qmgrs/TEST/QMQMCHKPT
   /QIBM/UserData/mqm/qmgrs/TEST/qmanager/QMQMOBJCAT
   /QIBM/UserData/mqm/qmgrs/TEST/qmanager/QMANAGER
   /QIBM/UserData/mqm/qmgrs/TEST/queues/SYSTEM.AUTH.DATA.QUEUE/q
   /QIBM/UserData/mqm/qmgrs/TEST/queues/SYSTEM.CHANNEL.INITQ/q
   /QIBM/UserData/mqm/qmgrs/TEST/queues/SYSTEM.CLUSTER.COMMAND.QUEUE/q
   /QIBM/UserData/mqm/qmgrs/TEST/queues/SYSTEM.CLUSTER.REPOSITORY.QUEUE/q
   /QIBM/UserData/mqm/qmgrs/TEST/queues/SYSTEM.CLUSTER.TRANSMIT.QUEUE/q
   /QIBM/UserData/mqm/qmgrs/TEST/queues/SYSTEM.PENDING.DATA.QUEUE/q
   /QIBM/UserData/mqm/qmgrs/TEST/queues/SYSTEM.ADMIN.COMMAND.QUEUE/q
   ```

5. `STRMQM TEST`
6. `RCRMQMOBJ OBJ(*ALL) OBJTYPE(*ALL) MQMNAME(TEST)`

IBM i *Restauración de los receptores de diario de un determinado gestor de colas en IBM i*

Utilice esta información para conocer los diferentes procedimientos para restaurar receptores de diario.

Si se ha eliminado un receptor que se necesita otra vez para una función de recuperación ulterior, la acción más corriente consiste en restaurar en una biblioteca del gestor de colas un receptor de diario de una copia de seguridad.

Ésta es una tarea simple, en la que los receptores de diario se deben restaurar con el mandato estándar de IBM i RSTOBJ:

```
RSTOBJ OBJ(QMQMDATA/AMQA000005) OBJTYPE(*JRNRCV) .........
```

Puede suceder que se tenga que restaurar una serie de receptores de diario, en lugar de tan solo un receptor. Por ejemplo, supongamos que AMQA000007 es el receptor más antiguo en las bibliotecas de IBM MQ y que se tienen que restaurar los receptores AMQA000005 y AMQA000006.

En este caso, restaure los receptores individualmente en orden cronológico inverso. No siempre es necesario hacerlo así, pero es muy conveniente. En situaciones graves, es posible que necesite utilizar el mandato de IBM i WRKJRNA para asociar los receptores de diario restaurados al diario.

Cuando se restauran diarios, el sistema crea automáticamente un receptor de diario conectado dándole un nuevo nombre en la secuencia de receptores de diario. Sin embargo, el nuevo nombre generado podría no coincidir con el del receptor de diario que se debe restaurar. Para salir al paso de este problema, se necesita intervención manual para crear en secuencia un receptor de diario con un nombre nuevo y un diario nuevo antes de restaurar el receptor de diario.

Por ejemplo, supongamos que surge este problema con respecto al diario guardado AMQAJRN y estos receptores de diario:

- AMQA000000
- AMQA100000
- AMQA200000
- AMQA300000
- AMQA400000
- AMQA500000
- AMQA600000
- AMQA700000
- AMQA800000
- AMQA900000

Al restaurar el diario AMQAJRN en una biblioteca del gestor de colas, el sistema crea automáticamente el receptor de diario AMQA000000. Este receptor generado de forma automática está en conflicto con uno de los receptores de diario existentes (AMQA000000) que desea restaurar y que será imposible hacerlo.

La solución sería:

1. Crear manualmente el siguiente receptor de diario (vea “Diarios del gestor de colas en IBM i” en la página 412):

   ```
   CRTJRNRCV JRNRCV(QMQRLIB/AMQA900001) THRESHOLD(XXXXX)
   ```

2. Crear manualmente el diario con el receptor de diario:

   ```
   CRTJRN JRN(QMGRLIB/AMQAJRN) MNGRCV(*SYSTEM) +
   JRNRCV(QMGRLIB/AMQA9000001) MSGQ(QMGRLIB/AMQAJRNMSG)
   ```

3. Restaurar los receptores de diario locales comprendidos entre AMQA000000 y AMQA900000.

IBM i Gestores de colas multiinstancia en IBM i

Los gestores de colas multiinstancia mejoran la disponibilidad conmutando automáticamente a un servidor en espera si el servidor activo falla. Los servidores activos y en espera son instancias múltiples del mismo gestor de colas; comparten los mismos datos del gestor de colas. Si la instancia activa falla, debe transferir su diario a la que está en espera pues asume que el gestor de colas puede volver a estructurar sus colas.

Configure los sistemas IBM i en los que está ejecutando gestores de colas multiinstancia para que, si la instancia activa de gestor de colas falla, el diario que está utilizando esté disponible para la instancia en espera que la sustituye. Puede diseñar sus propias tareas de configuración y administración para que el diario de la instancia activa esté disponible para la instancia que la sustituye. Si no desea perder mensajes, su diseño debe garantizar que el diario en espera sea consistente con el diario activo cuando se produzca la anomalía. Puede adaptar su diseño a partir de una de las dos configuraciones que se describen con ejemplos en los temas subsiguientes para actualizar la coherencia.

1. Duplique el diario del sistema que está ejecutando la instancia del gestor de colas activa en los sistemas que están ejecutando las instancias en espera.
2. Coloque el diario en una agrupación de almacenamiento auxiliar independiente (Independent Auxiliary Storage Pool - IASP) que se pueda transferir desde el sistema que ejecuta la instancia activa a la instancia en espera.

La primera solución no precisa hardware ni software adicional puesto que utiliza ASP básicas. La segunda solución necesita IASP conmutables que necesitan el soporte de agrupación en clúster de IBM i que está disponible como producto con licencia de IBM i (5761-SS1) opción 41 que se vende por separado.

IBM i *Fiabilidad y disponibilidad en IBM i*

El objetivo de los gestores de colas multiinstancia es mejorar la disponibilidad de las aplicaciones. Las restricciones tecnológicas y físicas apuntan que el usuario necesita distintas soluciones para poder resarcirse de una recuperación tras desastre, haciendo copias de seguridad de los gestores de colas y garantizando un funcionamiento continuo.

Al configurar la fiabilidad y la disponibilidad debe evaluar un gran número de factores que podrían resumirse en cuatro puntos característicos:

Recuperación tras desastre
Se ha optimizado para la recuperación después de un desastre importante que pueda destruir todos los activos locales.

La recuperación tras desastre en IBM i a menudo se basa en la duplicación geográfica de IASP.

Reserva
Se ha optimizado para poder efectuar la recuperación tras una anomalía localizada, que habitualmente se trata de un error humano o de un problema técnico imprevisto.

IBM MQ proporciona gestores de colas de copia de seguridad para hacer copia de seguridad de los gestores de colas de forma periódica. También puede utilizar la réplica asíncrona de diarios del gestor de colas para mejorar la efectividad de las copias de seguridad.

Disponibilidad
Se ha optimizado para restaurar rápidamente operaciones de forma que adquieran la apariencia de un servicio casi ininterrumpido pues pueden anticiparse a anomalías técnicas previsibles tales como un error de disco o de servidor.

Normalmente, la recuperación se mide en minutos, pero la detección a veces se prolonga más que el propio proceso de recuperación. Un gestor de colas multiinstancia le ayuda a configurar la *disponibilidad*.

Operación continua
Se ha optimizado para suministrar un servicio ininterrumpido.

Las soluciones de operación continua deben solucionar el problema de detección, lo que casi siempre implica someter el mismo trabajo a través de más de un sistema y, o bien utilizar el primer resultado o, si la exactitud de los datos constituye el objetivo principal, comparar como mínimo dos resultados.

Un gestor de colas multiinstancia le ayuda a configurar la *disponibilidad*. Sólo hay una instancia del gestor de colas activa a la vez. El cambiar a una instancia en espera se puede prolongar desde un poco más de diez segundos hasta unos quince minutos o más, en función de cómo se haya configurado, cargado y ajustado el sistema.

Un gestor de colas de varias instancias puede dar la apariencia de un servicio casi ininterrumpido si se utiliza con IBM MQ MQI clients reconectables, que son capaces de continuar el proceso sin que el programa de aplicación sepa realmente que se ha producido una interrupción del gestor de colas; consulte el tema Reconexión de cliente automatizada.

IBM i *Componentes de una solución de alta disponibilidad en IBM i*

Construya una solución de alta disponibilidad utilizando gestores de colas gestor de colas multiinstancia proporcionando un almacenamiento por red robusto para los datos del gestor de colas, réplica de diarios o almacenamiento IASP robusto para diarios de gestor de colas y utilizando clientes que se puedan volver a conectar (de aplicaciones configuradas como servicios de gestor de colas reiniciables).

Al detectar una anomalía del gestor de colas, un gestor de colas multiinstancia reacciona reanudando el inicio de otra instancia de gestor de colas en otro servidor. Para completar dicho inicio, la instancia debe acceder a los datos del gestor de colas compartidos en almacenamiento por red y a su copia del diario de gestor de colas local.

Para crear una solución de alta disponibilidad, debe gestionar la disponibilidad de los datos del gestor de colas, la vigencia del diario del gestor de colas local y, o bien crear aplicaciones cliente con capacidad de reconexión, o bien desplegar las aplicaciones como servicios de gestor de colas para que se reinicien automáticamente cuando se reanude el gestor de colas. La reconexión automática de cliente no está soportada en IBM MQ classes for Java.

Datos del gestor de colas

Coloque los datos del gestor de colas en almacenamiento por red que se comparta, altamente disponible y fiable, posiblemente utilizando discos RAID de nivel 1 o posteriores. El sistema de archivos debe cumplir con los requisitos de un sistema de archivos compartidos para gestores de colas multiinstancia; con objeto de obtener más información acerca de los requisitos de los sistemas de archivos compartidos, consulte Requisitos de los sistemas de archivos compartidos. Network File System 4 (NFS4) es un protocolo que cumple con estos requisitos.

Diarios de gestor de colas

También debe configurar los diarios de IBM i utilizados por las instancias del gestor de colas de forma que la instancia en espera pueda restaurar sus datos del gestor de colas a un estado coherente. En caso de interrupción del servicio, significa que deberá restaurar los diarios a su estado cuando la instancia activa falle. A diferencia de las soluciones de copia de seguridad o de recuperación ante siniestro, la restauración de diarios en un punto de comprobación anterior no es suficiente.

No puede compartir físicamente diarios entre varios sistemas IBM i en almacenamiento por red. Para restaurar diarios del gestor de colas a un estado coherente en el momento en que se produce la anomalía, o bien deberá transferir el diario físico que era local para la instancia del gestor de colas activa en el momento de la anomalía a una instancia nueva que se haya activado, o bien deberá actualizar duplicados del diario en instancias en espera en ejecución. El diario duplicado es una réplica del diario remoto que se ha mantenido exactamente en sincronización con el diario local que pertenece a la instancia fallida.

Existen tres configuraciones que constituyen puntos de inicio para designar cómo se gestionan los diarios para un gestor de colas multiinstancia,

1. la utilización de la réplica de diarios sincronizada (duplicación de diarios) desde la ASP de la instancia activa a las ASP de instancias en espera.

2. al transferir una IASP, ha efectuado una configuración para mantener el diario del gestor de colas desde la instancia activa a la instancia en espera que va a adoptar el papel de instancia activa.
3. la utilización duplicados de IASP secundarios sincronizados.

Consulte las opciones de ASP, para obtener más información sobre cómo poner los datos del gestor de colas en un iASP, en el mandato CRTMQM de IBM MQ IBM i.

Consulte también Alta disponibilidad en la información de IBM i en IBM Documentation.

Aplicaciones

Para crear un cliente para que se reconecte automáticamente al gestor de colas cuando el gestor de colas en espera se reanuda, conecte su aplicación al gestor de colas utilizando MQCONNX y especifique MQCNO_RECONNECT_Q_MGR en el campo **MQCNO** Opciones. Consulte Programas de ejemplo de alta disponibilidad para obtener tres programas de muestra utilizando clientes con capacidad de reconexión y Recuperación de la aplicación para obtener información sobre cómo diseñar aplicaciones cliente para la recuperación.

IBM i *Creación de una unidad compartida de datos de un gestor de colas utilizando NetServer en IBM i*

Cree una unidad compartida de red en un servidor IBM i para almacenar los datos de gestor de colas. Configure las conexiones de dos servidores, en los que se vayan a albergar la instancias del gestor de colas, para poder acceder a la unidad compartida de red.

Antes de empezar

- Para esta tarea se necesitan tres servidores IBM i. La unidad compartida de red se define en uno de los servidores, GAMMA. Los otros dos servidores, ALPHA y BETA, se conectarán a GAMMA.
- Instale IBM MQ en los tres servidores.
- Instale System i Navigator; consulte System i Navigator.

Acerca de esta tarea

- Cree el directorio del gestor de colas en GAMMA, y establezca la propiedad y los permisos correctos para los perfiles de usuario QMQM y QMQMADM. El directorio y los permisos se crean fácilmente instalando IBM MQ en GAMMA.
- Utilice System i Navigator para crear una unidad compartida para el directorio de datos del gestor de colas en GAMMA.
- Cree directorios en ALPHA y BETA que señalan al compartimiento.

Procedimiento

1. En GAMMA, cree el directorio en el que albergar los datos de gestor de colas con el perfil de usuario QMQM como propietario, y QMQMADM como el grupo primario.

   Consejo:

   Una forma rápida y fiable de crear el directorio con los permisos correctos es instalar IBM MQ en GAMMA.

   Posteriormente, si no desea ejecutar IBM MQ en GAMMA, desinstale IBM MQ. Tras la desinstalación, el directorio /QIBM/UserData/mqm/qmgrs permanece en GAMMA con el perfil de usuario QMQM como propietario, y QMQMADM como grupo primario.

   La tarea utiliza el directorio /QIBM/UserData/mqm/qmgrs en GAMMA para el compartimiento.
2. Inicie el asistente System i Navigator **Añadir conexión** y conéctese al sistema GAMMA.
   a) Efectúe una doble pulsación en el icono **System i Navigator** del escritorio de Windows.
   b) Pulse **Sí** para crear una conexión.

c) Siga las instrucciones del asistente **Añadir conexión** y cree una conexión del sistema IBM i a GAMMA.

La conexión a GAMMA se añade a **Mis conexiones**.

3. Añada un compartimiento de archivos nuevo en GAMMA.
   a) En la ventana **System i Navigator**, pulse la carpeta File Shares en My Connections/GAMMA/File Systems.
   b) En la ventana **Mis tareas**, pulse **Gestionar compartimientos de IBM i NetServer**.

      En el escritorio se abre una nueva ventana, **IBM i NetServer - GAMMA**, en la que se muestran los objetos compartidos.
   c) Pulse con el botón derecho del ratón en la carpeta Shared Objects > **Archivo** > **Nuevo** > **Archivo**.

      Se abre una nueva ventana **Compartición de archivos de IBM i NetServer - GAMMA**.
   d) Asigne un nombre a la unidad compartida, por ejemplo, WMQ.
   e) Establezca el control de acceso en Read/Write.
   f) Seleccione **Nombre de vía de acceso** examinando el directorio /QIBM/UserData/mqm/qmgrs que ha creado anteriormente y pulse **Aceptar**.

      La ventana **Compartimiento de archivos de IBM i NetServer-GAMMA** se cierra y WMQ se lista en la ventana de objetos compartidos.
4. Pulse el botón derecho del ratón **WMQ**, en la ventana de objetos compartidos. Pulse **Archivo** > **Permisos**.

   Se abre una ventana, **Permisos Qmgrs - GAMMA**, para el objeto /QIBM/UserData/mqm/qmgrs.
   a) Compruebe los permisos siguientes para QMQM, si todavía no se han establecido:

      ```
      Read
      Write
      Execute
      Management
      Existence
      Alter
      Reference
      ```
   b) Compruebe los permisos siguientes para QMQMADM, si todavía no se han establecido:

      ```
      Read
      Write
      Execute
      Reference
      ```
   c) Añada otros perfiles de usuario que desee otorgar a /QIBM/UserData/mqm/qmgrs.

      Por ejemplo, puede otorgar al perfil de usuario predeterminado (Public) los permisos Read y Execute para /QIBM/UserData/mqm/qmgrs.
5. Compruebe que todos los perfiles de usuario a los que se ha otorgado acceso a /QIBM/UserData/mqm/qmgrs en GAMMA tengan la misma contraseña que en los servidores que acceden a GAMMA.

   En concreto, asegúrese de que los perfiles de usuario QMQM en otros servidores, que vayan a acceder a la unidad compartida, tengan la misma contraseña que el perfil de usuario QMQM en GAMMA.

   Consejo: Pulse la carpeta My Connections/GAMMA/Users and Groups en System i Navigator para establecer las contraseñas. También puede utilizar los mandatos **CHFUSRPRF** y **CHGPWD**.

Resultados

Compruebe que pueda acceder a GAMMA, desde otros servidores, mediante la unidad compartida. Si está realizando las demás tareas, compruebe que puede acceder a GAMMA desde ALPHA y BETA utilizando la vía de acceso `/QNTC/GAMMA/WMQ`. Si el directorio `/QNTC/GAMMA` no existe en ALPHA o BETA, debe crear el directorio. En función del dominio de NetServer, deberá realizar una IPL en ALPHA o en BETA, antes de crear el directorio.

```
CRTDIR DIR('/QNTC/GAMMA')
```

Cuando haya comprobado que tiene acceso a `/QNTC/GAMMA/WMQ` desde ALPHA o BETA, emitiendo el mandato, `CRTMQM MQMNAME('QM1') MQMDIRP('/QNTC/GAMMA/WMQ')` crea `/QIBM/UserData/mqm/qmgrs/QM1` en GAMMA.

Qué hacer a continuación

Cree un gestor de colas multiinstancia siguiendo los pasos que figuran en cualquiera de las tareas, "Creación de un gestor de colas multiinstancia mediante la duplicación de diarios y NetServer en IBM i" en la página 438 o "Conversión de un gestor de colas de instancia única en un gestor de colas multiinstancia usando NetServer y duplicación de diarios en IBM i" en la página 442.

IBM i *Rendimiento de una migración tras error en IBM i*

El tiempo que se tarda en detectar si una instancia de gestor de colas ha fallado y, a continuación, para reanudar el procesamiento en modo de espera puede variar entre decenas de segundos hasta quince minutos o más, dependiendo de la configuración. El rendimiento debe ser un importante punto a tener en cuenta al diseñar y probar una solución de alta disponibilidad.

Existen ventajas y desventajas que se deben sopesar al decidir si configurar un gestor de colas multiinstancia para que utilice la réplica de diario o si utilizar una IASP. La duplicación necesita que el gestor de colas grabe síncronamente en un diario remoto. Desde el punto de vista del hardware, esta necesidad no afecta al rendimiento, pero desde el punto de vista del software se implica una longitud de vía de acceso superior al escribir en un diario remoto que si sólo se hace en un diario local, por lo que puede deducirse que el rendimiento se verá de alguna manera reducido al ejecutar un gestor de colas. No obstante, cuando el gestor de colas en espera realiza la sustitución, el retardo al sincronizar su diario local desde el diario remoto actualizado por la instancia activa antes de su fallo, normalmente resulta pequeño en comparación con el tiempo que IBM i necesita para detectar y transferir la IASP al servidor que ejecuta la instancia en espera del gestor de colas. Los tiempos de transferencia de la IASP pueden alcanzar de diez a quince minutos, en lugar de completarse en segundos. El tiempo de transferencia de la IASP depende del número de objetos que deban estar *disponibles* cuando la IASP se transfiere al sistema en espera y del tamaño de las vías de acceso o los índices, puesto que deben fusionarse.

Cuando el gestor de colas en espera realiza la sustitución, el retardo al sincronizar su diario local desde el diario remoto actualizado por la instancia activa antes de su fallo, normalmente resulta pequeño en comparación con el tiempo que IBM i necesita para detectar y transferir la ASP independiente al servidor que ejecuta la instancia en espera del gestor de colas. Los tiempos de transferencia de ASP independientes pueden alcanzar de diez a quince minutos, en lugar de completarse en segundos. El tiempo de transferencia de la ASP independiente depende del número de objetos que deban estar *disponibles* cuando la ASP independiente se transfiera al sistema en espera y del tamaño de las vías de acceso o los índices, puesto que deben fusionarse.

No obstante, la transferencia del diario no es el único factor que influye en el tiempo que la instancia en espera necesita para reanudarse de forma completa. También es necesario considerar el tiempo que tarda el sistema de archivos de red en liberar el bloqueo de los datos del gestor de colas que señala a la instancia en espera para tratar de continuar con su puesta en marcha, y también el tiempo necesario para recuperar las colas del diario para que la instancia pueda empezar a procesar los mensajes de nuevo. Deben añadirse todos estos otros orígenes de retardo al tiempo que se tarda en iniciar una instancia de espera. El tiempo total para cambiar consta de los siguientes componentes,

Tiempo de detección de anomalía
El tiempo que NFS necesita para liberar el bloqueo de los datos del gestor de colas y el que necesita la instancia en espera para continuar su proceso de inicio.

Tiempo de transferencia
En el caso de un clúster de alta disponibilidad (HA), el tiempo que IBM i tarda en transferir la IASP desde el sistema que hospeda la instancia activa a la instancia en espera, y en el caso de réplica de diario, el tiempo que tarda para actualizar el diario local en espera con los datos de la réplica remota.

Tiempo de inicio
El tiempo que tarda la nueva instancia del gestor de colas activa en reconstruir sus colas desde el último punto de comprobación en su diario restaurado y en reanudar el procesamiento de mensajes.

Nota:

Si la instancia en espera que ha que ha realizado la sustitución está configurada para replicarse de forma síncrona con la instancia anteriormente activa, el inicio podría retardarse durante mucho tiempo. Es posible que la instancia nueva activada no sea capaz de replicarse con su diario remoto, si el diario remoto está en el servidor que albergaba la instancia activa anteriormente y el servidor ha fallado.

El valor predeterminado de tiempo de espera para una respuesta síncrona es de un minuto. Puede configurar el retardo máximo antes de que la réplica caduque. O bien, puede configurar instancias en espera para iniciarse utilizando la réplica asíncrona en la instancia activa fallida. Más adelante, conmutará a la réplica síncrona, cuando la instancia fallida se esté ejecutando de nuevo en espera. La misma consideración se aplica al uso de duplicaciones ASP independientes síncronas.

Puede hacer mediciones de línea base separadas para estos componentes que le ayudarán a evaluar el tiempo global que le ha conducido hasta la sustitución por anomalía y a incidir en la decisión sobre el método de configuración que debe utilizar. Para tomar la mejor decisión en cuanto a la configuración, también debe tener en cuenta cómo harán la sustitución por anomalía las otras aplicaciones del mismo servidor y si hay otros procesos de copias de seguridad o de recuperación tras desastre que ya utilicen la IASP.

Los tiempos de transferencia de la IASP se pueden reducir ajustando la configuración del clúster:

1. Los perfiles de usuario de todos los sistemas del clúster deberían tener el mismo UID y GID para que no sea necesario que los procesos habilitados deban cambiar los UID y los GID.
2. Puede minimizar el número de objetos de base de datos del sistema y de agrupaciones de discos de usuario, ya que éstos deben ser fusionarse para crear la tabla de referencias cruzadas para el grupo de agrupación de discos.
3. Encontrará más consejos para mejorar el rendimiento en el IBM Redbook, *Implementing PowerHA for IBM i, SG24-7405*.

Una configuración utilizando ASP básicas, la duplicación de diarios y una pequeña configuración debería cambiar en nada más y nada menos que en decenas de segundos.

IBM i *Visión general de la combinación de las prestaciones de agrupación en clúster de IBM i con la agrupación en clúster de IBM MQ*

La ejecución de IBM MQ en IBM i y la utilización de las prestaciones de agrupación en clúster de IBM i puede proporcionar una solución de alta disponibilidad más completa que la utilización de la agrupación en clúster de IBM MQ únicamente.

Para tener esta prestación, debe configurar:

1. Los clústeres en la máquina de IBM i; consulte “Clústeres de IBM i” en la página 429
2. Una agrupación de almacenamiento auxiliar independiente (IASP), a la que mueve el gestor de colas; consulte “Agrupaciones de almacenamiento auxiliar independientes (IASP)” en la página 429
3. Un grupo de recursos de clúster (CRG); consulte “Grupos de recursos de clúster de dispositivos” en la página 429, en el que define:
   - El dominio de recuperación

- La IASP
- El programa de salida; consulte “Programa de salida de CRG de dispositivo” en la página 430

Clústeres de IBM i

Un clúster de IBM i es una colección de instancias, es decir, sistemas o particiones de IBM i que están enlazados lógicamente.

La finalidad de esta agrupación es permitir la copia de seguridad de cada una de las instancias, eliminando un único punto de anomalía y aumentado la flexibilidad de los datos y las aplicaciones. Una vez que se ha creado un clúster, pueden gestionarse los distintos tipos de grupo de recursos de clúster (CRG) para gestionar aplicaciones, datos y dispositivos en el clúster.

Para obtener más información, consulte Creación de un clúster y el mandato Crear clúster (CRTCLU).

Agrupaciones de almacenamiento auxiliar independientes (IASP)

Una IASP es un tipo de ASP de usuario que se utiliza como una extensión de un almacenamiento de un solo nivel. Es un elemento de almacenamiento que, debido a su independencia del almacenamiento del sistema, se puede manipular fácilmente sin tener que realizar una IPL del sistema.

Una IASP se puede conmutar fácilmente a otra instancia de sistema operativo o replicar en una IASP de destino en otra instancia de sistema operativo. Se pueden utilizar los métodos para conmutar una IASP entre instancias:

- El primer método requiere que todos los sistemas del clúster, y la torre de discos conmutables que contiene la IASP, estén conectados mediante un bucle HSL (High Speed Link).
- El segundo método requiere que las instancias del sistema operativo sean particiones en el mismo sistema de IBM i donde los procesadores de entrada/salida (IOP) se pueden conmutar entre particiones. No se requiere ningún hardware especial para poder replicar una IASP. La réplica se realiza mediante TCP/IP a través de la red.

Para obtener más información, consulte el mandato Configurar dispositivo ASP (CFGDEVASP).

Grupos de recursos de clúster de dispositivos

Hay distintos tipos de grupos de recursos de clúster (CRG). Para obtener más información sobre los distintos tipos de CRG disponibles, consulte Grupo de recursos de clúster.

Este tema se concentra en un CRG de dispositivo. Un CRG de dispositivo:

- Describe y gestiona los recursos de dispositivo como por ejemplo las agrupaciones de almacenamiento auxiliar independientes (IASP).
- Define el dominio de recuperación de los nodos de clúster
- Asigna un dispositivo, y
- Asigna el programa de salida que gestionará los sucesos del clúster.

El dominio de recuperación indica qué nodo del clúster se considerará el nodo primario. Los demás nodos se consideran copias de seguridad. Los nodos de copia de seguridad también están ordenados en el dominio de recuperación, y se especifica qué nodo es la primera copia de seguridad, la segunda copia de seguridad, etc., según el número de nodos que haya en el dominio de recuperación.

En caso de una anomalía de nodo primario, el programa de salida se ejecuta en todos los nodos del dominio de recuperación. A continuación, el programa de salida en ejecución en la primera copia de seguridad puede realizar las inicializaciones necesarias para convertir este nodo en el nuevo nodo primario.

Para obtener más información, consulte Creación de CRG de dispositivo y el mandato Crear grupo de recursos de clúster (CRTCRG).

Programa de salida de CRG de dispositivo

El servicio de recursos del clúster del sistema operativo llama a un programa de salida de CRG de dispositivo cuando se produce un suceso en uno de los nodos que define el dominio de recuperación; por ejemplo, un suceso de sustitución por anomalía o conmutación.

Un suceso de sustitución por anomalía se produce cuando el nodo primario del clúster falla y los CRG se conmutan con todos los recursos que gestionan, y un suceso de conmutación se produce cuando un CRG específico se conmuta manualmente desde el nodo primario al nodo de copia de seguridad.

En cualquiera de los dos casos, el programa de salida es el responsable de inicializar e iniciar todos los programas que estaban en ejecución en el nodo primario anterior, lo que convierte el primer nodo de copia de seguridad en el nuevo nodo primario.

Por ejemplo, con IBM MQ, el programa de salida debería ser el responsable de iniciar el subsistema de IBM MQ (QMQM) y los gestores de colas. Los gestores de colas se deberían configurar para iniciar automáticamente los escuchas y servicios, como por ejemplo los supervisores.

Un programa de salida de ejemplo, AMQSCRG4, está disponible en IBM i.

Configuración de IASP conmutable

Se puede configurar IBM MQ para aprovechar las prestaciones de agrupación en clúster de IBM i. Para ello:

1. Cree un clúster de IBM i entre los sistemas de centro de datos
2. Mueva el gestor de colas a una IASP.

   "Mover, o eliminar, un gestor de colas a, o desde, una agrupación de almacenamiento auxiliar independiente" en la página 431 contiene código de ejemplo para ayudarle a realizar esta operación.
3. Debe crear un CRG definiendo el dominio de recuperación, la IASP y el programa de salida.

   "Configuración de un grupo de recursos de clúster de dispositivo" en la página 430 contiene código de ejemplo para ayudarle a realizar esta operación.

Conceptos relacionados

"ASPs independientes y alta disponibilidad" en la página 451
Las ASP independientes permiten mover aplicaciones y datos entre servidores. La flexibilidad de las ASP independientes las ha convertido en la base de algunas soluciones de alta disponibilidad de IBM i. Cuando se plantee la posibilidad de utilizar una ASP o una ASP independiente para el diario de gestor de colas, debería tener en cuenta otra configuración de alta disponibilidad basada en ASP independientes.

IBM i *Configuración de un grupo de recursos de clúster de dispositivo*

Programa de ejemplo para configurar un grupo de recursos de clúster de dispositivo (CRG).

Acerca de esta tarea

En el ejemplo siguiente, tenga en cuenta que:

- [NOMBRE SITIO PRIMARIO] y [NOMBRE SITIO COPIA DE SEGURIDAD] podrían ser dos series distintas cualesquiera de ocho caracteres o menos.
- [IP PRIMARIA] e [IP DE COPIA DE SEGURIDAD] son las IP que se utilizarán para la duplicación.

Procedimiento

1. Identifique el nombre del clúster.
2. Identifique el nombre de programa de salida de CRG y la biblioteca.
3. Determine el nombre del nodo primario y de los nodos de copia de seguridad que definirá este CRG.
4. Identifique la IASP que gestionará este CRG, y asegúrese de que se ha creado en el nodo primario.
5. Cree una descripción de dispositivo en los nodos de copia de seguridad mediante este mandato:

```
 CRTDEVASP DEVD([IASP NAME]) RSRCNAME([IASP NAME])
```

6. Añada la dirección IP de toma de control a todos los nodos mediante el mandato:

```
ADDTCPIFC INTNETADR(' [TAKEOVER IP]') LIND([LINE DESC])
SUBNETMASK('[SUBNET MASK]') AUTOSTART(*NO)
```

7. Inicie la dirección IP de toma de control solo en el nodo primario mediante el mandato:

```
STRTCPIFC INTNETADR('[TAKEOVER IP')
```

8. Opcional: Si la IASP es conmutable, invoque este mandato:

```
CRTCRG CLUSTER([CLUSTER NAME]) CRG( [CRG NAME]) CRGTYPE(*DEV) EXITPGM([EXIT LIB]/[EXIT
NAME])
USRPRF([EXIT PROFILE]) RCYDMN(( [PRIMARY NODE] *PRIMARY) ([BACKUP NAME] *BACKUP))
EXITPGMFMT(EXTP0200) CFGOBJ(([IAPS NAME] *DEVD *ONLINE '[TAKEOVER IP]')
```

9. Opcional: Si se va a duplicar la IASP, invoque este mandato:

```
CRTCRG CLUSTER([CLUSTER NAME]) CRG([CRG NAME]) CRGTYPE(*DEV) EXITPGM([EXIT LIB]/[EXIT NAME])
USRPRF([EXIT PROFILE]) RCYDMN(([PRIMARY NODE] *PRIMARY *LAST [PRIMARY SITE NAME] ('[PRIMARY
IP]'))
[BACKUP NAME] *BACKUP *LAST [BACKUP SITE NAME] ('[BACKUP IP]'))) EXITPGMFMT(EXTP0200)
CFGOBJ(([IAPS NAME] *DEVD *ONLINE '[TAKEOVER IP]'))
```

IBM i *Mover, o eliminar, un gestor de colas a, o desde, una agrupación de almacenamiento auxiliar independiente*

Un programa de ejemplo para mover un gestor de colas a una agrupación de almacenamiento auxiliar independiente (IASP) y mandatos para eliminar un gestor de colas de una IASP.

Acerca de esta tarea

En el ejemplo siguiente, tenga en cuenta que:

- [NOMBRE GESTOR] es el nombre del gestor de colas.
- [NOMBRE IASP] es el nombre de la IASP.
- [BIBLIOTECA GESTOR] es el nombre de la biblioteca del gestor de colas.
- [DIRECTORIO GESTOR] es el nombre del directorio del gestor de colas.

Procedimiento

1. Identifique el nodo primario y los nodos de copia de seguridad.
2. Realice el procedimiento siguiente en el nodo primario:
   a) Asegúrese de que el gestor de colas ha finalizado.
   b) Asegúrese de que la IASP esté establecida en `vary on` mediante el mandato

```
VRYCFG CFGOBJ([IASP NAME]) CFGTYPE(*DEV) STATUS(*ON)
```

   c) Cree el directorio de gestores de colas bajo la IASP.
   Habrá un directorio bajo la raíz con el nombre de la IASP, que es:

```
QSH CMD('mkdir -p /[IASP_NAME]/QIBM/UserData/mqm/qmgrs/')
```

   d) Mueva los objetos de IFS del gestor al directorio de gestores de colas que acaba de crear bajo la IASP mediante el mandato siguiente:

```
QSH CMD('mv /QIBM/UserData/mqm/qmgrs/[MANAGER NAME]
/[IASP NAME]/QIBM/UserData/mqm/qmgrs')
```

e) Cree un archivo de salvar temporal denominado MGRLIB mediante el mandato:

```
CRTSAVF QGPL/MGRLIB
```

f) Guarde la biblioteca de gestores de colas en el archivo de salvar MGRLIB, mediante el mandato siguiente:

```
SAVLIB LIB([MANGER LIBRARY]) DEV(*SAVF) SAVF(QGPL/MGRLIB)
```

g) Suprima la biblioteca de gestores de colas mediante el mandato siguiente, e ignore todos los mensajes de consulta:

```
DLTLIB [MANAGER LIBRARY]
```

h) Restaure la biblioteca de gestores de colas en la IASP mediante el mandato siguiente:

```
RSTLIB SAVLIB([MANAGER LIBRARY]) DEV(*SAVF) SAVF(QGPL/MGRLIB)
RSTASPDEV([IASP NAME])
```

i) Suprima el archivo de salvar temporal mediante el mandato siguiente:

```
 DLTF FILE(QGPL/MGRLIB)
```

j) Cree un enlace simbólico a los objetos de IFS del gestor de colas bajo la IASP, mediante el mandato siguiente:

```
ADDLNK OBJ('/[IASP NAME]/QIBM/UserData/mqm/qmgrs/[MANAGER NAME]')
NEWLNK('/QIBM/UserData/mqm/qmgrs/[MANAGER NAME]')
```

k) Conecte a la IASP mediante el mandato siguiente:

```
SETASPGRP [IASP NAME]
```

l) Inicie el gestor de colas mediante el mandato:

```
STRMQM [MANAGER NAME]
```

3. Realice el procedimiento siguiente en el nodo o nodos de copia de seguridad:

a) Cree un directorio de gestores de colas temporal mediante el mandato siguiente:

```
QSH CMD('mkdir -p /[IASP NAME]/QIBM/UserData/mqm/qmgrs/[MANAGER NAME]')
```

b) Cree un enlace simbólico al directorio temporal de gestores de colas mediante el mandato:

```
ADDLNK OBJ('/[IASP NAME]/QIBM/UserData/mqm/qmgrs/[MANAGER NAME]')
NEWLNK('/QIBM/UserData/mqm/qmgrs/[MANAGER NAME]')
```

c) Suprima el directorio temporal mediante el mandato siguiente:

```
QSH CMD('rm -r /[IASP NAME]')
```

d) Añada lo siguiente al final del archivo /QIBM/UserData/mqm/mqs.ini:

```
QueueManager:
Name=[MANAGER NAME]
Prefix=/QIBM/UserData/mqm
Library=[MANAGER LIBRARY]
Directory=[MANAGER DIRECTORY]
```

4. Para eliminar un gestor de colas de una IASP, emita los mandatos siguientes:

a) VRYCFG CFGOBJ([NOMBRE IASP]) CFGTYPE(*DEV) STATUS(*ON)

b) SETASPGRP [NOMBRE IASP]

c) ENDMQM [NOMBRE GESTOR]

d) DLTMQM [NOMBRE GESTOR]

IBM i *Configuración de un diario duplicado para ASP en IBM i*

Configure un gestor de colas de instancias múltiples robusto utilizando una réplica síncrona entre diarios duplicados.

Una configuración de gestor de colas duplicadas utiliza diarios que se crean en agrupaciones de almacenamiento auxiliar (ASP) básicas o independientes.

En IBM i, los datos del gestor de colas se escriben en diarios y en un sistema de archivos. Los diarios contienen la copia maestra de los datos del gestor de colas. Los diarios se comparten entre sistemas mediante la replicación de diarios síncrona o asíncrona. Para reiniciar una instancia de gestor de colas se necesita una mezcla de diarios locales y remotos. El reinicio del gestor de colas lee registros de diario de la mezcla de diarios locales y remotos en el servidor y los datos del gestor de colas en el sistema de archivos de red compartidos. Los datos del sistema de archivos aceleran el reinicio del gestor de colas. Los puntos de comprobación se almacenan en el sistema de archivos, marcando puntos de sincronización entre el sistema de archivos y los diarios. Los registros de diario almacenados antes del punto de comprobación no son necesarios para reinicios típicos del gestor de colas. Sin embargo, los datos del sistema de archivos pueden no estar actualizados y los registros de diario después del punto de comprobación se utilizan para completar el reinicio del gestor de colas. Los datos de los diarios adjuntos a la instancia se mantienen actualizados para que el reinicio se pueda completar satisfactoriamente.

Pero incluso los registros de diario pueden no estar actualizados si el diario remoto del servidor en espera se estaba replicando asíncronamente y la anomalía se ha producido antes de la sincronización. En el caso en que decida reiniciar un gestor de colas mediante un diario remoto no sincronizado, la instancia del gestor de colas en espera puede volver a procesar mensajes suprimidos antes de que la instancia activa haya fallado o no procesar mensajes recibidos antes de que la instancia activa haya fallado.

Otra posibilidad aunque remota, es que el sistema de archivos contenga el registro de punto de comprobación más reciente y un diario remoto no sincronizado en espera no lo contenga. En este caso, el gestor de colas no se reinicia automáticamente. Puede esperar hasta que el diario remoto esté sincronizado o iniciar el gestor de colas en espera en frío desde el sistema de archivos. Incluso aunque en este caso el sistema de archivos contenga un punto de comprobación más reciente de los datos del gestor de colas que el diario remoto, no contendrá todos los mensajes procesados antes de la anomalía de la instancia activa. Es posible que algunos mensajes se vuelvan a procesar y que otros no se procesen después de un reinicio en frío que no está sincronizado con los diarios.

Con un gestor de colas multiinstancia, el sistema de archivos también se utiliza para controlar qué instancia de un gestor de colas está activa y cuál está en espera. La instancia activa adquiere un bloqueo para los datos del gestor de colas. La que está en espera, espera para adquirir el bloqueo y cuando lo hace, se convierte en la instancia activa. La instancia activa libera el bloqueo si finaliza normalmente. El sistema de archivos libera el bloqueo si detecta que la instancia activa ha fallado o no puede acceder al sistema de archivos. El sistema de archivos debe cumplir los requisitos para detectar la anomalía; consulte el apartado Requisitos para sistemas de archivos compartidos.

La arquitectura de gestores de colas multiinstancia en IBM i proporciona un reinicio automático después de la anomalía del servidor o el gestor de colas. También soporta la restauración de los datos del gestor de colas después de la anomalía del sistema de archivos en el que está almacenado el gestor de colas.

En la Figura 24 en la página 434, si ALPHA falla, puede reiniciar manualmente QM1 en BETA, mediante el diario duplicado. Al añadir la posibilidad del gestor de colas multiinstancia a QM1, la instancia en espera de QM1 se reanuda automáticamente en BETA si la instancia activa en ALPHA falla. QM1 también se puede reanudar automáticamente si es el servidor ALPHA el que falla, no solo la instancia activa de QM1. Una vez que BETA se convierte en el host de la instancia del gestor de colas activa, la instancia en espera se puede iniciar en ALPHA.

La Figura 24 en la página 434 muestra una configuración que duplica diarios entre dos instancias de un gestor de colas mediante NetServer para almacenar datos de gestor de colas. Es posible que desee expandir el patrón para incluir más diarios y, por consiguiente, más instancias. Siga las reglas de denominación de diarios que se explican en el tema “Diarios del gestor de colas en IBM i” en la página

412. Actualmente, el número de instancias en ejecución del gestor de colas está limitado a dos, uno está activo y el otro está en espera.

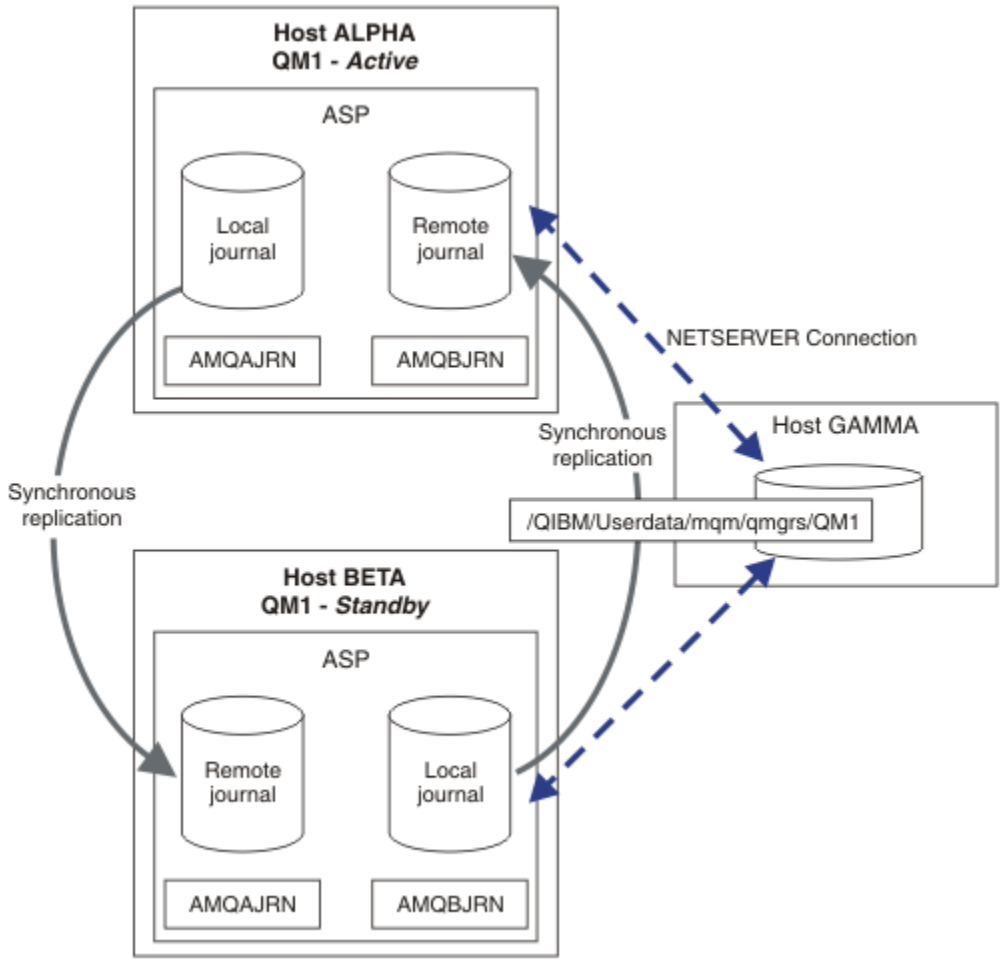


Figura 24. Duplicar un diario del gestor de colas

El diario local de QM1 en el host ALPHA se llama AMQAJRN (o con su nombre completo, QMQM1/AMQAJRN) y en BETA el diario es QMQM1/AMQBJRN. Cada diario local replica los diarios remotos en todas las otras instancias del gestor de colas. Si el gestor de colas se configura con dos instancias, un diario local se replica en un diario remoto.

Réplica de diarios remotos *SYNC o *ASYNC

Los diarios de IBM i se duplican mediante el registro por diario síncrono (*SYNC) o asíncrono (*ASYNC); consulte Gestión de diarios remotos.

El modo de réplica en Figura 24 en la página 434 es *SYNC, no *ASYNC. *ASYNC es más rápido, pero si se produce una anomalía cuando el estado del diario remoto es *ASYNCPEND, el diario local y remoto no son coherentes. El diario remoto debe ponerse al día con el diario local. Si elige *SYNC, a continuación, el sistema local espera al diario remoto antes de retroceder desde una llamada que necesite una escritura completa. Los diarios local y remoto permanecen coherentes el uno con el otro. Sólo si la operación *SYNC tarda más de un tiempo designado[1]y el registro por diario remoto se desactiva, las revistas se desincronizan. Se registra un error en la cola de mensajes de diario y en QSYSOPR. El gestor de colas detecta este mensaje, escribe un error en el registro cronológico de errores del gestor de colas y desactiva la réplica remota del diario del gestor de colas. La instancia del gestor de colas activo se

[1] El tiempo designado es de 60 segundos en IBM i 5 y en el rango de 1 a 3600 segundos en IBM i 6.1 en adelante.

reanuda sin que se realice un registro por diario remoto en este diario. Cuando el servidor remoto vuelve a estar disponible, debe reactivar manualmente la réplica síncrona del diario remoto. Los diarios se vuelven a sincronizar.

Un problema con la configuración *SYNC/*SYNC ilustrado en Figura 24 en la página 434 es cómo toma el control la instancia del gestor de colas en espera en BETA. Tan pronto como la instancia del gestor de colas en BETA escribe su primer mensaje persistente, intenta actualizar el diario remoto en ALPHA. Si la causa de que el control pase de ALPHA a BETA era la anomalía de ALPHA y ALPHA sigue sin funcionar, el registro por diario remoto en ALPHA falla. BETA espera a que ALPHA responda y después desactiva el registro por diario remoto y sigue procesando mensajes sólo con el registro por diario local. BETA debe esperar un poco para detectar que ALPHA no está funcionando, lo que provoca un período de inactividad.

La opción de establecer un registro por diario remoto en *SYNC o *ASYNC es cuestión de negociación. Tabla 24 en la página 435 resumen las negociaciones entre un par de gestores de colas sobre la utilización de un registro por diario *SYNC y *ASYNC:

Tabla 24. Opciones de registro por diario remoto

Activo	En espera	*SYNC	*ASYNC
***SYNC**		1. Cambio y migración tras error coherentes 2. La instancia en espera no se reanuda inmediatamente después de una migración tras error. 3. El registro por diario remoto debe estar disponible todo el tiempo 4. El rendimiento del gestor de colas depende del registro por diario remoto	1. Cambio y migración tras error coherentes 2. El registro por diario remoto se debe conmutar a *SYNC cuando el servidor en espera está disponible 3. El registro por diario remoto debe permanecer disponible después de reiniciarse 4. El rendimiento del gestor de colas depende del registro por diario remoto
***ASYNC**		1. No es una combinación sensible	1. Algunos mensajes se pueden perder o duplicar después de una sustitución por anomalía o una conmutación 2. La instancia en espera no puede estar disponible todo el tiempo para que instancia activa continúe sin retardo. 3. El rendimiento no depende del registro por diario remoto

***SYNC / *SYNC**

Las instancia del gestor de colas activo utiliza el registro por diario *SYNC y cuando se inicia la instancia del gestor de colas en espera, intenta inmediatamente utilizar el registro por diario *SYNC.

1. El diario remoto es transaccionalmente coherente con el diario local del gestor de colas activo. Si el gestor de colas se cambia a la instancia en espera, se puede reanudar inmediatamente. La instancia en espera normalmente se reanuda sin ninguna pérdida o duplicación de mensajes. Los mensajes sólo se pierden o se duplican si el registro por diario remoto ha fallado desde el último punto de comprobación y el gestor de colas activo anteriormente no se puede reiniciar.
2. Si el gestor de colas pasa a la instancia en espera tras un error, no se podrá iniciar inmediatamente. La instancia del gestor de colas en espera se activa con el registro por diario *SYNC. La causa de la sustitución por anomalía puede impedir el registro por diario remoto en el servidor que alberga la instancia en espera. El gestor de colas espera hasta que se detecta el

problema antes de procesar los mensajes persistentes. Se registra un error en la cola de mensajes de diario y en QSYSOPR. El gestor de colas detecta este mensaje, escribe un error en el registro cronológico de errores del gestor de colas y desactiva la réplica remota del diario del gestor de colas. La instancia del gestor de colas activo se reanuda sin que se realice un registro por diario remoto en este diario. Cuando el servidor remoto vuelve a estar disponible, debe reactivar manualmente la réplica síncrona del diario remoto. Los diarios se vuelven a sincronizar.

3. El servidor en el que se ha replicado el diario remoto siempre debe estar disponible para mantener el diario remoto. El diario remoto se replica normalmente en el mismo servidor que alberga el gestor de colas en espera. Es posible que el servidor no esté disponible. Se registra un error en la cola de mensajes de diario y en QSYSOPR. El gestor de colas detecta este mensaje, escribe un error en el registro cronológico de errores del gestor de colas y desactiva la réplica remota del diario del gestor de colas. La instancia del gestor de colas activo se reanuda sin que se realice un registro por diario remoto en este diario. Cuando el servidor remoto vuelve a estar disponible, debe reactivar manualmente la réplica síncrona del diario remoto. Los diarios se vuelven a sincronizar.
4. El registro por diario remoto es más lento que el registro por diario local y sustancialmente más lento si los servidores están separados por una larga distancia. El gestor de colas debe esperar al registro por diario remoto, lo que reduce el rendimiento del gestor de colas.

La configuración *SYNC/*SYNC entre un par de servidores tiene la desventaja de que se produce un retardo al reanudar la instancia en espera después de la sustitución por anomalía. La configuración *SYNC/*ASYNC no tiene este problema.

*SYNC/*SYNC garantiza que no se pierdan mensajes después de la conmutación o la sustitución por anomalía mientras haya un diario remoto disponible. Si desea reducir el riesgo de la pérdida de mensajes después del cambio o la migración tras error, tiene dos opciones. Detenga la instancia activa si el diario remoto se vuelve inactivo o cree diarios remotos en más de un servidor.

***SYNC / *ASYNC**

La instancia del gestor de colas activo utiliza el registro por diario *SYNC y cuando la instancia del gestor de colas en espera se inicia, utiliza el registro por diario *ASYNC. Poco después de que el servidor que alberga la instancia en espera nueva esté disponible, el operador del sistema debe conmutar el diario remoto en la instancia activa a *SYNC. Cuando el operador cambia el registro por diario remoto de *ASYNC a *SYNC, la instancia activa se detiene si el estado del diario remoto es *ASYNCPEND. La instancia del gestor de colas activo espera hasta que las entradas de diario restantes se transfieren al diario remoto. Cuando el diario remoto se ha sincronizado con el diario local, la espera nueva vuelve a ser transaccionalmente coherente con la instancia activa nueva. Desde la perspectiva de los gestores de colas multiinstancia, en una configuración *SYNC/*ASYNC, el operador del sistema de IBM i tiene una tarea adicional. El operador debe conmutar el registro de diario remoto a *SYNC además de reiniciar la instancia del gestor de colas que ha sufrido una anomalía.

1. El diario remoto es transaccionalmente coherente con el diario local del gestor de colas activo. Si la instancia del gestor de colas activa se ha conmutado o falla en la instancia en espera, la instancia en espera se puede reanudar inmediatamente. La instancia en espera normalmente se reanuda sin ninguna pérdida o duplicación de mensajes. Los mensajes sólo se pierden o se duplican si el registro por diario remoto ha fallado desde el último punto de comprobación y el gestor de colas activo anteriormente no se puede reiniciar.
2. El operador del sistema debe cambiar el diario remoto de *ASYNC a *SYNC poco después de que el sistema que alberga la instancia activa vuelva a estar disponible. El operador puede esperar a que el diario remoto se ponga al día antes de conmutar el diario remoto a *SYNC. El operador también puede conmutar inmediatamente la instancia remota a *SYNC y forzar que la instancia activa espere a que el diario de instancia en espera se haya puesto al día. Cuando el registro por diario remoto se establece en *SYNC, la instancia en espera es generalmente transaccionalmente coherente con la instancia activa. Los mensajes sólo se pierden o se duplican si el registro por diario remoto ha fallado desde el último punto de comprobación y el gestor de colas activo anteriormente no se puede reiniciar.
3. Cuando la configuración se ha restaurado de una conmutación o una sustitución por anomalía, el servidor en el que el diario remoto está alojado debe estar disponible todo el tiempo.

Elija *SYNC/*ASYNC cuando desee que el gestor de colas en espera se reanude rápidamente después de una sustitución por anomalía. Debe restaurar manualmente el valor del diario remoto a *SYNC en la instancia activa nueva. La configuración *SYNC/*ASYNC coincide con el patrón normal de administrar un par de gestores de colas multiinstancia. Cuando una instancia sufre una anomalía, hay un tiempo antes de reiniciar la instancia en espera durante el que no se puede realizar una migración tras error de la instancia activa.

***ASYNC / *ASYNC**
Los servidores que albergan los gestores de colas activo y en espera están configurados para utilizar el registro por diario remoto *ASYNC.

1. Cuando se producen la conmutación o la sustitución por anomalía, el gestor de colas sigue con el diario en el servidor nuevo. El diario no se puede sincronizar cuando tiene lugar la conmutación o sustitución por anomalía. En consecuencia, los mensajes se pueden perder o duplicar.
2. La instancia activa se ejecuta incluso aunque el servidor que alberga el gestor de colas en espera no esté disponible. El diario local se replica asíncronamente con el servidor en espera cuando está disponible.
3. El rendimiento del gestor de colas local no se ve afectado por el registro por diario remoto.

Elija *ASYNC/*ASYNC si el rendimiento es su principal requisito y si está preparado para perder o duplicar algunos mensajes después de la conmutación o la sustitución por anomalía.

***ASYNC / *SYNC**
No hay razón para utilizar esta combinación de opciones.

Activación del gestor de colas desde un diario remoto

Los diarios se pueden replicar síncrona o asíncronamente. Es posible que el diario remoto no esté activo, o es posible que se esté actualizando con el diario local. Puede que el diario remoto se esté actualizando incluso aunque se esté replicando de forma síncrona, porque se haya activado recientemente. Las reglas que el gestor de colas aplica al estado del diario remoto que utiliza durante el inicio, son las siguientes.

1. El inicio en espera falla si debe repetirse desde el diario remoto en el recurso en espera y el estado del diario es *FAILED o *INACTPEND.
2. Cuando empieza la activación del recurso en espera, el estado del diario remoto en espera debe ser *ACTIVE o *INACTIVE. Si el estado es *INACTIVE, es posible que la activación falle si no se han replicado todos los datos del diario.

   La anomalía se produce si los datos del gestor de colas del sistema de archivos de red tiene un registro de punto de comprobación más reciente que el presente en el diario remoto. La anomalía no se puede producirá mientras el diario remoto se active dentro del intervalo máximo de 30 minutos predeterminado entre puntos de comprobación. Si el gestor de colas en espera lee un registro de punto de comprobación más reciente del sistema de archivos, no se inicia.

   Tiene una opción: espere a que se pueda restaurar el diario local en el servidor activo o realice un inicio en frío del gestor de colas en espera. Si opta por el inicio en frío, el gestor de colas se inicia sin datos de diario y confía en la coherencia y la integridad de los datos del gestor de colas del sistema de archivos.

   Nota: Si realiza un inicio frío de un gestor de colas, corre el riesgo de perder o duplicar mensajes después del último punto de comprobación. Las transacciones de mensaje se escribieron en el diario pero es posibles que algunas de ellas no se hayan escrito en los datos del gestor de colas en el sistema de archivos. Cuando realiza un inicio frío de un gestor de colas, se inicia un diario nuevo y las transacciones que no se escriben en los datos del gestor de colas del sistema de archivos se pierden.
3. La activación del gestor de colas en espera a que el estado del diario remoto en espera cambie de *ASYNCPEND o *SYNCPEND a *ASYNC o *SYNC. Los mensajes se escriben en el registro de trabajo del controlador de ejecución de forma periódica.

   Nota: En este caso, la activación espera en el diario remoto local al gestor de colas en espera que se está activando. El gestor de colas espera también un tiempo antes de continuar sin un diario remoto.

Espera cuando intenta escribir de forma síncrona en su diario remoto (o diarios) y el diario no está disponible.

4. La activación se detendrá cuando el estado del diario cambie a `*FAILED` o `*INACTPEND`.

Los nombres y los estados de los diarios locales y remotos que deben utilizarse en la activación se escriben en el registro cronológico de errores del gestor de colas.

IBM i *Creación de un gestor de colas multiinstancia mediante la duplicación de diarios y NetServer en IBM i*

Cree un gestor de colas multiinstancia para ejecutarlo en dos servidores IBM i. Los datos del gestor de colas almacenan en un tercer servidor IBM i en el que se utiliza NetServer. El diario del gestor de colas se duplica entre los dos servidores mediante el registro por diario remoto. El mandato **ADDMQMJRN** se utiliza para simplificar la creación de los diarios remotos.

Antes de empezar

1. La tarea requiere tres servidores IBM i. Instale IBM MQ en dos de ellos, ALPHA y BETA, en el ejemplo. El producto debe ser como mínimo IBM WebSphere MQ 7.0.1 Fix Pack 1.
2. El tercer servidor es un servidor IBM i, conectado mediante NetServer a ALPHA y BETA. Se utiliza para compartir los datos del gestor de colas. No es necesario que tenga una instalación de IBM MQ. Es útil instalar IBM MQ en el servidor como un paso temporal para configurar los directorios del gestor de colas y los permisos.
3. Asegúrese de que el perfil de usuario de `QMQM` tenga la misma contraseña en los tres servidores.
4. Instale IBM i NetServer; consulte i5/OS NetServer.

Acerca de esta tarea

Siga estos pasos para crear la configuración descrita en la Figura 25 en la página 441. Los datos del gestor de colas se conectan mediante IBM i NetServer.

- Cree conexiones desde ALPHA y BETA a la unidad compartida de directorios de GAMMA que va a almacenar los datos del gestor de colas. La tarea también configura los permisos necesarios, los perfiles de usuario y las contraseñas.
- Añada entradas de base de datos relacional (RDBE) a los sistemas IBM i en los que se vayan a ejecutar las instancias del gestor de colas. Las entradas RDBE se utilizan para conectar con los sistemas IBM i que se utilizan para el registro por diario remoto.
- Cree el gestor de colas `QM1` en el servidor IBM i, ALPHA.
- Añada la información de control de gestor de colas para `QM1` en el otro servidor IBM i, BETA.
- Cree diarios remotos en los servidores IBM i para ambas instancias del gestor de colas. Cada gestor de colas escribe en el diario local. El diario local se replica en el diario remoto. El mandato **ADDMQMJRN** simplifica la adición de diarios y las conexiones.
- Inicie el gestor de colas, permitiendo una instancia en espera.

Procedimiento

1. Lleve a cabo la tarea “Creación de una unidad compartida de datos de un gestor de colas utilizando NetServer en IBM i” en la página 425.

   Como resultado, ALPHA y BETA tienen una participación, `/QNTC/GAMMA/WMQ`, que apunta a `/QIBM/UserData/mqm/qmgrs` en GAMMA. Los perfiles de usuario `QMQM` y `QMQMADM` tienen los permisos necesarios y `QMQM` tiene contraseñas coincidentes en los tres sistemas.
2. Añada entradas de base de datos relacional (RDBE) a los sistemas IBM i en los que se vayan a alojar las instancias del gestor de colas.

   a) En ALPHA cree la conexión con BETA.

```
ADDRDBDIRE RDB(BETA) RMTLOCNAME(BETA *IP) RMTAUTMTH(*USRIDPWD)
```

b) En BETA cree las conexiones con ALPHA.

```
ADDRDBDIRE RDB(ALPHA) RMTLOCNAME(ALPHA *IP) RMTAUTMTH(*USRIDPWD)
```

3. Cree el gestor de colas QM1 en ALPHA, guardando los datos del gestor de colas en GAMMA.

```
CRTMQM MQMNAME(QM1) UDLMSGQ(SYSTEM.DEAD.LETTER.QUEUE)
MQMDIRP(' /QNTC/GAMMA/WMQ ')
```

La vía de acceso, utiliza NetServer para crear los datos del gestor de colas.

4. Ejecutar en ALPHA. El mandato añade un diario remoto en BETA para.

```
ADDMQMJRN MQMNAME(QM1) RMTJRNRDB(BETA)
```

crea entradas de diario en su diario local en ALPHA cuando la instancia activa de está en ALPHA. El diario local en ALPHA se replica con el diario remoto en BETA.

5. Utilice el mandato,, para inspeccionar los datos de configuración de IBM MQ creados por en ALPHA.

La información se necesita en el paso siguiente.

En este ejemplo, se crea la siguiente configuración en ALPHA para:

```
Name=QM1
Prefix=/QIBM/UserData/mqm
Library=QMQM1
Directory=QM1
DataPath= /QNTC/GAMMA/WMQ /QM1
```

6. Cree una instancia de gestor de colas de QM1 en BETA utilizando el mandato. Ejecute el mandato siguiente en BETA para modificar la información de control del gestor de colas en BETA.

```
ADDMQMINF MQMNAME(QM1)
PREFIX('/QIBM/UserData/mqm')
MQMDIR(QM1)
MQMLIB(QMQM1)
DATAPATH(' /QNTC/GAMMA/WMQ /QM1 ')
```

Consejo: Copie y pegue la información de configuración. La stanza del gestor de colas es la misma en ALPHA y BETA.

7. Ejecutar en BETA. El mandato añade un diario local en BETA y un diario remoto en ALPHA para.

```
ADDMQMJRN MQMNAME(QM1) RMTJRNRDB(ALPHA)
```

crea entradas de diario en su diario local en BETA cuando la instancia activa de está en BETA. El diario local en BETA se replica con el diario remoto en ALPHA.

Nota: Como alternativa, puede configurar el registro por diario remoto de BETA a ALPHA mediante el registro por diario asíncrono.

Utilice este mandato para configurar el registro por diario asíncrono de BETA a ALPHA, en lugar del mandato en el paso “7” en la página 439.

```
ADDMQMJRN MQMNAME (QM1) RMTJRNRDB (ALPHA) RMTJRNDLV (*ASYNC)
```

Si el servidor o el registro por diario en ALPHA es el origen de la anomalía, BETA se inicia sin esperar a que se repliquen entradas del diario nuevas en ALPHA.

Cambie la modalidad de réplica a *SYNC, utilizando el mandato, cuando ALPHA vuelva a estar en línea.

Utilice la información en "Configuración de un diario duplicado para ASP en IBM i" en la página 433 para decidir si desea duplicar los diarios síncrona o asíncronamente o utilizar una mezcla de ambas modalidades. El valor predeterminado consiste en replicar síncronamente con un período de espera de 60 segundos para obtener una respuesta del diario remoto.

8. Verifique que los diarios en ALPHA y BETA estén habilitados y que el estado de la réplica de diario remoto sea.
   a) En ALPHA:

```
WRKMQMJRN MQMNAME(QM1)
```

   b) En BETA:

```
WRKMQMJRN MQMNAME(QM1)
```

9. Inicie las instancias del gestor de colas en ALPHA y BETA.
   a) Inicie la primera instancia en ALPHA, convirtiéndola en la instancia activa. Habilite la conmutación a una instancia en espera.

```
STRMQM MQMNAME(QM1) STANDBY(*YES)
```

   b) Inicie la segunda instancia en BETA, convirtiéndola en la instancia en espera.

```
STRMQM MQMNAME(QM1) STANDBY(*YES)
```

Resultados

Se utiliza para comprobar el estado del gestor de colas:

1. El estado de la instancia del gestor de colas en ALPHA debe ser.
2. El estado de la instancia del gestor de colas en BETA debe ser.

Ejemplo

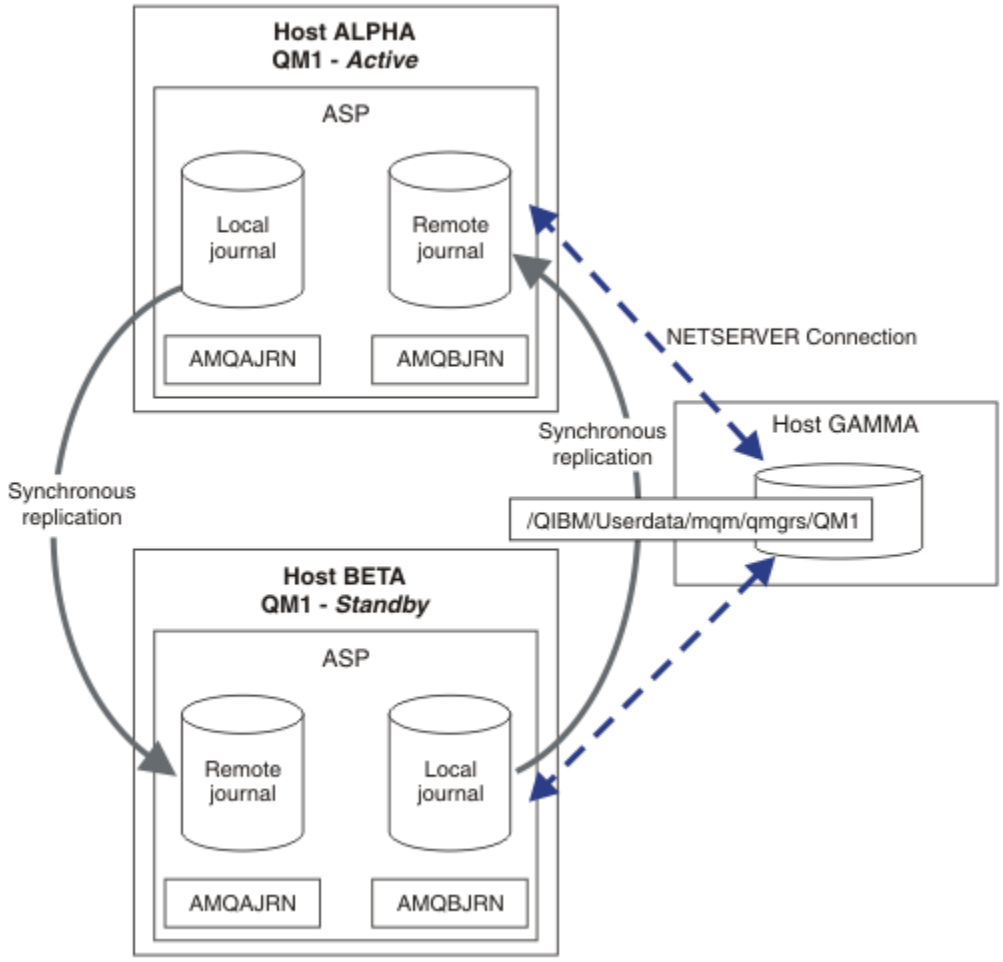


Figura 25. Configuración de diario duplicada

Qué hacer a continuación

- Verifique que las instancias activa y en espera se conmutan automáticamente. Puede ejecutar los programas de alta disponibilidad de muestra para probar la conmutación; consulte el tema Programas de ejemplo de alta disponibilidad. Los programas de ejemplo son clientes 'C'. Puede ejecutarlos desde una plataforma Windows o Unix.
  1. Inicie los programas de ejemplo de alta disponibilidad.
  2. En ALPHA, finalice el gestor de colas solicitando una conmutación:

     ```
     ENDMQM MQMNAME(QM1) OPTION(*IMMED) ALSWITCH(*YES)
     ```

  3. Compruebe que la instancia de en BETA esté activa.
  4. Reiniciar en ALPHA

     ```
     STRMQM MQMNAME(QM1) STANDBY(*YES)
     ```

- Busque configuraciones de alta disponibilidad alternativas:
  1. Utilice NetServer para colocar los datos del gestor de colas en un servidor Windows.

2. En lugar de utilizar el registro por diario remoto para duplicar el diario del gestor de colas, almacene el diario en una ASP independiente. Utilice la agrupación en clúster de IBM i para transferir la ASP independiente de ALPHA a BETA.

IBM i *Conversión de un gestor de colas de instancia única en un gestor de colas multiinstancia usando NetServer y duplicación de diarios en IBM i*

Convierta un gestor de colas de una sola instancia en un gestor de colas multiinstancia. Mueva los datos del gestor de colas a una unidad compartida de red conectada por NetServer. Duplique el diario del gestor de colas en un segundo servidor IBM i, utilizando el registro por diario remoto.

Antes de empezar

1. La tarea requiere tres servidores IBM i. La instalación existente de IBM MQ, en el servidor ALPHA en el ejemplo, debe ser como mínimo de IBM WebSphere MQ 7.0.1 Fix Pack 1. ALPHA está ejecutando un gestor de colas llamado `QM1` en el ejemplo.
2. Instale IBM MQ en el segundo servidor IBM i, BETA, en el ejemplo.
3. El tercer servidor es un servidor IBM i, conectado mediante NetServer a ALPHA y BETA. Se utiliza para compartir los datos del gestor de colas. No es necesario que tenga una instalación de IBM MQ. Es útil instalar IBM MQ en el servidor como un paso temporal para configurar los directorios del gestor de colas y los permisos.
4. Asegúrese de que el perfil de usuario de `QMQM` tenga la misma contraseña en los tres servidores.
5. Instale IBM i NetServer; consulte i5/OS NetServer.

Acerca de esta tarea

Realice los pasos siguientes para convertir un gestor de colas de una sola instancia en el gestor de colas multiinstancia que se muestra en la Figura 26 en la página 446. El gestor de colas de instancia única se borra en la tarea y después se vuelve a crear, almacenando los datos del gestor de colas en la unidad compartida de red conectada por NetServer. Este procedimiento es más fiable que mover los directorios del gestor de colas y los archivos a la unidad compartida de red mediante el mandato **CPY**.

- Cree conexiones desde ALPHA y BETA a la unidad compartida de directorios de GAMMA que va a almacenar los datos del gestor de colas. La tarea también configura los permisos necesarios, los perfiles de usuario y las contraseñas.
- Añada entradas de base de datos relacional (RDBE) a los sistemas IBM i en los que se vayan a ejecutar las instancias del gestor de colas. Las entradas RDBE se utilizan para conectar con los sistemas IBM i que se utilizan para el registro por diario remoto.
- Guarde los registros y las definiciones del gestor de colas, detenga el gestor de colas y suprímalo.
- Vuelva a crear el gestor de colas, almacenando los datos del gestor de colas en la unidad compartida de red en GAMMA.
- Añada la segunda instancia del gestor de colas al otro servidor.
- Cree diarios remotos en los servidores IBM i para ambas instancias del gestor de colas. Cada gestor de colas escribe en el diario local. El diario local se replica en el diario remoto. El mandato **ADDMQMJRN** simplifica la adición de diarios y las conexiones.
- Inicie el gestor de colas, permitiendo una instancia en espera.

Nota:

En el paso “4” en la página 443 de la tarea, suprime el gestor de colas de una sola instancia, `QM1`. Al suprimir el gestor de colas se suprimen todos los mensajes persistentes en las colas. Por esta razón, complete el proceso de todos los mensajes almacenados por el gestor de colas antes de convertirlo. Si no es posible procesar todos los mensajes, haga una copia de seguridad de la biblioteca del gestor de colas antes de “4” en la página 443. Restaure la biblioteca del gestor de colas después del paso “5” en la página 443.

Nota:

En el paso “5” en la página 443 de la tarea, vuelva a crear QM1. Aunque el gestor de colas tenga el mismo nombre, tiene un identificador de gestor de colas diferente. La agrupación en clúster del gestor de colas utiliza el identificador del gestor de colas. Para suprimir y volver a crear un gestor de colas en un clúster, debe eliminar primero el gestor de colas del clúster; consulte Eliminación de un gestor de colas de un clúster: Método alternativo o Eliminación de un gestor de colas de un clúster. Cuando haya vuelto a crear el gestor de colas, añádalo al clúster. Aunque tiene el mismo nombre que antes, los otros gestores de colas del clúster lo consideran un gestor de colas nuevo.

Procedimiento

1. Lleve a cabo la tarea “Creación de una unidad compartida de datos de un gestor de colas utilizando NetServer en IBM i” en la página 425.

   Como resultado, ALPHA y BETA tienen una participación, /QNTC/GAMMA/WMQ, que apunta a /QIBM/UserData/mqm/qmgrs en GAMMA. Los perfiles de usuario QMQM y QMQMADM tienen los permisos necesarios y QMQM tiene contraseñas coincidentes en los tres sistemas.
2. Añada entradas de base de datos relacional (RDBE) a los sistemas IBM i en los que se vayan a alojar las instancias del gestor de colas.
   a) En ALPHA cree la conexión con BETA.

   ```
   ADDRDBDIRE RDB(BETA) RMTLOCNAME(BETA *IP) RMTAUTMTH(*USRIDPWD)
   ```

   b) En BETA cree las conexiones con ALPHA.

   ```
   ADDRDBDIRE RDB(ALPHA) RMTLOCNAME(ALPHA *IP) RMTAUTMTH(*USRIDPWD)
   ```
3. Cree los scripts que vuelven a crear los objetos del gestor de colas.

   ```
   QSAVEQMGR LCLQMGRNAM(QM1) FILENAME('*CURLIB/QMQSC(QM1)')
   OUTPUT(*REPLACE) MAKEAUTH(*YES) AUTHFN('*CURLIB/QMAUT(QM1)')
   ```
4. Detenga el gestor de colas y suprímalo.

   ```
   ENDMQM MQMNAME(QM1) OPTION(*IMMED) ENDCCTJOB(*YES) RCDMQMIMG(*YES) TIMEOUT(15)
   DLTMQM MQMNAME(QM1)
   ```
5. Cree el gestor de colas QM1 en ALPHA, guardando los datos del gestor de colas en GAMMA.

   ```
   CRTMQM MQMNAME(QM1) UDLMSGQ(SYSTEM.DEAD.LETTER.QUEUE)
   MQMDIRP(' /QNTC/GAMMA/WMQ ')
   ```

   La vía de acceso, utiliza NetServer para crear los datos del gestor de colas.
6. Vuelva a crear los objetos del gestor de colas QM1 a partir de las definiciones guardadas.

   ```
   STRMQMMQSC SRCMBR(QM1) SRCFILE(*CURLIB/QMQSC) MQMNAME(QM1)
   ```
7. Aplique las autorizaciones de la información guardada.
   a) Compile el programa de autorización guardado.

   ```
   CRTCLPGM PGM(*CURLIB/QM1) SRCFILE(*CURLIB/QMAUT)
   SRCMBR(QM1) REPLACE(*YES)
   ```

   b) Ejecute el programa para aplicar las autorizaciones.

   ```
   CALL PGM(*CURLIB/QM1)
   ```

c) Renueve la información de seguridad para QM1.

```
RFRMQMAUT MQMNAME(QM1)
```

8. Ejecutar en ALPHA. El mandato añade un diario remoto en BETA para.

```
ADDMQMJRN MQMNAME(QM1) RMTJRNRDB(BETA)
```

crea entradas de diario en su diario local en ALPHA cuando la instancia activa de está en ALPHA. El diario local en ALPHA se replica con el diario remoto en BETA.

9. Utilice el mandato,, para inspeccionar los datos de configuración de IBM MQ creados por en ALPHA.

La información se necesita en el paso siguiente.

En este ejemplo, se crea la siguiente configuración en ALPHA para:

```
Name=QM1
Prefix=/QIBM/UserData/mqm
Library=QMQM1
Directory=QM1
DataPath= /QNTC/GAMMA/WMQ /QM1
```

10. Cree una instancia de gestor de colas de QM1 en BETA utilizando el mandato. Ejecute el mandato siguiente en BETA para modificar la información de control del gestor de colas en BETA.

```
ADDMQMINF MQMNAME(QM1)
PREFIX('/QIBM/UserData/mqm')
MQMDIR(QM1)
MQMLIB(QMQM1)
DATAPATH(' /QNTC/GAMMA/WMQ /QM1 ')
```

Consejo: Copie y pegue la información de configuración. La stanza del gestor de colas es la misma en ALPHA y BETA.

11. Ejecutar en BETA. El mandato añade un diario local en BETA y un diario remoto en ALPHA para.

```
ADDMQMJRN MQMNAME(QM1) RMTJRNRDB(ALPHA)
```

crea entradas de diario en su diario local en BETA cuando la instancia activa de está en BETA. El diario local en BETA se replica con el diario remoto en ALPHA.

Nota: Como alternativa, puede configurar el registro por diario remoto de BETA a ALPHA mediante el registro por diario asíncrono.

Utilice este mandato para configurar el registro por diario asíncrono de BETA a ALPHA, en lugar del mandato en el paso “7” en la página 439.

```
ADDMQMJRN MQMNAME (QM1) RMTJRNRDB (ALPHA) RMTJRNDLV (*ASYNC)
```

Si el servidor o el registro por diario en ALPHA es el origen de la anomalía, BETA se inicia sin esperar a que se repliquen entradas del diario nuevas en ALPHA.

Cambie la modalidad de réplica a *SYNC, utilizando el mandato, cuando ALPHA vuelva a estar en línea.

Utilice la información en “Configuración de un diario duplicado para ASP en IBM i” en la página 433 para decidir si desea duplicar los diarios síncrona o asíncronamente o utilizar una mezcla de ambas modalidades. El valor predeterminado consiste en replicar síncronamente con un período de espera de 60 segundos para obtener una respuesta del diario remoto.

12. Verifique que los diarios en ALPHA y BETA estén habilitados y que el estado de la réplica de diario remoto sea.

a) En ALPHA:

```
WRKMQMJRN MQMNAME(QM1)
```

b) En BETA:

```
WRKMQMJRN MQMNAME(QM1)
```

13. Inicie las instancias del gestor de colas en ALPHA y BETA.
    a) Inicie la primera instancia en ALPHA, convirtiéndola en la instancia activa. Habilite la conmutación a una instancia en espera.

```
STRMQM MQMNAME(QM1) STANDBY(*YES)
```

    b) Inicie la segunda instancia en BETA, convirtiéndola en la instancia en espera.

```
STRMQM MQMNAME(QM1) STANDBY(*YES)
```

Resultados

Se utiliza para comprobar el estado del gestor de colas:

1. El estado de la instancia del gestor de colas en ALPHA debe ser.
2. El estado de la instancia del gestor de colas en BETA debe ser.

Ejemplo

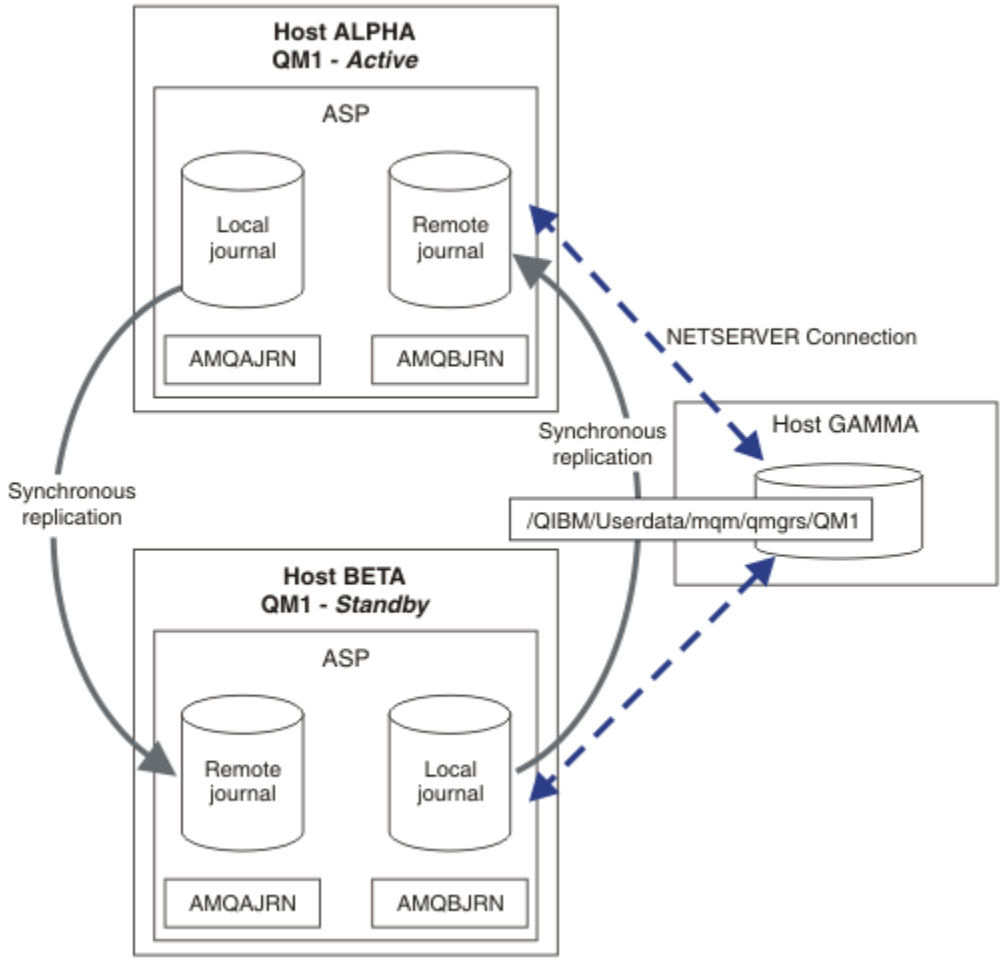


Figura 26. Configuración de diario duplicada

Qué hacer a continuación

- Verifique que las instancias activa y en espera se conmutan automáticamente. Puede ejecutar los programas de alta disponibilidad de muestra para probar la conmutación; consulte el tema Programas de ejemplo de alta disponibilidad. Los programas de ejemplo son clientes 'C'. Puede ejecutarlos desde una plataforma Windows o Unix.
  1. Inicie los programas de ejemplo de alta disponibilidad.
  2. En ALPHA, finalice el gestor de colas solicitando una conmutación:

     ```
     ENDMQM MQMNAME(QM1) OPTION(*IMMED) ALSWITCH(*YES)
     ```

  3. Compruebe que la instancia de en BETA esté activa.
  4. Reiniciar en ALPHA

     ```
     STRMQM MQMNAME(QM1) STANDBY(*YES)
     ```

- Busque configuraciones de alta disponibilidad alternativas:
  1. Utilice NetServer para colocar los datos del gestor de colas en un servidor Windows.

2. En lugar de utilizar el registro por diario remoto para duplicar el diario del gestor de colas, almacene el diario en una ASP independiente. Utilice la agrupación en clúster de IBM i para transferir la ASP independiente de ALPHA a BETA.

IBM i *Configuración de diario ASP independiente conmutada en IBM i*

No es necesario que duplique un diario ASP independiente para crear una configuración de gestor de colas multiinstancia. Debe automatizar un medio para transferir la ASP independiente desde el gestor de colas activo al gestor de colas en espera. Hay soluciones alternativas de alta disponibilidad que se pueden aplicar con una ASP independiente y no todas ellas requieren el uso de un gestor de colas multiinstancia.

Cuando utilice una ASP independiente, no es necesario que duplique el diario de gestor de colas. Si ha instalado la gestión de clústeres y los servidores que hospedan las instancias del gestor de colas se encuentran en el mismo grupo de recursos de clúster, a continuación, el diario del gestor de colas se podrá transferir automáticamente a otro servidor que quede dentro de una distancia corta del servidor activo, si el host que ejecuta la instancia activa falla. También puede transferir el diario manualmente, como parte de una conmutación planificada, o puede escribir un procedimiento de mandato para transferir la ASP independiente programáticamente.

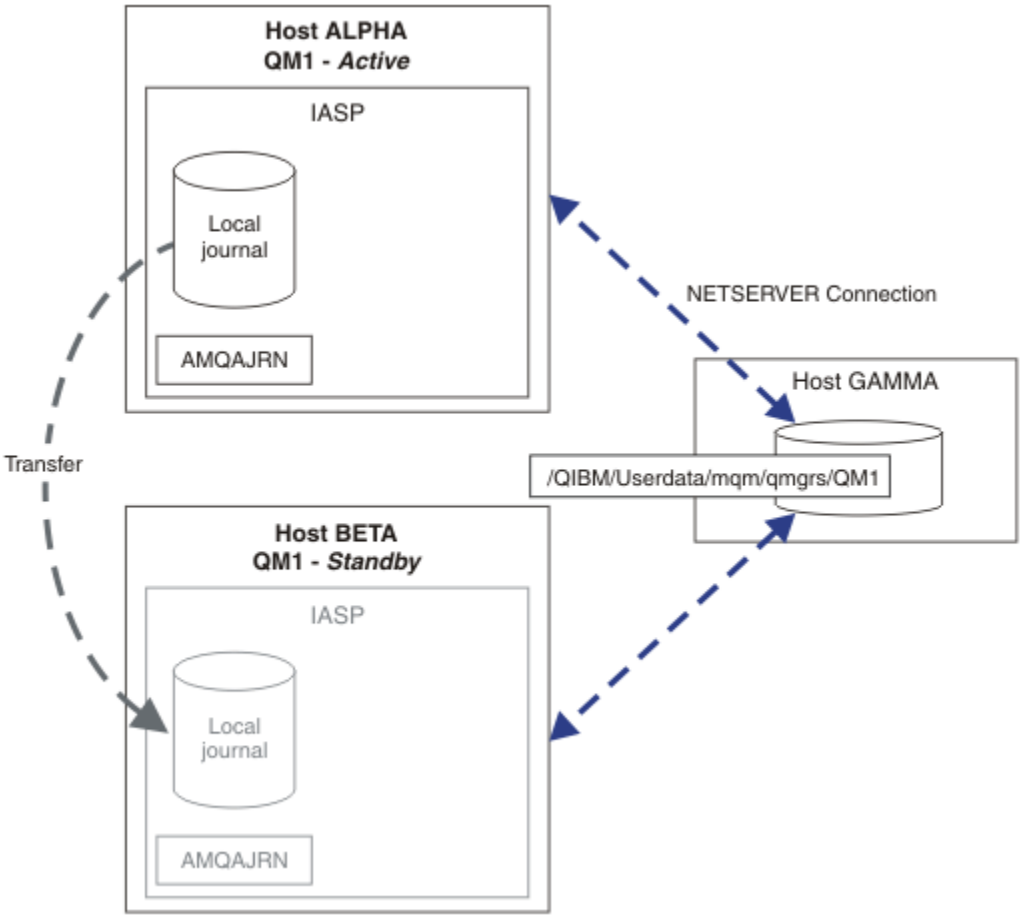


Figura 27. Transferir un diario de gestor de colas utilizando la ASP independiente

Para la operación del gestor de colas multiinstancia, los datos del gestor de colas deben almacenarse en un sistema de archivos compartido. El sistema de archivos puede estar alojado en una gran variedad de plataformas distintas. No puede almacenar datos del gestor de colas multiinstancia en una ASP ni en una ASP independiente.

El sistema de archivos compartidos desempeña dos roles en la configuración: los mismos datos del gestor de colas se comparten entre todas las instancias del gestor de colas. El sistema de archivos tiene

un protocolo de bloqueo robusto que garantiza que sólo una instancia del gestor de colas pueda acceder a los datos del gestor de colas una vez éste se haya iniciado. Si el gestor de colas falla o se interrumpen las comunicaciones con el servidor de archivos, el sistema de archivos debe liberar el bloqueo de los datos del gestor de colas mantenido por la instancia activa que ya no se puede comunicarse con el servidor de archivos. La instancia del gestor de colas en espera puede entonces obtener acceso de lectura/escritura a los datos del gestor de colas. El protocolo del sistema de archivos debe ajustarse a un conjunto de reglas para funcionar correctamente con gestores de colas multiinstancia; consulte “Componentes de una solución de alta disponibilidad en IBM i” en la página 424.

El mecanismo de bloqueo serializa el inicio del mandato del gestor de colas de inicio y controla qué instancia del gestor de colas está activa. Una vez el gestor de colas pasa a estar activo, vuelva a crear sus colas desde el diario local que el usuario o el clúster HA (de alta disponibilidad) ha transferido al servidor en espera. Los clientes que se pueden volver a conectar y que están esperando la reconexión con el mismo gestor de colas se reconectan y todas las transacciones en curso se restituyen. Las aplicaciones que están configuradas para iniciarse como servicios de gestor de colas se inician.

Debe asegurarse de que el diario local de la instancia del gestor de colas activa que haya fallado en la ASP independiente se transfiera al servidor que hospeda la nueva instancia del gestor de colas en espera activada, o bien configurando el gestor de recursos de clúster, o bien transfiriendo la ASP independiente manualmente. La utilización de ASP independientes no excluye la configuración de diarios remotos y de la duplicación, si decide utilizar ASP independientes para la copia de seguridad y la recuperación tras desastre, ni utilizar duplicación de diarios remotos para la configuración de gestores de cola multiinstancia.

Si ha decidido utilizar la ASP independiente, puede tener en cuenta que existen configuraciones alternativas de alta disponibilidad. La información relacionada de estas soluciones se describe en “ASPs independientes y alta disponibilidad” en la página 451.

1. En lugar de utilizar gestores de colas multiinstancia, instale y configure un gestor de colas de una sola instancia exclusivamente en una ASP independiente, y utilice los servicios de alta disponibilidad de IBM i para conmutar el gestor de colas. Probablemente necesitará aumentar la solución con un monitor de gestor de colas para detectar si el gestor de colas ha fallado independientemente del servidor. Esta es la base de la solución que se proporciona en el *Supportpac MC41: Configuring IBM MQ for iSeries for High Availability*.
2. Utilice ASP independientes y la duplicación entre sistemas (XSM) para duplicar la ASP independiente, en lugar de conmutar la ASP independiente en el bus local. Ello permite ampliar la cobertura geográfica de la solución de ASP independiente en lo que se refiere al tiempo necesario para escribir registros de anotaciones a través de una larga distancia.

IBM i *Creación de un gestor de colas multiinstancia usando una ASP independiente y NetServer en IBM i*

Cree un gestor de colas multiinstancia para ejecutarlo en dos servidores IBM i. Los datos del gestor de colas se almacenan en un servidor IBM i utilizando NetServer. El diario del gestor de colas se almacena en una ASP independiente. Utilice la agrupación en clúster de IBM i o un procedimiento manual para transferir la ASP independiente que contiene el diario del gestor de colas al otro servidor IBM i.

Antes de empezar

1. La tarea requiere tres servidores IBM i. Instale IBM MQ en dos de ellos, ALPHA y BETA, en el ejemplo. El producto debe ser como mínimo IBM WebSphere MQ 7.0.1 Fix Pack 1.
2. El tercer servidor es un servidor IBM i, conectado mediante NetServer a ALPHA y BETA. Se utiliza para compartir los datos del gestor de colas. No es necesario que tenga una instalación de IBM MQ. Es útil instalar IBM MQ en el servidor como un paso temporal para configurar los directorios del gestor de colas y los permisos.
3. Asegúrese de que el perfil de usuario de `QMQM` tenga la misma contraseña en los tres servidores.
4. Instale IBM i NetServer; consulte i5/OS NetServer.
5. Cree procedimientos para transferir la ASP independiente desde el gestor de colas fallido al recurso en espera que lo sustituye. En el *SupportPac MC41: Configuring IBM MQ for iSeries for High Availability*

puede encontrar algunas técnicas útiles para diseñar sus procedimientos de transferencia de ASP independiente.

Acerca de esta tarea

Siga estos pasos para crear la configuración descrita en la Figura 28 en la página 450. Los datos del gestor de colas se conectan mediante IBM i NetServer.

- Cree conexiones desde ALPHA y BETA a la unidad compartida de directorios de GAMMA que va a almacenar los datos del gestor de colas. La tarea también configura los permisos necesarios, los perfiles de usuario y las contraseñas.
- Cree el gestor de colas QM1 en el servidor IBM i, ALPHA.
- Añada la información de control de gestor de colas para QM1 en el otro servidor IBM i, BETA.
- Inicie el gestor de colas, permitiendo una instancia en espera.

Procedimiento

1. Lleve a cabo la tarea "Creación de una unidad compartida de datos de un gestor de colas utilizando NetServer en IBM i" en la página 425.

   Como resultado, ALPHA y BETA tienen una participación, /QNTC/GAMMA/WMQ, que apunta a /QIBM/UserData/mqm/qmgrs en GAMMA. Los perfiles de usuario QMQM y QMQMADM tienen los permisos necesarios y QMQM tiene contraseñas coincidentes en los tres sistemas.
2. Cree el gestor de colas QM1 en ALPHA, guardando los datos del gestor de colas en GAMMA.

   ```
   CRTMQM MQMNAME(QM1) UDLMSGQ(SYSTEM.DEAD.LETTER.QUEUE)
   MQMDIRP(' /QNTC/GAMMA/WMQ ')
   ```

   La vía de acceso, utiliza NetServer para crear los datos del gestor de colas.
3. Utilice el mandato,, para inspeccionar los datos de configuración de IBM MQ creados por en ALPHA.

   La información se necesita en el paso siguiente.

   En este ejemplo, se crea la siguiente configuración en ALPHA para:

   ```
   Name=QM1
   Prefix=/QIBM/UserData/mqm
   Library=QMQM1
   Directory=QM1
   DataPath= /QNTC/GAMMA/WMQ /QM1
   ```
4. Cree una instancia de gestor de colas de QM1 en BETA utilizando el mandato. Ejecute el mandato siguiente en BETA para modificar la información de control del gestor de colas en BETA.

   ```
   ADDMQMINF MQMNAME(QM1)
   PREFIX('/QIBM/UserData/mqm')
   MQMDIR(QM1)
   MQMLIB(QMQM1)
   DATAPATH(' /QNTC/GAMMA/WMQ /QM1 ')
   ```

   Consejo: Copie y pegue la información de configuración. La stanza del gestor de colas es la misma en ALPHA y BETA.
5. Inicie las instancias del gestor de colas en ALPHA y BETA.
   a) Inicie la primera instancia en ALPHA, convirtiéndola en la instancia activa. Habilite la conmutación a una instancia en espera.

      ```
      STRMQM MQMNAME(QM1) STANDBY(*YES)
      ```

   b) Inicie la segunda instancia en BETA, convirtiéndola en la instancia en espera.

```
STRMQM MQMNAME(QM1) STANDBY(*YES)
```

Resultados

Se utiliza para comprobar el estado del gestor de colas:

1. El estado de la instancia del gestor de colas en ALPHA debe ser.
2. El estado de la instancia del gestor de colas en BETA debe ser.

Ejemplo

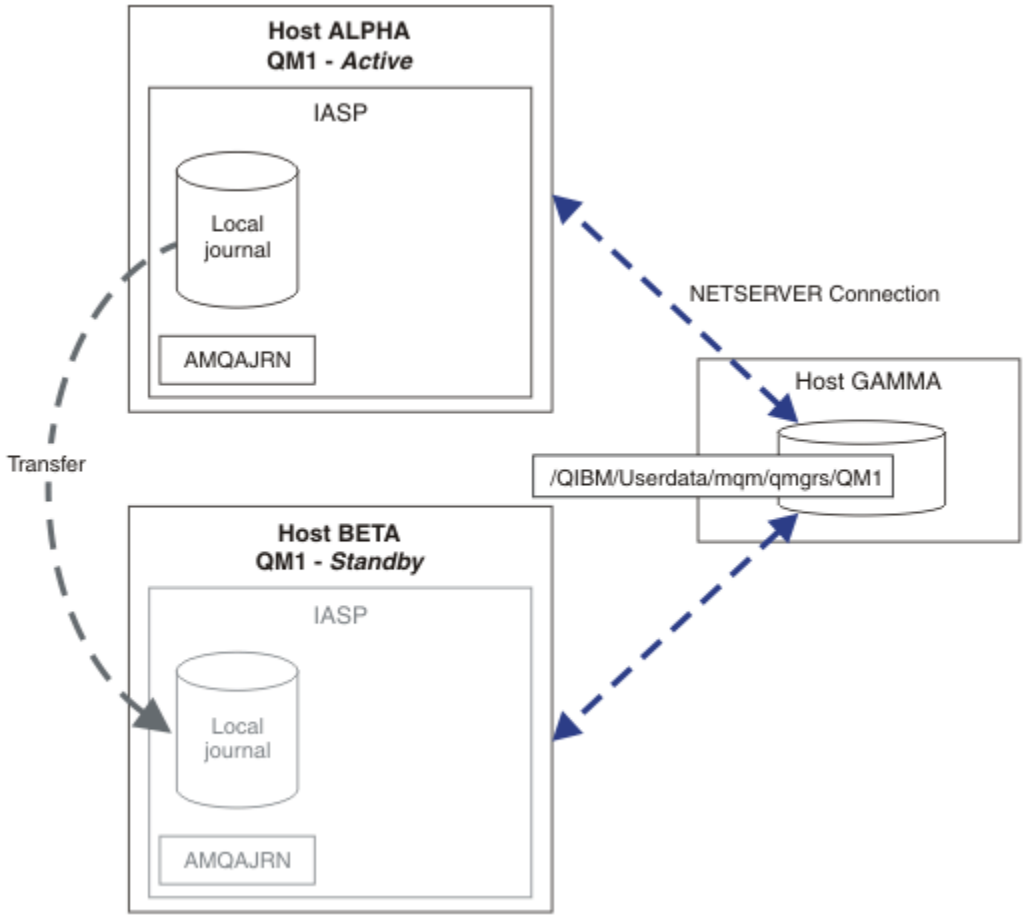


Figura 28. Transferir un diario de gestor de colas utilizando la ASP independiente

Qué hacer a continuación

- Verifique que las instancias activa y en espera se conmutan automáticamente. Puede ejecutar los programas de alta disponibilidad de muestra para probar la conmutación; consulte el tema Programas de ejemplo de alta disponibilidad. Los programas de ejemplo son clientes 'C'. Puede ejecutarlos desde una plataforma Windows o Unix.
  1. Inicie los programas de ejemplo de alta disponibilidad.
  2. En ALPHA, finalice el gestor de colas solicitando una conmutación:

     ```
     ENDMQM MQMNAME(QM1) OPTION(*IMMED) ALSWITCH(*YES)
     ```

  3. Compruebe que la instancia de en BETA esté activa.

4. Reiniciar en ALPHA

```
STRMQM MQMNAME(QM1) STANDBY(*YES)
```

- Busque configuraciones de alta disponibilidad alternativas:
  1. Utilice NetServer para colocar los datos del gestor de colas en un servidor IBM i.
  2. En lugar de utilizar una ASP independiente para transferir el diario del gestor de colas al servidor en espera, utilice el registro por diario remoto para duplicar el diario en el servidor en espera.

IBM i *ASPs independientes y alta disponibilidad*

Las ASP independientes permiten mover aplicaciones y datos entre servidores. La flexibilidad de las ASP independientes las ha convertido en la base de algunas soluciones de alta disponibilidad de IBM i. Cuando se plantee la posibilidad de utilizar una ASP o una ASP independiente para el diario de gestor de colas, debería tener en cuenta otra configuración de alta disponibilidad basada en ASP independientes.

Las agrupaciones de almacenamiento auxiliar (ASP) son un bloque de creación de la arquitectura de IBM i. Las unidades de disco se agrupan para formar una única ASP. Al colocar objetos en distintas ASP se puede evitar que los datos de una ASP se infecten debido a las anomalías en el disco de la otra ASP.

Cada servidor IBM i tiene al menos una ASP *básica*, conocida como la agrupación de almacenamiento auxiliar del sistema. Ésta se denomina ASP1 y, a veces se la conoce como *SYSBAS. Puede configurar hasta 31 ASP de *usuario* básicas adicionales que sean fáciles de distinguir en el sistema de la ASP desde el punto de vista de la aplicación, porque comparten el mismo espacio de nombres. Si utiliza varias ASP básicas para distribuir aplicaciones por muchos discos, puede mejorar el rendimiento y reducir el tiempo de recuperación. La utilización de varias ASP básicas también proporciona un cierto grado de aislamiento contra una posible anomalía del disco, pero no mejora la fiabilidad general.

Las ASPs independientes son un tipo de ASP especial. A menudo se conocen como agrupaciones de discos independientes. Las agrupaciones de discos independientes son un componente clave de la alta disponibilidad de IBM i. Puede almacenar datos y aplicaciones que se consideren a sí mismos como independientes del sistema actual con el que están conectados en unidades de almacenamiento de disco independientes. Puede configurar ASPs independientes conmutables o no conmutables. Desde la perspectiva de la disponibilidad, normalmente sólo tratará con ASP independientes conmutables, que se pueden transferir automáticamente de servidor a servidor. Como consecuencia, podrá mover las aplicaciones y los datos de la ASP independiente de servidor a servidor.

Al contrario que las ASP de usuario básicas, las ASPs independientes no comparten el mismo espacio de nombres que la ASP de sistema. Las aplicaciones que funcionan con ASP de usuario necesitan cambios para que puedan funcionar también con una ASP independiente. Debe verificar si su software y el software de terceros que está utilizando funciona en un entorno de ASP independiente.

Cuando la ASP independiente se adjunta a otro servidor distinto, el espacio de nombres de la ASP independiente tiene que combinarse con el espacio de nombres de la ASP de sistema. Este proceso se denomina *habilitación* de la ASP independiente. Puede habilitar una ASP independiente sin hacer una carga de programa inicial en el servidor. El soporte de agrupación en clúster es necesario para transferir ASPs independientes automáticamente de un servidor al otro.

Creación de soluciones fiables con ASPs independientes

El registro por diario en una ASP independiente, en lugar del registro por diario en una ASP y la utilización de la réplica de diarios constituye un medio alternativo para que el gestor de colas en espera consiga una copia del diario local a partir de la instancia del gestor de colas fallida. Para transferir automáticamente la ASP independiente a otro servidor, deberá haber instalado y configurado el soporte de agrupación en clúster. Existen varias soluciones de alta disponibilidad para ASPs independientes que se basan en el soporte de clústeres y en el duplicado de disco de nivel bajo, que pueden combinarse o sustituirse utilizando gestores de colas multiinstancia.

La siguiente lista describe los componentes que resultan necesarios para crear una solución fiable basada en ASPs independientes.

Registro por diario
Los gestores de colas y otras aplicaciones utilizan diarios para escribir datos persistentes de forma segura en el disco para protegerlos ante posibles pérdidas de datos en la memoria debidas a una anomalía en el servidor. Esto a veces se denomina la coherencia de momentos puntuales. No garantiza la coherencia de las múltiples actualizaciones que tienen lugar durante un período de tiempo.

Control de compromiso
El uso de las transacciones globales permite coordinar las actualizaciones de mensajes y bases de datos a fin de que los datos escritos en el diario sean coherentes. Otorga consistencia a lo largo del tiempo mediante el uso de un protocolo de compromiso de dos fases.

Disco conmutado

Los discos conmutados son gestionados por el grupo de recursos de clúster de dispositivo (CRG) en un clúster de alta disponibilidad. El grupo de recursos de clúster conmuta las ASPs independientes automáticamente en un nuevo servidor cuando se produce una parada no planificada. Los grupos de recursos de clúster están limitados geográficamente en la medida del bus de entrada/salida local.

La configuración de su diario local en una ASPs independiente conmutable permite transferir el diario a un servidor diferente y reanudar el procesamiento de mensajes. Todos los cambios efectuados en mensajes persistentes sin un control de punto de sincronismo o sin la confirmación del control de punto de sincronismo, causan la pérdida de los mismos a menos que la ASP independiente sufra una anomalía.

Si utiliza tanto el control registro por diario y de compromiso en ASPs independientes conmutables, puede transferir los diarios de base de datos y los diarios de gestor de colas a un servidor distinto y reanudar el procesamiento de transacciones sin ninguna pérdida de consistencia ni de transacciones confirmadas.

Duplicación entre sistemas (XSM)
XSM duplica la ASP independiente primaria en una ASP independiente secundaria geográficamente remota por toda la red TCP/IP y transfiere el control automáticamente en caso de que se produzca una anomalía. Tiene la posibilidad de configurar una duplicación síncrona o asíncrona. La duplicación síncrona reduce el rendimiento del gestor de colas porque los datos se duplican antes de que finalicen la operaciones de escritura en el sistema de producción, pero garantizar que la ASP independiente secundaria esté actualizada. Si utiliza la duplicación asíncrona no puede garantizar que la ASP independiente secundaria esté actualizada. La duplicación asíncrona no conserva la coherencia de la ASP independiente secundaria.

Existen tres tecnologías XSM.

Duplicación geográfica
La duplicación geográfica es una ampliación de la agrupación en clúster que permite conmutar las ASPs independientes en una amplia zona. Tanto en modalidad síncrona como asíncrona. Sólo se puede garantizar una alta disponibilidad en modalidad síncrona, pero la separación de las ASPs independientes podría afectar demasiado al rendimiento. Puede combinar con la duplicación geográfica con el disco conmutado para proporcionar una alta disponibilidad local y una recuperación tras desastre remota.

Duplicación síncrona de distancia corta
La duplicación síncrona de distancia corta es un servicio de nivel de dispositivo que proporciona una duplicación síncrona local rápida que cubre distancias más largas que el bus local. Puede combinarla con un gestor de colas multiinstancia para obtener una alta disponibilidad del gestor de colas y, al tener dos copias de la ASP independiente, obtener una alta disponibilidad del diario del gestor de colas.

Duplicación síncrona de distancia larga
La duplicación síncrona de distancia larga es un servicio de nivel de dispositivo que proporciona duplicación asíncrona y resulta adecuada para hacer copias de seguridad y conseguir recuperaciones tras desastre en distancias más largas, pero no resulta una opción habitual para la alta disponibilidad, porque sólo conserva la coherencia de un momento puntual en lugar de una vigencia prolongada.

Los puntos más importantes que debe tener en cuenta para tomar una decisión son,

¿ASP o ASP independiente?
No es necesario que ejecute un clúster de alta disponibilidad de IBM i para utilizar gestores de colas multiinstancia. Puede elegir ASP independientes, si ya está utilizando ASP independientes o si tiene requisitos de disponibilidad para otras aplicaciones que requieran ASP independientes. Podría valer la pena combinar las ASP independientes con gestores de colas multiinstancia para sustituir la supervisión del gestor de colas como una forma para detectar una posible anomalía del gestor de colas.

¿Disponibilidad?
¿Cuál es el objetivo de tiempo de recuperación (RTO)? Si necesita la apariencia de un comportamiento casi ininterrumpido... ¿qué solución tiene el tiempo de recuperación más rápido?

¿Disponibilidad de diario?
Cómo se elimina el diario como un único punto de error. Podrá adoptar una solución de hardware utilizando dispositivos RAID 1 o mejores, o podría combinar o utilizar una solución de software utilizando la réplica de diarios o el duplicado de discos.

¿Distancia?
Qué distancia separa las instancias del gestor de colas activa y en espera. ¿Sus usuarios pueden tolerar la degradación del rendimiento al hacer réplicas síncronas por distancias superiores a unos 250 metros?

¿Habilidades?
Es necesario trabajar para automatizar las tareas administrativas implicadas en el mantenimiento y el ejercicio de la solución de forma regular. Las habilidades necesarias para realizar la automatización son diferentes para las soluciones basadas en ASPs y en ASPs independientes.

IBM i *Borrado de un gestor de colas multiinstancia en IBM i*

Antes de suprimir un gestor de colas multiinstancia, detenga el registro por diario remoto y elimine las instancias del gestor de colas.

Antes de empezar

1. En este ejemplo, hay dos instancias del gestor de colas `QM1` definidas en los servidores ALPHA y BETA. ALPHA es la instancia activa y BETA está en espera. Los datos de gestor de colas asociados al gestor de colas `QM1` se almacenan en el servidor IBM i GAMMA, utilizando NetServer. Consulte “Creación de un gestor de colas multiinstancia mediante la duplicación de diarios y NetServer en IBM i” en la página 438.
2. ALPHA y BETA deben estar conectados para que IBM MQ pueda suprimir todos los diarios remotos definidos.
3. Verifique que sea posible acceder al directorio `/QNTC` y a la compartición de archivos de directorio de servidor mediante los mandatos del sistema **EDTF** o **WRKLNK**

Acerca de esta tarea

Antes de suprimir un gestor de colas multiinstancia de un servidor mediante el mandato **DLTMQM**, elimine todas las instancias del gestor de colas de otros servidores mediante el mandato **RMVMQMINF**.

Cuando elimine una instancia de gestor de colas mediante el mandato **RMVMQMINF**, se suprimirán los diarios locales y remotos cuyo prefijo sea `AMQ`, y estén asociados a la instancia. También se suprimirá la información de configuración sobre la instancia de gestor de colas, local respecto al servidor.

No ejecute el mandato **RMVMQMINF** en el servidor que contenga la instancia restante del gestor de colas. Al hacerlo, se impide que **DLTMQM** funcione correctamente.

Suprima el gestor de colas mediante el mandato **DLTMQM**. Los datos del gestor de colas se eliminan del compartimiento de red. Se suprimen los diarios locales y remotos cuyo prefijo sea `AMQ` y estén asociados a la instancia. **DLTMQM** también suprime la información de configuración de la instancia del gestor de colas, local respecto al servidor.

En el ejemplo sólo hay dos instancias del gestor de colas. IBM MQ da soporte a una configuración multiinstancia en ejecución que tenga una instancia de gestor de colas activa y una instancia en espera. Si ha creado instancias de gestor de colas adicionales para utilizarlas en configuraciones que están en ejecución, elimínelas mediante el mandato **RMVMQMINF**, antes de suprimir la instancia restante.

Procedimiento

1. Ejecute el mandato **CHGMQMJRN RMTJRNSTS** (*INACTIVE) en cada servidor para desactivar el registro por diario remoto entre las instancias del gestor de colas.

   a) En ALPHA:

   ```
   CHGMQMJRN MQMNAME('QM1')
   RMTJRNRDB('BETA') RMTJRNSTS(*INACTIVE)
   ```

   b) En BETA:

   ```
   CHGMQMJRN MQMNAME('QM1')
   RMTJRNRDB('ALPHA') RMTJRNSTS(*INACTIVE)
   ```

2. Ejecute el mandato **ENDMQM** en ALPHA, la instancia de gestor de colas activa, para detener ambas instancias de QM1.

   ```
   ENDMQM MQMNAME(QM1) OPTION(*IMMED) INSTANCE(*ALL) ENDCCTJOB(*YES)
   ```

3. Ejecute el mandato **RMVMQMINF** en ALPHA para eliminar los recursos del gestor de colas de la instancia de ALPHA y BETA.

   ```
   RMVMQMINF MQMNAME(QM1)
   ```

   RMVMQMINF elimina, de ALPHA, la información de configuración del gestor de colas para QM1. Si el nombre de diario tiene el prefijo AMQ, se suprime, de ALPHA, el diario local asociado a QM1. Si el nombre de diario tiene el prefijo AMQ y se ha creado un diario remoto, también se elimina, de BETA, el diario remoto.

4. Ejecute el mandato **DLTMQM** en BETA para suprimir QM1.

   ```
   DLTMQM MQMNAME(QM1)
   ```

   DLTMQM suprime los datos del gestor de colas de la unidad compartida de red en GAMMA. Elimina, de BETA, la información de configuración del gestor de colas para QM1. Si el nombre de diario tiene el prefijo AMQ, se suprime, de BETA, el diario local asociado a QM1. Si el nombre de diario tiene el prefijo AMQ y se ha creado un diario remoto, también se elimina, de ALPHA, el diario remoto.

Resultados

DLTMQM y **RMVMQMINF** suprimen los diarios locales y remotos que han creado **CRTMQM** y **ADDMQJRN**. Los mandatos también suprimen los destinatarios de diario. Los diarios y los receptores de diario deben seguir el convenio de denominación, que implica tener nombres que empiecen por AMQ. **DLTMQM** y **RMVMQMINF** eliminan los objetos de gestor de colas, los datos del gestor de colas, y la información de configuración del gestor de colas del archivo `mqs.ini`.

Qué hacer a continuación

Un enfoque alternativo es emitir los mandatos siguientes después de desactivar el registro por diario en el paso “1” en la página 454, y antes de finalizar las instancias del gestor de colas. O, si no ha seguido el convenio de denominación, debe suprimir los diarios y los receptores de diario por nombre.

1. En ALPHA:

```
RMVMQMJRN MQMNAME('QM1') RMTJRNRDB('BETA')
```

2. En BETA:

```
RMVMQMJRN MQMNAME('QM1') RMTJRNRDB('ALPHA')
```

Después de suprimir los diarios, prosiga con el resto de los pasos.

IBM i *Copia de seguridad de un gestor de colas multiinstancia en IBM i*

En el procedimiento se muestra cómo realizar una copia de seguridad de los objetos de gestor de colas en el servidor local y de los datos del gestor de colas en el archivo del servidor de red. Adapte el ejemplo para realizar una copia de seguridad de datos para otros gestores de colas.

Antes de empezar

En este ejemplo, los datos de gestor de colas asociados al gestor de colas QM1 se almacenan en el servidor IBM i llamado GAMMA, utilizando NetServer. Consulte “Creación de un gestor de colas multiinstancia mediante la duplicación de diarios y NetServer en IBM i” en la página 438 IBM MQ está instalado en los servidores ALPHA y BETA. El gestor de colas, QM1, está configurado en ALPHA y BETA.

Acerca de esta tarea

IBM i no permite guardar datos desde un directorio remoto. Guarde los datos del gestor de colas en un sistema de archivos remoto utilizando los procedimientos de copia de seguridad local en el servidor del sistema de archivos. En esta tarea, el sistema de archivos de red está en un servidor IBM i, GAMMA. Se realiza una copia de seguridad de los datos del gestor de colas en un archivo de salvar en GAMMA.

Si el sistema de archivos de red estuviera en Windows o Linux, podría almacenar los datos del gestor de colas en un archivo comprimido y luego guardarlos. Si tiene un sistema de copia de seguridad como, por ejemplo, Tivoli Storage Manager, utilícelo para realizar la copia de seguridad de los datos del gestor de colas.

Procedimiento

1. Cree un archivo de salvar en ALPHA para la biblioteca del gestor de colas asociada a QM1.

   Utilice el nombre de la biblioteca del gestor de colas para denominar el archivo de salvar.

```
CRTSAVF FILE(QGPL/QMQM1)
```

2. Guarde la biblioteca del gestor de colas en el archivo de salvar en ALPHA.

```
SAVLIB LIB(QMQM1) DEV(*SAVF) SAVF(QGPL/QMQM1)
```

3. Cree un archivo de salvar para el directorio de datos del gestor de colas, en GAMMA.

   Utilice el nombre del gestor de colas para denominar el archivo de salvar.

```
CRTSAVF FILE(QGPL/QMDQM1)
```

4. Guarde la copia de los datos del gestor de colas del directorio local en GAMMA.

```
SAV DEV('/QSYS.LIB/QGPL.LIB/QMDQM1.FILE') OBJ('/QIBM/Userdata/mqm/qmgrs/QM1')
```

IBM i *Mandatos para configurar gestores de colas multiinstancia*

IBM MQ tiene mandatos para simplificar las tareas de configurar la réplica de diarios, añadir nuevas instancias de gestor de colas y configurar gestores de colas para utilizar la ASP independiente.

Los mandatos del diario para crear y gestionar diarios locales y remotos son,

ADDMQMJRN
Con este mandato puede crear diarios locales y remotos con nombres para una instancia del gestor de colas y configurar si la réplica es síncrona o asíncrona, cual es el tiempo de espera de la síncrona y si el diario remoto debe activarse inmediatamente.

CHGMQMJRN
El mandato modifica el tiempo de espera, el estado y los parámetros de entrega que afectan a los diarios duplicados.

RMVMQMJRN
Elimina los diarios *remotos* con nombre de una instancia del gestor de colas.

WRKMQMJRN
Lista el estado de los diarios locales y remotos para una instancia de gestor de colas local.

Añada y gestione instancias del gestor de colas adicionales utilizando los mandatos siguientes, que modifican el archivo `mqs.ini`.

ADDMQMINF
El mandato utiliza la información que ha extraído del archivo `mqs.ini` con el mandato DSPMQMINF para añadir una nueva instancia del gestor de colas en un servidor IBM i distinto.

RMVMQMINF
Elimina una instancia del gestor de colas. Utilice este mandato, o bien para eliminar una instancia de un gestor de colas existente, o bien para eliminar la información de configuración de un gestor de colas que se ha suprimido de un servidor distinto.

El mandato **CRTMQM** tiene tres parámetros que le ayudarán a configurar un gestor de colas multiinstancia.

MQMDIRP(*DFT| *prefijo-directorio*)
Utilice este parámetro para seleccionar un punto de montaje que se correlacione con los datos del gestor de colas en almacenamiento por red.

ASP(*SYSTEM|*ASPDEV|*número-agrupación-almacenamiento-auxiliar*)
Especifique *SYSTEM o un *número-agrupación-almacenamiento-auxiliar* para colocar el diario del gestor de colas en el sistema o en una ASP de usuario básica. Seleccione la opción *ASPDEV y establezca también un nombre de dispositivo utilizando el parámetro **ASPDEV** para colocar el diario del gestor de colas en una ASP independiente.

ASPDEV(*ASP|*nombre-dispositivo*)
Especifique un *nombre-dispositivo* de un dispositivo de ASP independiente primario o secundario. Si selecciona *ASP obtendrá el mismo resultado que si especifica **ASP** (*SYSTEM).

IBM i Consideraciones sobre la migración tras error de disco y el rendimiento en IBM i

Utilice distintas agrupaciones de almacenamiento auxiliares para mejorar el rendimiento y la fiabilidad.

Si utiliza un número elevado de mensajes persistentes o mensajes de gran tamaño en sus aplicaciones, el tiempo empleado en guardar dichos mensajes en el disco pasará a ser un factor fundamental en el rendimiento del sistema.

Asegúrese de que tiene suficiente activación de disco para hacer frente a esta posibilidad; también podría tener una "Auxiliary Storage Pool" (agrupación de almacenamiento auxiliar - ASP) aparte en la que guardar los receptores de diario del gestor de colas.

Puede especificar en qué ASP se almacena la biblioteca de gestores de cola y los diarios cuando crea el gestor de colas mediante el parámetro ASP de **CRTMQM**. Por omisión, los diarios y la biblioteca de gestores de colas así como los datos IFS se almacenan en el sistema ASP.

Las ASP permiten aislar los objetos en una o varias unidades de disco específicas. Con ello, también se reduce la pérdida de datos debido a una anomalía en el soporte de disco. En la mayoría de los casos, sólo se pierden los datos almacenados en unidades de disco de la ASP afectada.

Se le recomienda almacenar la biblioteca de gestores de colas y los datos de diario en ASP de usuario separadas a fin de que la raíz del sistema de archivos IFS proporcione sustitución por anomalía y reduzca la contención de disco.

Para obtener más información, consulte Copia de seguridad y recuperación en la documentación de IBM i.

IBM i Utilización de SAVLIB para guardar bibliotecas de IBM MQ en IBM i

No puede utilizar `SAVLIB LIB(*ALLUSR)` para guardar las bibliotecas de IBM MQ porque estas bibliotecas tiene nombres que empiezan por Q.

Puede utilizar `SAVLIB LIB(QM*)` para guardar todas las bibliotecas del gestor de colas, pero sólo si está utilizando un dispositivo de guardar distinto de *SAVF. Para `DEV(*SAVF)`, debe utilizar un mandato SAVLIB para todas y cada una de las bibliotecas de gestor de colas del sistema.

IBM i Desactivación temporal de IBM MQ for IBM i

En esta sección se explica cómo desactivar temporalmente (finalizar ordenadamente) IBM MQ for IBM i.

Para desactivar temporalmente IBM MQ for IBM i:

1. Inicie una nueva sesión interactiva de IBM MQ for IBM i, asegurándose de no acceder a ningún objeto.
2. Asegúrese de que tiene:
   - Autorización *ALLOBJ o autorización de gestión de objetos para la biblioteca QMQM
   - Autorización suficiente para utilizar el mandato ENDSBS
3. Advierta a todos los usuarios de que va a detener IBM MQ for IBM i.
4. Los pasos que debe seguir a continuación dependen de si desea concluir (desactivar temporalmente) un solo gestor de colas (mientras que otros pueden seguir existiendo) (consulte “Conclusión de un gestor de colas individual para IBM MQ for IBM i” en la página 458) o todos los gestores de colas (consulte “Conclusión de todos los gestores de colas para IBM MQ for IBM i” en la página 459).
5. Concluya el servidor mqweb especificando el mandato siguiente en qshell:

```
/QIBM/ProdData/mqm/bin/endmqweb
```

Parámetro de ENDMQM ENDCCTJOB(*YES)

El parámetro ENDCCTJOB(*YES) de ENDMQM funciona de forma diferente en IBM MQ for IBM i V6.0 y posteriores en comparación con las versiones anteriores.

En las versiones anteriores, cuando se especifica ENDCCTJOB(*YES), MQ finaliza las aplicaciones por la fuerza.

En IBM MQ for IBM i V6.0 o posteriores, cuando se especifica ENDCCTJOB(*YES), las aplicaciones no finalizan sino que se desconectan del gestor de colas.

Si especifica ENDCCTJOB(*YES) y tiene aplicaciones que no están escritas para detectar que un gestor de colas está finalizando, la próxima vez que se emita una nueva llamada MQI, la llamada devolverá un error MQRC_CONNECTION_BROKEN (2009).

Como alternativa al uso de ENDCCTJOB(*YES), utilice el parámetro ENDCCTJOB(*NO) y la opción 22 de WRKMQM (Trabajar con trabajos) para finalizar manualmente los trabajos de aplicaciones que impedirán que un gestor de colas se reinicie.

IBM i Conclusión de un gestor de colas individual para IBM MQ for IBM i

Utilice esta información para conocer los tres tipos de conclusión.

En los procedimientos que se describen a continuación, se utiliza el nombre de gestor de colas de ejemplo QMgr1 y el nombre de subsistema de ejemplo SUBX. Sustituya estos nombres por sus propios valores si es necesario.

Conclusión planificada

Conclusión planificada de un gestor de colas en IBM i

1. Antes de cerrar, ejecute:

```
RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) MQMNAME(QMgr1) DSPJRNDTA(*YES)
```

2. Para concluir el gestor de colas, ejecute:

```
ENDMQM MQMNAME(QMgr1) OPTION(*CNTRLD)
```

   Si QMgr1 no finaliza, el canal o las aplicaciones probablemente están ocupados.
3. Si debe concluir QMgr1 inmediatamente, ejecute lo siguiente:

```
ENDMQM MQMNAME(QMgr1) OPTION(*IMMED)
ENDCCTJOB(*YES) TIMEOUT(15)
```

Conclusión no planificada

1. Para concluir el gestor de colas, ejecute:

```
ENDMQM MQMNAME(QMgr1) OPTION(*IMMED)
```

   Si QMgr1 no finaliza, el canal o las aplicaciones probablemente están ocupados.
2. Si necesita concluir QMgr1 inmediatamente, ejecute lo siguiente:

```
ENDMQM MQMNAME(QMgr1) OPTION(*IMMED)
ENDCCTJOB(*YES) TIMEOUT(15)
```

Cierre en condiciones anómalas

1. Para concluir el gestor de colas, ejecute:

```
ENDMQM MQMNAME(QMgr1) OPTION(*IMMED)
```

   Si QMgr1 no finaliza, continúe con el paso 3 siempre que:

   - QMgr1 se encuentre en su propio subsistema, o bien
   - Pueda finalizar todos los gestores de colas que comparten el mismo subsistema que QMgr1. Utilice el procedimiento de conclusión no planificada para todos los gestores de colas de este tipo.
2. Cuando haya llevado a cabo todos los pasos del procedimiento para todos los gestores de colas que comparten el subsistema (SUBX en nuestros ejemplos), ejecute:

```
ENDSBS SUBX *IMMED
```

   Si este mandato no se lleva a cabo correctamente, concluya todos los gestores de colas, utilizando el procedimiento de conclusión no planificada, y realice una IPL en la máquina.

Aviso: No utilice ENDJOBABN para trabajos de IBM MQ que no finalicen como resultado de ENDJOB o ENDSBS, a menos que esté preparado para realizar una IPL en la máquina inmediatamente después.

3. Inicie el subsistema, ejecutando:

```
STRSBS SUBX
```

4. Concluya el gestor de colas inmediatamente, ejecutando:

```
ENDMQM MQMNAME(QMgr1) OPTION(*IMMED)
ENDCCTJOB(*YES) TIMEOUT(10)
```

5. Reinicie el gestor de colas, ejecutando:

```
STRMQM MQMNAME(QMgr1)
```

Si esto falla, y:

- Ha reiniciado la máquina realizando una IPL, o bien
- Tiene un solo gestor de colas

Reorganice la memoria compartida de IBM MQ ejecutando:

```
ENDMQM MQMNAME(*ALL) OPTION(*IMMED)
ENDCCTJOB(*YES) TIMEOUT(15)
```

antes de repetir el paso 5.

Si el gestor de colas tarda más de unos cuantos segundos en reiniciarse, IBM MQ añade mensajes de estado de forma intermitente en el registro de trabajo, detallando el progreso del inicio.

Si sigue teniendo problemas para reiniciar el gestor de colas, póngase en contacto con el servicio de soporte de IBM. Cualquier otra acción que pudiera realizar podría dañar el gestor de colas, dejando a IBM MQ en un estado imposible de recuperar.

IBM i Conclusión de todos los gestores de colas para IBM MQ for IBM i

Utilice esta información para conocer los tres tipos de conclusión.

Los procedimientos son prácticamente los mismos que para un solo gestor de colas, pero utilizando *ALL en lugar del nombre del gestor de colas cuando sea posible, o utilizando un mandato repetidamente con cada nombre de gestor de colas correspondiente. En todos los procedimientos, se utiliza el nombre de gestor de colas de ejemplo QMgr1 y el nombre de subsistema de ejemplo SUBX. Puede sustituir estos nombres por otros que desee.

Conclusión planificada

1. Una hora antes de la conclusión, ejecute:

```
RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) MQMNAME(QMgr1) DSPJRNDTA(*YES)
```

Repita este paso para cada gestor de colas que desee concluir.

2. Para concluir el gestor de colas, ejecute:

```
ENDMQM MQMNAME(QMgr1) OPTION(*CNTRLD)
```

Repita este paso para cada gestor de colas que desee concluir; los mandatos separados pueden ejecutarse en paralelo.

Si algún gestor de colas no finaliza al cabo de un período de tiempo razonable (por ejemplo, 10 minutos), continúe en el paso 3.

3. Para concluir todos los gestores de colas inmediatamente, ejecute lo siguiente:

```
ENDMQM MQMNAME(*ALL) OPTION(*IMMED)
ENDCCTJOB(*YES) TIMEOUT(15)
```

Conclusión no planificada

1. Para concluir un gestor de colas, ejecute:

```
ENDMQM MQMNAME(QMgr1) OPTION(*IMMED)
```

   Repita este paso para cada gestor de colas que desee concluir; los mandatos separados pueden ejecutarse en paralelo.

   Si los gestores de colas no finalizan, el canal o las aplicaciones probablemente están ocupados.

2. Si necesita concluir los gestores de colas inmediatamente, ejecute lo siguiente:

```
ENDMQM MQMNAME(*ALL) OPTION(*IMMED)
ENDCCTJOB(*YES) TIMEOUT(15)
```

Cierre en condiciones anómalas

1. Para concluir los gestores de colas, ejecute:

```
ENDMQM MQMNAME(QMgr1) OPTION(*IMMED)
```

   Repita este paso para cada gestor de colas que desee concluir; los mandatos separados pueden ejecutarse en paralelo.

2. Finalice los subsistemas (SUBX en nuestros ejemplos), ejecutando:

```
ENDSBS SUBX *IMMED
```

   Repita este paso para cada subsistema que desee concluir; los mandatos separados pueden ejecutarse en paralelo.

   Si este mandato no se ejecuta correctamente, realice una IPL en el sistema.

   Aviso: No utilice ENDJOBABN para trabajos que no finalicen como resultado de ENDJOB o ENDSBS, a menos que esté preparado para realizar una IPL en el sistema inmediatamente después.

3. Inicie los subsistemas, ejecutando:

```
STRSBS SUBX
```

   Repita este paso para cada subsistema que desee iniciar.

4. Concluya los gestores de colas inmediatamente, ejecutando:

```
ENDMQM MQMNAME(*ALL) OPTION(*IMMED)
ENDCCTJOB(*YES) TIMEOUT(15)
```

5. Reinicie los gestores de colas, ejecutando:

```
STRMQM MQMNAME(QMgr1)
```

Repita este paso para cada gestor de colas que desee iniciar.

Si algún gestor de colas tarda más de unos cuantos segundos en reiniciarse, IBM MQ mostrará mensajes de estado de forma intermitente que detallan el progreso del inicio.

Si sigue teniendo problemas para reiniciar algún gestor de colas, póngase en contacto con el servicio de soporte de IBM. Cualquier otra acción que pudiera realizar podría dañar los gestores de colas, dejando a MQSeries o IBM MQ en un estado imposible de recuperar.

z/OS Administering IBM MQ for z/OS

IBM MQ for z/OS can be controlled and managed by MQSC and PCF commands, by a set of utilities and programs provided with the product, and by authorized applications.

For details of how to administer IBM MQ for z/OS and the different administrative tasks you might have to undertake, see the following links.

You can also administer IBM MQ for z/OS using the IBM MQ Explorer running in a Linux shell. For more information, see “Administración utilizando IBM MQ Explorer” on page 122.

Related concepts

IBM MQ for z/OS concepts

Related tasks

“Administración de IBM MQ” on page 7

Para administrar los gestores de colas de IBM MQ y los recursos asociados, elija el método que prefiera de un conjunto de tareas que puede utilizar para activar y gestionar estos recursos.

Planning your IBM MQ environment on z/OS

Configuring queue managers on z/OS

z/OS Issuing queue manager commands on z/OS

You can control most of the operational environment of IBM MQ by using control commands. You can issue MQSC and PCF commands from the IBM MQ for z/OS console, the initialization input data sets, the batch utility CSQUTIL, or authorized applications.

About this task

You use MQSC commands, in batch or interactive mode, to administer queue managers directly. You use PCF commands to help you create applications that administer queue managers. MQSC commands are in human-readable text form, whereas PCF commands let applications create requests and read the replies without having to parse text strings. Like MQSC commands, applications issue PCF commands by sending them as messages to the command input queue.

The following topics describe how you issue queue manager commands from the IBM MQ for z/OS console, the initialization input data sets, the batch utility CSQUTIL, or from authorized applications.

Not all commands can be issued from all sources. See “Sources from which you can issue MQSC and PCF commands on IBM MQ for z/OS” on page 461.

Related tasks

Preparing sample applications for the TSO environment on z/OS

Related information

Administering IBM MQ using MQSC commands

z/OS Sources from which you can issue MQSC and PCF commands on IBM MQ for z/OS

You can issue MQSC and PCF commands from the IBM MQ for z/OS console, the initialization input data sets, the batch utility CSQUTIL, or from authorized applications. Not all commands can be issued from all these sources.

Which MQSC and PCF commands can control each IBM MQ object

Table 1 in "Command summary for IBM MQ for z/OS" maps which MQSC and PCF commands can be used on IBM MQ for z/OS to alter, define, delete and display each IBM MQ object. See also MQSC commands reference and “Utilización de los formatos de mandato programable de IBM MQ” on page 27.

List of sources from which commands can be issued

If you are a suitably authorized user, you can issue IBM MQ commands from the following sources:

- The z/OS console or equivalent (such as SDSF/TSO).

  See also “Using the operations and control panels on z/OS” on page 475.

  Note: When using the z/OS console, you need to add `/cpf` to the start of a command, where `cpf` is the command prefix for the queue manager subsystem.
- The initialization input data sets CSQINP1, CSQINP2, CSQINPT and CSQINPX.

  See “Initialization commands for IBM MQ for z/OS” on page 472.
- The z/OS master get command routine, MGCRE (SVC 34).
- The IBM MQ batch utility programs such as CSQUTIL, which processes a list of commands in a sequential data set.

  See “Using the IBM MQ for z/OS utilities” on page 483.
- Suitably authorized applications, sending commands as messages to the SYSTEM.COMMAND.INPUT queue.

  The application can be any of the following:

  – A batch region program
  – A CICS application
  – An IMS application
  – A TSO application
  – An application program or utility on another IBM MQ system

  See “Writing programs to administer IBM MQ for z/OS” on page 492 and Preparing sample applications for the TSO environment on z/OS.

Not all commands can be issued from all sources

Commands are classified according to where they can be issued from:

1
CSQINP1

2
CSQINP2

C
The z/OS console

R
The command server and command queue, by means of CSQUTIL, CSQINPT, CSQINPX, or authorized applications.

Within the command descriptions in MQSC commands reference, these sources are identified by the use of the characters 1, 2, C, and R in each command description. Table 2 in "Command summary for IBM MQ for z/OS" summarizes the MQSC commands and the sources from which they can be issued.

Related tasks
Preparing sample applications for the TSO environment on z/OS

Related information

Administering IBM MQ using MQSC commands

z/OS Command summary for IBM MQ for z/OS

A summary of the main MQSC and PCF commands, and of the sources from which you can run MQSC commands on IBM MQ for z/OS.

Table 25 on page 463 maps which MQSC and PCF commands can be used on IBM MQ for z/OS to alter, define, delete and display each IBM MQ object.

Table 25. Summary of the main MQSC and PCF commands by object type

MQSC command	**ALTER**	**DEFINE**	**DISPLAY**	**DELETE**
PCF command	**Change**	**Create/Copy**	**Inquire**	**Delete**
AUTHINFO	X	X	X	X
CFSTATUS			X	
CFSTRUCT	X	X	X	X
CHANNEL	X	X	X	X
CHSTATUS			X	
NAMELIST	X	X	X	X
PROCESS	X	X	X	X
QALIAS	M	M	M	M
QCLUSTER			M	
QLOCAL	M	M	M	M
QMGR	X		X	
QMODEL	M	M	M	M
QREMOTE	M	M	M	M
QUEUE	P	P	X	P
QSTATUS			X	
STGCLASS	X	X	X	X

Key to table symbols:

- M = MQSC only
- P = PCF only
- X = both

There are many other MQSC and PCF commands which allow you to manage other IBM MQ resources, and carry out other actions in addition to those summarized in Table 25 on page 463.

Table 26 on page 464 shows every MQSC command, and where each command can be issued from.

- CSQINP1 initialization input data set
- CSQINP2 initialization input data set
- z/OS console (or equivalent)
- SYSTEM.COMMAND.INPUT queue and command server (from applications, CSQUTIL, or the CSQINPX initialization input data set)

Table 26. Sources from which to run MQSC commands				
Command	**CSQINP1**	**CSQINP2**	**z/OS console**	**Command input queue and server**
ALTER AUTHINFO		X	X	X
ALTER BUFFPOOL		X	X	X
ALTER CFSTRUCT		X	X	X
ALTER CHANNEL		X	X	X
ALTER NAMELIST		X	X	X
ALTER PROCESS		X	X	X
ALTER PSID			X	X
ALTER QALIAS		X	X	X
ALTER QLOCAL		X	X	X
ALTER QMGR		X	X	X
ALTER QMODEL		X	X	X
ALTER QREMOTE		X	X	X
ALTER SECURITY	X	X	X	X
ALTER SMDS		X	X	X
ALTER STGCLASS		X	X	X
ALTER SUB			X	X
ALTER TOPIC		X	X	X
ALTER TRACE	X	X	X	X
ARCHIVE LOG	X	X	X	X
BACKUP CFSTRUCT			X	X
CLEAR QLOCAL		X	X	X
CLEAR TOPICSTR			X	X
DEFINE AUTHINFO		X	X	X
DEFINE BUFFPOOL	X			
DEFINE CFSTRUCT		X	X	X
DEFINE CHANNEL		X	X	X
DEFINE LOG			X	X
DEFINE MAXSMSGS		X	X	X
DEFINE NAMELIST		X	X	X
DEFINE PROCESS		X	X	X
DEFINE PSID	X		X	X
DEFINE QALIAS		X	X	X
DEFINE QLOCAL		X	X	X

Table 26. Sources from which to run MQSC commands (continued)

Command	CSQINP1	CSQINP2	z/OS console	Command input queue and server
DEFINE QMODEL		X	X	X
DEFINE QREMOTE		X	X	X
DEFINE STGCLASS		X	X	X
DEFINE SUB			X	X
DEFINE TOPIC		X	X	X
DELETE AUTHINFO		X	X	X
DELETE BUFFPOOL		X	X	X
DELETE CFSTRUCT		X	X	X
DELETE CHANNEL			X	X
DELETE NAMELIST		X	X	X
DELETE PROCESS		X	X	X
DELETE PSID			X	X
DELETE QALIAS		X	X	X
DELETE QLOCAL		X	X	X
DELETE QMODEL		X	X	X
DELETE QREMOTE		X	X	X
DELETE STGCLASS		X	X	X
DELETE SUB			X	X
DELETE TOPIC		X	X	X
DISPLAY ARCHIVE	X	X	X	X
DISPLAY AUTHINFO		X	X	X
DISPLAY CFSTATUS			X	X
DISPLAY CFSTRUCT		X	X	X
DISPLAY CHANNEL		X	X	X
DISPLAY CHINIT			X	X
DISPLAY CHLAUTH		X	X	X
DISPLAY CHSTATUS			X	X
DISPLAY CLUSQMGR			X	X
DISPLAY CMDSERV	X	X	X	X
DISPLAY CONN		X	X	X
DISPLAY GROUP		X	X	X
DISPLAY LOG	X	X	X	X
DISPLAY MAXSMSGS		X	X	X

Table 26. Sources from which to run MQSC commands (continued)

Command	CSQINP1	CSQINP2	z/OS console	Command input queue and server
DISPLAY NAMELIST		X	X	X
DISPLAY PROCESS		X	X	X
DISPLAY PUBSUB		X	X	X
DISPLAY QALIAS		X	X	X
DISPLAY QCLUSTER		X	X	X
DISPLAY QLOCAL		X	X	X
DISPLAY QMGR		X	X	X
DISPLAY QMODEL		X	X	X
DISPLAY QREMOTE		X	X	X
DISPLAY QSTATUS		X	X	X
DISPLAY QUEUE		X	X	X
DISPLAY SBSTATUS			X	X
DISPLAY SECURITY			X	X
DISPLAY SMDS		X	X	X
DISPLAY SMDSCONN		X	X	X
DISPLAY STGCLASS		X	X	X
DISPLAY SUB			X	X
DISPLAY SYSTEM	X	X	X	X
DISPLAY TCLUSTER		X	X	X
DISPLAY THREAD		X	X	X
DISPLAY TOPIC		X	X	X
DISPLAY TPSTATUS		X	X	X
DISPLAY TRACE	X	X	X	X
DISPLAY USAGE		X	X	X
MOVE QLOCAL		X	X	X
PING CHANNEL			X	X
RECOVER BSDS			X	X
RECOVER CFSTRUCT			X	X
REFRESH CLUSTER			X	X
REFRESH QMGR		X	X	X
REFRESH SECURITY			X	X
RESET CFSTRUCT			X	X
RESET CHANNEL			X	X

Table 26. Sources from which to run MQSC commands (continued)

Command	CSQINP1	CSQINP2	z/OS console	Command input queue and server
RESET CLUSTER			X	X
RESET QMGR		X	X	X
RESET QSTATS		X	X	X
RESET SMDS			X	X
RESET TPIPE			X	X
RESOLVE CHANNEL			X	X
RESOLVE INDOUBT		X	X	X
RESUME QMGR			X	X
RVERIFY SECURITY		X	X	X
SET ARCHIVE	X	X	X	X
SET CHLAUTH		X	X	X
SET LOG	X	X	X	X
SET SYSTEM	X	X	X	X
START CHANNEL			X	X
START CHINIT		X	X	X
START CMDSERV	X	X	X	
START LISTENER			X	X
START QMGR			X	
START SMDSCONN		X	X	X
START TRACE	X	X	X	X
STOP CHANNEL			X	X
STOP CHINIT			X	X
STOP CMDSERV	X	X	X	
STOP LISTENER			X	X
STOP QMGR			X	X
STOP SMDSCONN		X	X	X
STOP TRACE	X	X	X	X
SUSPEND QMGR			X	X

In MQSC commands, each command description identifies the sources from which that command can be run.

z/OS Using MQSC to start and stop a queue manager on z/OS

An introduction to using control commands on IBM MQ for z/OS: After you have installed IBM MQ, use MQSC commands to start and stop a queue manager.

Before you begin

After you have installed IBM MQ, it is defined as a formal z/OS subsystem. This message appears during any initial program load (IPL) of z/OS:

```
CSQ3110I +CSQ1 CSQ3UR00 - SUBSYSTEM ssnm INITIALIZATION COMPLETE
```

where *ssnm* is the IBM MQ subsystem name.

From now on, you can start the queue manager for that subsystem *from any z/OS console that has been authorized to issue system control commands* ; that is, a z/OS SYS command group. You must issue the START command from the authorized console, you cannot issue it through JES or TSO.

If you are using queue sharing groups, you must start RRS first, and then Db2®, before you start the queue manager.

About this task

When a queue manager stops under normal conditions, its last action is to take a termination checkpoint. This checkpoint, and the logs, give the queue manager the information it needs to restart.

The following steps contain information about the START and STOP commands, and contain a brief overview of start-up after an abnormal termination has occurred.

Procedure

1. Start a queue manager

   You start a queue manager by issuing a START QMGR command. However, you cannot successfully use the START command unless you have appropriate authority. See the Setting up security on z/OS for information about IBM MQ security. The following code shows examples of the START command. Note that you must prefix an MQSC command with a command prefix string (CPF).

   ```
   +CSQ1  START QMGR
   +CSQ1  START QMGR PARM(NEWLOG)
   ```

   See START QMGR for information about the syntax of the START QMGR command.

   You cannot run the queue manager as a batch job or start it using a z/OS command START. These methods are likely to start an address space for IBM MQ that then ends abnormally. Nor can you start a queue manager from the CSQUTIL utility program or a similar user application.

   You can, however, start a queue manager from an APF-authorized program by passing a START QMGR command to the z/OS MGCRE (SVC 34) service.

   If you are using queue sharing groups, the associated Db2 systems and RRS must be active when you start the queue manager.

   Start options

   When you start a queue manager, a system parameter module is loaded. You can specify the name of the system parameter module in one of two ways:

   - With the PARM parameter of the /cpf START QMGR command, for example

     ```
     /cpf START QMGR PARM(CSQ1ZPRM)
     ```

   - With a parameter in the startup procedure, for example, code the JCL EXEC statement as

     ```
     //MQM  EXEC PGM=CSQYASCP,PARM='ZPARM(CSQ1ZPRM)'
     ```

A system parameter module provides information specified when the queue manager was customized.

You can use the **QMGRPROD** option to specify the product against which the queue manager usage is to be recorded, and the **AMSPROD** option to specify the equivalent for AMS if that is used. See the MQSC START QMGR command for details of the permitted values.

An example JCL EXEC statement follows:

```
//MQM EXEC  PGM=CSQYASCP,PARM='QMGRPROD(MQ)'
```

See z/OS MVS Product Management for more information on measured usage and product registration.

You can also use the ENVPARM option to substitute one or more parameters in the JCL procedure for the queue manager.

For example, you can update your queue manager startup procedure, so that the **DDname** CSQINP2 is a variable. This means that you can change the CSQINP2 **DDname** without changing the startup procedure. This is useful for implementing changes, providing back-outs for operators, and queue manager operations.

Suppose your start-up procedure for queue manager CSQ1 looked like this:

```
//CSQ1MSTR PROC INP2=NORM
//MQMESA  EXEC PGM=CSQYASCP
//STEPLIB DD  DISP=SHR,DSN=thlqual.SCSQANLE
//     DD  DISP=SHR,DSN=thlqual.SCSQAUTH
//     DD  DISP=SHR,DSN=db2qual.SDSNLOAD
//BSDS1  DD  DISP=SHR,DSN=myqual.BSDS01
//BSDS2  DD  DISP=SHR,DSN=myqual.BSDS02
//CSQP0000 DD  DISP=SHR,DSN=myqual.PSID00
//CSQP0001 DD  DISP=SHR,DSN=myqual.PSID01
//CSQP0002 DD  DISP=SHR,DSN=myqual.PSID02
//CSQP0003 DD  DISP=SHR,DSN=myqual.PSID03
//CSQINP1 DD  DISP=SHR,DSN=myqual.CSQINP(CSQ1INP1)
//CSQINP2 DD  DISP=SHR,DSN=myqual.CSQINP(CSQ1&INP2.)
//CSQOUT1 DD  SYSOUT=*
//CSQOUT2 DD  SYSOUT=*
```

If you then start your queue manager with the following command:

```
+CSQ1  START QMGR
```

then the CSQINP2 used is a member called CSQ1NORM.

However, suppose you are putting a new suite of programs into production so that the next time you start queue manager CSQ1, the CSQINP2 definitions are to be taken from member CSQ1NEW. To do this, you would start the queue manager with this command:

```
+CSQ1  START QMGR ENVPARM('INP2=NEW')
```

and CSQ1NEW would be used instead of CSQ1NORM. Note: z/OS limits the KEYWORD=value specifications for symbolic parameters (as in INP2=NEW) to 255 characters.

Starting after an abnormal termination

IBM MQ automatically detects whether restart follows a normal shutdown or an abnormal termination.

Starting a queue manager after it ends abnormally is different from starting it after the STOP QMGR command has been issued. After STOP QMGR, the system finishes its work in an orderly way and takes a termination checkpoint before stopping. When you restart the queue manager, it uses information from the system checkpoint and recovery log to determine the system status at shutdown.

However, if the queue manager ends abnormally, it terminates without being able to finish its work or take a termination checkpoint. When you restart a queue manager after an abend, it refreshes its knowledge of its status at termination using information in the log, and notifies you of the status of various tasks. Normally, the restart process resolves all inconsistent states. But, in some cases, you must take specific steps to resolve inconsistencies.

User messages on start-up

When you start a queue manager successfully, the queue manager produces a set of startup messages.

2. Stop a queue manager.

Before stopping a queue manager, all IBM MQ-related write-to-operator-with-reply (WTOR) messages must receive replies, for example, getting log requests. Each of the following commands terminates a running queue manager.

```
+CSQ1  STOP QMGR
+CSQ1  STOP QMGR MODE(QUIESCE)
+CSQ1  STOP QMGR MODE(FORCE)

+CSQ1  STOP QMGR MODE(RESTART)
```

The command STOP QMGR defaults to STOP QMGR MODE(QUIESCE).

In QUIESCE mode, IBM MQ does not allow any new connection threads to be created, but allows existing threads to continue; it terminates only when all threads have ended. Applications can request to be notified in the event of the queue manager quiescing. Therefore, use the QUIESCE mode where possible so that applications that have requested notification have the opportunity to disconnect. See What happens during termination for details.

If the queue manager does not terminate in a reasonable time in response to a STOP QMGR MODE(QUIESCE) command, use the DISPLAY CONN command to determine whether any connection threads exist, and take the necessary steps to terminate the associated applications. If there are no threads, issue a STOP QMGR MODE(FORCE) command.

The STOP QMGR MODE(QUIESCE) and STOP QMGR MODE(FORCE) commands deregister IBM MQ from the MVS Automatic Restart Manager (ARM), preventing ARM from restarting the queue manager automatically. The STOP QMGR MODE(RESTART) command works in the same way as the STOP QMGR MODE(FORCE) command, except that it does not deregister IBM MQ from ARM. This means that the queue manager is eligible for immediate automatic restart.

If the IBM MQ subsystem is not registered with ARM, the STOP QMGR MODE(RESTART) command is rejected and the following message is sent to the z/OS console:

```
CSQY205I ARM element arm-element is not registered
```

If this message is not issued, the queue manager is restarted automatically. For more information about ARM, see “Using the z/OS Automatic Restart Manager (ARM)” on page 550.

Only cancel the queue manager address space if STOP QMGR MODE(FORCE) does not terminate the queue manager.

If a queue manager is stopped by either canceling the address space or by using the command STOP QMGR MODE(FORCE), consistency is maintained with connected CICS or IMS systems. Resynchronization of resources is started when a queue manager restarts and is completed when the connection to the CICS or IMS system is established.

Note: When you stop your queue manager, you might find message IEF352I is issued. z/OS issues this message if it detects that failing to mark the address space as unusable would lead to an integrity exposure. You can ignore this message.

Stop messages

After issuing a STOP QMGR command, you get the messages CSQY009I and CSQY002I, for example:

```
CSQY009I +CSQ1 ' STOP QMGR' COMMAND ACCEPTED FROM
USER(userid), STOP MODE(FORCE)
CSQY002I +CSQ1 QUEUE MANAGER STOPPING
```

Where `userid` is the user ID that issued the STOP QMGR command, and the MODE parameter depends on that specified in the command.

When the STOP command has completed successfully, the following messages are displayed on the z/OS console:

```
CSQ9022I +CSQ1 CSQYASCP ' STOP QMGR' NORMAL COMPLETION
CSQ3104I +CSQ1 CSQ3EC0X - TERMINATION COMPLETE
```

If you are using ARM, and you did not specify MODE(RESTART), the following message is also displayed:

```
CSQY204I +CSQ1 ARM DEREGISTER for element arm-element type
arm-element-type successful
```

You cannot restart the queue manager until the following message has been displayed:

```
CSQ3100I +CSQ1 CSQ3EC0X - SUBSYSTEM ssnm READY FOR START COMMAND
```

z/OS Issuing commands from a z/OS console or equivalent

You can issue IBM MQ MQSC and PCF commands from a z/OS console or its equivalent. You can also issue IBM MQ commands from anywhere where you can issue z/OS commands, such as SDSF or by a program using the MGCRE macro. When using the z/OS console, you add `/cpf` to the start of a command.

Before you begin

Not all commands can be issued by the z/OS console. Within the command description topics (the children of MQSC commands reference), each command that can be issued by the console is identified by the character 'C'. Table 2 in "Command summary for IBM MQ for z/OS" summarizes the MQSC commands and the sources from which they can be issued.

You cannot issue IBM MQ commands using the IMS/SSR command format from an IMS terminal. This function is not supported by the IMS adapter.

The input field provided by SDSF might not be long enough for some commands, particularly those commands for channels.

The maximum amount of data that can be displayed as a result of a command typed in at the console is 32 KB.

About this task

If you are a suitably authorized user, you can issue IBM MQ commands from the z/OS console or equivalent (such as SDSF/TSO).

When using the z/OS console, you need to add `/cpf` to the start of a command, where `cpf` is the command prefix for the queue manager subsystem.

The following steps refer to commands and attributes using their MQSC command names rather than their PCF names.

Procedure

- Use command prefix strings

  Each IBM MQ command must be prefixed with a command prefix string (CPF).

  Because more than one IBM MQ subsystem can run under z/OS, the CPF is used to indicate which IBM MQ subsystem processes the command.

  For example, to start the queue manager for a subsystem called CSQ1, where CPF is ' +CSQ1 ', you issue the following command from the operator console:

  ```
  +CSQ1 START QMGR
  ```

  This CPF must be defined in the subsystem name table (for the subsystem CSQ1), as described in Defining command prefix strings (CPFs). In the examples, the string +CSQ1 is used as the command prefix.
- Use the z/OS console to issue commands

  You can type simple commands from the z/OS console, for example:

  ```
  +CSQ1 DISPLAY QUEUE(TRANSMIT.QUEUE.PROD) TYPE(QLOCAL)
  ```

  However, for complex commands or for sets of commands that you issue frequently, the other methods of issuing commands are better.
- Receive command responses

  Direct responses to commands are sent to the console that issued the command. IBM MQ supports the *Extended Console Support* (EMCS) function available in z/OS, and therefore consoles with 4 byte IDs can be used. Additionally, all commands except START QMGR and STOP QMGR support the use of Command and Response Tokens (CARTs) when the command is issued by a program using the MGCRE macro.

Related tasks

"Using the operations and control panels on z/OS" on page 475
You use these panels for defining, displaying, altering, or deleting IBM MQ objects. Use the panels for day-to-day administration and for making small changes to objects.

Preparing sample applications for the TSO environment on z/OS

z/OS Initialization commands for IBM MQ for z/OS

Initialization commands can be used to control the queue manager startup.

Commands in the initialization input data sets are processed when IBM MQ is initialized on queue manager startup. Three types of command can be issued from the initialization input data sets:

- Commands to define IBM MQ entities that cannot be defined elsewhere, for example DEFINE BUFFPOOL.

  These commands must reside in the data set identified by the DD name CSQINP1. They are processed before the restart phase of initialization. They cannot be issued through the console, operations and control panels, or an application program. The responses to these commands are written to the sequential data set that you refer to in the CSQOUT1 statement of the started task procedure.
- Commands to define IBM MQ objects that are recoverable after restart. These definitions must be specified in the data set identified by the DD name CSQINP2. They are stored in page set zero. CSQINP2 is processed after the restart phase of initialization. The responses to these commands are written to the sequential data set that you refer to in the CSQOUT2 statement of the started task procedure.
- Commands to manipulate IBM MQ objects. These commands must also be specified in the data set identified by the DD name CSQINP2. For example, the IBM MQ-supplied sample contains an ALTER

QMGR command to specify a dead-letter queue for the subsystem. The response to these commands is written to the CSQOUT2 output data set.

Note: If IBM MQ objects are defined in CSQINP2, IBM MQ attempts to redefine them each time the queue manager is started. If the objects already exist, the attempt to define them fails. If you need to define your objects in CSQINP2, you can avoid this problem by using the REPLACE parameter of the DEFINE commands, however, this overrides any changes that were made during the previous run of the queue manager.

Sample initialization data set members are supplied with IBM MQ for z/OS. They are described in Sample definitions supplied with IBM MQ.

Initialization commands for distributed queuing

You can also use the CSQINP2 initialization data set for the START CHINIT command. If you need a series of other commands to define your distributed queuing environment (for example, starting listeners), IBM MQ provides a third initialization input data set, called CSQINPX, that is processed as part of the channel initiator started task procedure.

The MQSC commands contained in the data set are executed at the end of channel initiator initialization, and output is written to the data set specified by the CSQOUTX DD statement. You might use the CSQINPX initialization data set to start listeners for example.

A sample channel initiator initialization data set member is supplied with IBM MQ for z/OS. It is described in Sample definitions supplied with IBM MQ.

Initialization commands for publish/Subscribe

If you need a series of commands to define your publish/subscribe environment (for example, when defining subscriptions), IBM MQ provides a fourth initialization input data set, called CSQINPT.

The MQSC commands contained in the data set are executed at the end of publish/subscribe initialization, and output is written to the data set specified by the CSQOUTT DD statement. You might use the CSQINPT initialization data set to define subscriptions for example.

A sample publish/subscribe initialization data set member is supplied with IBM MQ for z/OS. It is described in Sample definitions supplied with IBM MQ.

z/OS Private and global definitions on IBM MQ for z/OS

When you define an object on IBM MQ for z/OS, you can choose whether you want to share that definition with other queue managers (a *global* definition), or whether the object definition is to be used by one queue manager only (a *private* definition). This is called the object *disposition*.

Global definition

If your queue manager belongs to a queue sharing group, you can choose to share any object definitions you make with the other members of the group. This means that you have to define an object once only, reducing the total number of definitions required for the whole system.

Global object definitions are held in a *shared repository* (a Db2 shared database), and are available to all the queue managers in the queue sharing group. These objects have a disposition of GROUP.

Private definition

If you want to create an object definition that is required by one queue manager only, or if your queue manager is not a member of a queue sharing group, you can create object definitions that are not shared with other members of a queue sharing group.

Private object definitions are held on page set zero of the defining queue manager. These objects have a disposition of QMGR.

You can create private definitions for all types of IBM MQ objects except CF structures (that is, channels, namelists, process definitions, queues, queue managers, storage class definitions, and authentication information objects), and global definitions for all types of objects except queue managers.

IBM MQ automatically copies the definition of a group object to page set zero of each queue manager that uses it. You can alter the copy of the definition temporarily if you want, and IBM MQ allows you to refresh the page set copies from the repository copy if required.

IBM MQ always tries to refresh the page set copies from the repository copy at startup (for channel commands, this is done when the channel initiator restarts), or if the group object is changed.

Note: The copy of the definition is refreshed from the definition of the group, only if the definition of the group has changed after you created the copy of the definition.

This ensures that the page set copies reflect the version on the repository, including any changes that were made when the queue manager was inactive. The copies are refreshed by generating DEFINE REPLACE commands, therefore there are circumstances under which the refresh is not performed, for example:

- If a copy of the queue is open, a refresh that changes the usage of the queue fails.
- If a copy of a queue has messages on it, a refresh that deletes that queue fails.
- If a copy of a queue would require ALTER with FORCE to change it.

In these circumstances, the refresh is not performed on that copy, but is performed on the copies on all other queue managers.

If the queue manager is shut down and then restarted stand-alone, any local copies of objects are deleted, unless for example, the queue has associated messages.

There is a third object disposition that applies to local queues only. This allows you to create shared queues. The definition for a shared queue is held on the shared repository and is available to all the queue managers in the queue sharing group. In addition, the messages on a shared queue are also available to all the queue managers in the queue sharing group. This is described in Shared queues and queue sharing groups. Shared queues have an object disposition of SHARED.

The following table summarizes the effect of the object disposition options for queue managers started stand-alone, and as a member of a queue sharing group.

Disposition	Stand-alone queue manager	Member of a queue sharing group
QMGR	Object definition held on page set zero.	Object definition held on page set zero.
GROUP	Not allowed.	Object definition held in the shared repository. Local copy held on page set zero of each queue manager in the group.
SHARED	Not allowed.	Queue definition held in the shared repository. Messages available to any queue manager in the group.

Manipulating global definitions

If you want to change the definition of an object that is held in the shared repository, you need to specify whether you want to change the version on the repository, or the local copy on page set zero. Use the object disposition as part of the command to do this.

z/OS Directing commands to different queue managers on z/OS

You can use the *command scope* to control on which queue manager the command runs.

You can choose to execute a command on the queue manager where it is entered, or on a different queue manager in the queue sharing group. You can also choose to issue a particular command in parallel on all the queue managers in a queue sharing group. This is possible for both MQSC commands and PCF commands.

This is determined by the *command scope*. The command scope is used with the object disposition to determine which version of an object you want to work with.

For example, you might want to alter some of the attributes of an object, the definition of which is held in the shared repository.

- You might want to change the version on one queue manager only, and not make any changes to the version on the repository or those in use by other queue managers.
- You might want to change the version in the shared repository for future users, but leave existing copies unchanged.
- You might want to change the version in the shared repository, but also want your changes to be reflected immediately on all the queue managers in the queue sharing group that hold a copy of the object on their page set zero.

Use the command scope to specify whether the command is executed on this queue manager, another queue manager, or all queue managers. Use the object disposition to specify whether the object you are manipulating is in the shared repository (a group object), or is a local copy on page set zero (a queue manager object).

You do not have to specify the command scope and object disposition to work with a shared queue because every queue manager in the queue sharing group handles the shared queue as a single queue.

z/OS Using the operations and control panels on z/OS

You use these panels for defining, displaying, altering, or deleting IBM MQ objects. Use the panels for day-to-day administration and for making small changes to objects.

Before you begin

The IBM MQ for z/OS operations and controls panels (CSQOREXX) might not support all new function and parameters added from version 7 onwards. For example, there are no panels for the direct manipulation of topic objects or subscriptions. Use one of the following supported mechanisms to administer publish/subscribe definitions and other system controls that are not directly available from other panels:

1. IBM MQ Explorer
2. z/OS console
3. Programmable Command Format (PCF) messages
4. COMMAND function of CSQUTIL
5. IBM MQ Console

Note that the generic **Command** action in the CSQOREXX panels allows you to issue any valid MQSC command, including SMDS related commands. You can use all the commands that the COMMAND function of CSQUTIL issues.

You cannot issue the IBM MQ commands directly from the command line in the panels.

To use the operations and control panels, you must have the correct security authorization; this is described in the User IDs for command security and command resource security.

You cannot provide a user ID and password using CSQUTIL, or the CSQOREXX panels. Instead, if you user ID has UPDATE authority to the BATCH profile in MQCONN, you can bypass the **CHCKLOCL**(*REQUIRED* setting. See Using **CHCKLOCL** on locally bound applications for more information.

If you are setting up or changing many objects, use the COMMAND function of the CSQUTIL utility program. See “Using the CSQUTIL utility for IBM MQ for z/OS” on page 485.

About this task

The operations and control panels support the controls for the channel initiator (for example, to start a channel or a TCP/IP listener), for clustering, and for security. They also enable you to display information about threads and page set usage.

The panels work by sending MQSC type IBM MQ commands to a queue manager, through the system command input queue.

Example

This is the panel that displays when you start a panel session:

```
                  IBM MQ for z/OS - Main Menu

Complete fields. Then press Enter.

Action  . . . . . . . . . . 1     0. List with filter   4. Manage
                                  1. List or Display    5. Perform
                                  2. Define like        6. Start
                                  3. Alter              7. Stop
                                  8. Command
Object type . . . . . . . . CHANNEL       +
Name  . . . . . . . . . . . *
Disposition . . . . . . . . A  Q=Qmgr, C=Copy, P=Private, G=Group,
                               S=Shared, A=All

Connect name  . . . . . . . MQ1C  - local queue manager or group
Target queue manager  . . . MQ1C
        - connected or remote queue manager for command input
Action queue manager  . . . MQ1C  - command scope in group
Response wait time  . . . . 30    5 - 999 seconds

(C) Copyright IBM Corporation 1993, 2024. All rights reserved.

Command ===>
 F1=Help     F2=Split     F3=Exit      F4=Prompt    F9=SwapNext F10=Messages
F12=Cancel
```

From this panel you can perform actions such as these:

- Choose the local queue manager you want and whether you want the commands issued on that queue manager, on a remote queue manager, or on another queue manager in the same queue sharing group as the local queue manager. Over type the queue manager name if you need to change it.
- Select the action you want to perform by typing in the appropriate number in the **Action** field.
- Specify the object type that you want to work with. Press function key F1 for help about the object types if you are not sure what they are.
- Specify the disposition of the object type that you want to work with.
- Display a list of objects of the type specified. Type in an asterisk (*) in the **Name** field and press **Enter** to display a list of objects (of the type specified) that have already been defined on the action queue manager. You can then select one or more objects to work with in sequence. All the actions are available from the list.

Note: You are recommended to make choices that result in a list of objects being displayed, and then work from that list. Use the **Display** action, because that is allowed for all object types.

z/OS Invocation and rules for the operations and control panels

You can control IBM MQ and issue control commands through the ISPF panels.

How to access the IBM MQ operations and control panels

If the ISPF/PDF primary options menu has been updated for IBM MQ, you can access the IBM MQ operations and control panels from that menu. For details about updating the menu, see the Task 20: Set up the operations and control panels.

You can access the IBM MQ operations and control panels from the TSO command processor panel (typically option 6 on the ISPF/PDF primary options menu). The name of the exec that you run to do this is CSQOREXX. It has two parameters; `thlqual` is the high-level qualifier for the IBM MQ libraries to be used, and `langletter` is the letter identifying the national language libraries to be used (for example, E for U.S. English). The parameters can be omitted if the IBM MQ libraries are permanently installed in your ISPF setup. Alternatively, you can issue CSQOREXX from the TSO command line.

These panels are designed to be used by operators and administrators with a minimum of formal training. Read these instructions with the panels running and try out the different tasks suggested.

Note: While using the panels, temporary dynamic queues with names of the form SYSTEM.CSQOREXX.* are created.

Rules for the operations and control panels

See Rules for naming IBM MQ objects about the general rules for IBM MQ character strings and names. However, there are some rules that apply only to the operations and control panels:

- Do not enclose strings, for example descriptions, in single or double quotation marks.
- If you include an apostrophe or quotation mark in a text field, you do not have to repeat it or add an escape character. The characters are saved exactly as you type them; for example:

  ```
  This is Maria's queue
  ```

  The panel processor doubles them for you to pass them to IBM MQ. However, if it has to truncate your data to do this, it does so.
- You can use uppercase or lowercase characters in most fields, and they are folded to uppercase characters when you press Enter. The exceptions are:
  - Storage class names and coupling facility structure names, which must start with uppercase A through Z and be followed by uppercase A through Z or numeric characters.
  - Certain fields that are not translated. These include:
    - Application ID
    - Description
    - Environment data
    - Object names (but if you use a lowercase object name, you might not be able to enter it at a z/OS console)
    - Remote system name
    - Trigger data
    - User data
- In names, leading blanks and leading underscores are ignored. Therefore, you cannot have object names beginning with blanks or underscores.
- Underscores are used to show the extent of blank fields. When you press Enter, trailing underscores are replaced by blanks.
- Many description and text fields are presented in multiple parts, each part being handled by IBM MQ independently. This means that trailing blanks are retained and the text is not contiguous.

Blank fields

When you specify the **Define** action for an IBM MQ object, each field on the define panel contains a value. See the general help (extended help) for the display panels for information about where IBM MQ gets the values. If you type over a field with blanks, and blanks are not allowed, IBM MQ puts the installation default value in the field or prompts you to enter the required value.

When you specify the **Alter** action for an IBM MQ object, each field on the alter panel contains the current value for that field. If you type over a field with blanks, and blanks are not allowed, the value of that field is unchanged.

z/OS Objects and actions on z/OS

The operations and control panels offer you many different types of object and a number of actions that you can perform on them.

The actions are listed on the initial panel and enable you to manipulate the objects and display information about them. These objects include all the IBM MQ objects, together with some extra ones. The objects fall into the following categories.

- Queues, processes, authentication information objects, namelists, storage classes and CF structures
- Channels
- Cluster objects
- Queue manager and security
- Connections
- System

Refer to Actions for a cross-reference table of the actions which can be taken with the IBM MQ objects.

Queues, processes, authentication information objects, namelists, storage classes and CF structures

These are the basic IBM MQ objects. There can be many of each type. They can be listed, listed with filter, defined, and deleted, and have attributes that can be displayed and altered, using the LIST or DISPLAY, LIST with FILTER, DEFINE LIKE, MANAGE, and ALTER actions. (Objects are deleted using the MANAGE action.)

This category consists of the following objects:

QLOCAL	Local queue
QREMOTE	Remote queue
QALIAS	Alias queue for indirect reference to a queue
QMODEL	Model queue for defining queues dynamically
QUEUE	Any type of queue
QSTATUS	Status of a local queue
PROCESS	Information about an application to be started when a trigger event occurs
AUTHINFO	Authentication information: definitions required to perform Certificate Revocation List (CRL) checking using LDAP servers
NAMELIST	List of names, such as queues or clusters
STGCLASS	Storage class
CFSTRUCT	coupling facility (CF) structure
CFSTATUS	Status of a CF structure

Channels

Channels are used for distributed queuing. There can be many of each type, and they can be listed, listed with filter, defined, deleted, displayed, and altered. They also have other functions available using the START, STOP and PERFORM actions. PERFORM provides reset, ping, and resolve channel functions.

This category consists of the following objects:

CHANNEL	Any type of channel
SENDER	Sender channel

SERVER	Server channel
RECEIVER	Receiver channel
REQUESTER	Requester channel
CLUSRCVR	Cluster-receiver channel
CLUSSDR	Cluster-sender channel
SVRCONN	Server-connection channel
CLNTCONN	Client-connection channel
CHSTATUS	Status of a channel connection

Cluster objects

Cluster objects are created automatically for queues and channels that belong to a cluster. The base queue and channel definitions can be on another queue manager. There can be many of each type, and names can be duplicated. They can be listed, listed with filter, and displayed. PERFORM, START, and STOP are also available through the LIST actions.

This category consists of the following objects:

CLUSQ	Cluster queue, created for a queue that belongs to a cluster
CLUSCHL	Cluster channel, created for a channel that belongs to a cluster
CLUSQMGR	Cluster queue manager, the same as a cluster channel but identified by its queue manager name

Cluster channels and cluster queue managers do have the PERFORM, START and STOP actions, but only indirectly through the DISPLAY action.

Queue manager and security

Queue manager and security objects have a single instance. They can be listed, and have attributes that can be displayed and altered (using the LIST or DISPLAY, and ALTER actions), and have other functions available using the PERFORM action.

This category consists of the following objects:

MANAGER	Queue manager: the PERFORM action provides suspend and resume cluster functions
SECURITY	Security functions: the PERFORM action provides refresh and reverify functions

Connection

Connections can be listed, listed with filter and displayed.

This category consists only of the connection object, CONNECT.

System

A collection of other functions. This category consists of the following objects:

SYSTEM	System functions
CONTROL	Synonym for SYSTEM

The functions available are:

LIST or DISPLAY	Display queue sharing group, distributed queuing, page set, or data set usage information.
PERFORM	Refresh or reset clustering

START Start the channel initiator or listeners

STOP Stop the channel initiator or listeners

Actions

The actions that you can perform for each type of object are shown in the following table:

Table 27. Valid operations and control panel actions for IBM MQ objects

Object	Alter	Define like	Manage (1)	List or Display	List with Filter	Perform	Start	Stop
AUTHINFO	X	X	X	X	X			
CFSTATUS				X				
CFSTRUCT	X	X	X	X	X			
CHANNEL	X	X	X	X	X	X	X	X
CHSTATUS				X	X			
CLNTCONN	X	X	X	X	X			
CLUSCHL				X	X	X(2)	X(2)	X(2)
CLUSQ				X	X			
CLUSQMGR				X	X	X(2)	X(2)	X(2)
CLUSRCVR	X	X	X	X	X	X	X	X
CLUSSDR	X	X	X	X	X	X	X	X
CONNECT				X	X			
CONTROL				X		X	X	X
MANAGER	X			X		X		
NAMELIST	X	X	X	X	X			
PROCESS	X	X	X	X	X			
QALIAS	X	X	X	X	X			
QLOCAL	X	X	X	X	X			
QMODEL	X	X	X	X	X			
QREMOTE	X	X	X	X	X			
QSTATUS				X	X			
QUEUE	X	X	X	X	X			
RECEIVER	X	X	X	X	X	X	X	X
REQUESTER	X	X	X	X	X	X	X	X
SECURITY	X			X		X		
SENDER	X	X	X	X	X	X	X	X
SERVER	X	X	X	X	X	X	X	X
SVRCONN	X	X	X	X	X		X	X
STGCLASS	X	X	X	X	X			

Table 27. Valid operations and control panel actions for IBM MQ objects (continued)								
Object	**Alter**	**Define like**	**Manage (1)**	**List or Display**	**List with Filter**	**Perform**	**Start**	**Stop**
SYSTEM				X		X	X	X

Note:

1. Provides `Delete` and other functions
2. Using the `List or Display` action

z/OS Object dispositions on z/OS

You can specify the *disposition* of the object with which you need to work. The disposition signifies where the object **definition** is kept, and how the object behaves.

The disposition is significant only if you are working with any of the following object types:

- queues
- channels
- processes
- namelists
- storage classes
- authentication information objects

If you are working with other object types, the disposition is disregarded.

Permitted values are:

Q
QMGR. The object definitions are on the page set of the queue manager and are accessible only by the queue manager.

C
COPY. The object definitions are on the page set of the queue manager and are accessible only by the queue manager. They are local copies of objects defined as having a disposition of GROUP.

P
PRIVATE. The object definitions are on the page set of the queue manager and are accessible only by the queue manager. The objects have been defined as having a disposition of QMGR or COPY.

G
GROUP. The object definitions are in the shared repository, and are accessible by all queue managers in the queue sharing group.

S
SHARED. This disposition applies only to local queues. The queue definitions are in the shared repository, and are accessible by all queue managers in the queue sharing group.

A
ALL. If the action queue manager is either the target queue manager, or *, objects of **all** dispositions are included; otherwise, objects of QMGR and COPY dispositions only are included. This is the default.

z/OS Selecting a queue manager, defaults, and levels using the ISPF control panel on z/OS

You can use the CSQOREXX exec in ISPF to control your queue managers.

While you are viewing the initial panel, you are not connected to any queue manager. However, as soon as you press Enter, you are connected to the queue manager, or a queue manager in the queue sharing group named in the **Connect name** field. You can leave this field blank; this means that you are using the default

queue manager for batch applications. This is defined in CSQBDEFV (see Task 19: Set up Batch, TSO, and RRS adapters for information about this).

Use the **Target queue manager** field to specify the queue manager where the actions you request are to be performed. If you leave this field blank, it defaults to the queue manager specified in the **Connect name** field. You can specify a target queue manager that is not the one you connect to. In this case, you would normally specify the name of a remote queue manager object that provides a queue manager alias definition (the name is used as the *ObjectQMgrName* when opening the command input queue). To do this, you must have suitable queues and channels set up to access the remote queue manager.

The **Action queue manager** field allows you to specify a queue manager that is in the same queue sharing group as the queue manager specified in the **Target queue manager** field to be the queue manager where the actions you request are to be performed. If you specify * in this field, the actions you request are performed on all queue managers in the queue sharing group. If you leave this field blank, it defaults to the value specified in the **Target queue manager** field. The **Action queue manager** field corresponds to using the CMDSCOPE command modifier described in The MQSC commands.

Queue manager defaults

If you leave any queue manager fields blank, or choose to connect to a queue sharing group, a secondary window opens when you press **Enter**. This window confirms the names of the queue managers you will be using. Press **Enter** to continue. When you return to the initial panel after having made some requests, you find fields completed with the actual names.

Queue manager levels

If the action queue manager is not at IBM MQ 8.0.0 or later, some fields are not displayed, and some values cannot be entered. A few objects and actions are disallowed. In such cases, a secondary window opens asking for you to confirm that you want to proceed.

z/OS Using the function keys and command line with the ISPF control panels on z/OS

To use the panels, you must use the function keys or enter the equivalent commands in the ISPF control panel command area.

- Function keys
  - Processing your actions
  - “Displaying IBM MQ user messages” on page 483
  - Canceling your actions
  - Getting help
- Using the command line

Function keys

The function keys have special settings for IBM MQ. (This means that you cannot use the ISPF default values for the function keys; if you have previously used the `KEYLIST OFF ISPF` command anywhere, you must type `KEYLIST ON` in the command area of any operations and control panel and then press Enter to enable the IBM MQ settings.)

These function key settings can be displayed on the panels, as shown in “Using the operations and control panels on z/OS” on page 475. If the settings are not shown, type `PFSHOW` in the command area of any operations and control panel and then press **Enter**. To remove the display of the settings, use the command `PFSHOW OFF`.

The function key settings in the operations and control panels conform to CUA standards. Although you can change the key setting through normal ISPF procedures (such as the **KEYLIST** utility), you are not recommended to do so.

Note: Using the **PFSHOW** and **KEYLIST** commands affects any other logical ISPF screens that you have, and their settings remain when you leave the operations and control panels.

Processing your actions

Press **Enter** to carry out the action requested on a panel. The information from the panel is sent to the queue manager for processing.

Each time you press **Enter** in the panels, IBM MQ generates one or more operator messages. If the operation was successful, you get confirmation message CSQ9022I, otherwise you get some error messages.

Displaying IBM MQ user messages

Press function key F10 in any panel to see the IBM MQ user messages.

Canceling your actions

On the initial panel, both F3 and F12 exit the operations and control panels and return you to ISPF. No information is sent to the queue manager.

On any other panel, press function keys F3 or F12 to leave the current panel **ignoring any data you have typed since last pressing Enter**. Again, no information is sent to the queue manager.

- F3 takes you straight back to the initial panel.
- F12 takes you back to the previous panel.

Getting help

Each panel has help panels associated with it. The help panels use the ISPF protocols:

- Press function key F1 on any panel to see general help (extended help) about the task.
- Press function key F1 with the cursor on any field to see specific help about that field.
- Press function key F5 from any field help panel to get the general help.
- Press function key F3 to return to the base panel, that is, the panel from which you pressed function key F1.
- Press function key F6 from any help panel to get help about the function keys.

If the help information carries on into a second or subsequent pages, a **More** indicator is displayed in the upper-right of the panel. Use these function keys to navigate through the help pages:

- F11 to get to the next help page (if there is one).
- F10 to get back to the previous help page (if there is one).

Using the command line

You never need to use the command line to issue the commands used by the operations and control panels because they are available from function keys. The command line is provided to allow you to enter normal ISPF commands (like **PFSHOW**).

The ISPF command PANELID ON displays the name of the current CSQOREXX panel.

The command line is initially displayed in the default position at the bottom of the panels, regardless of what ISPF settings you have. You can use the SETTINGS ISPF command from any of the operations and control panels to change the position of the command line. The settings are remembered for subsequent sessions with the operations and control panels.

z/OS Using the IBM MQ for z/OS utilities

IBM MQ for z/OS provides a set of utility programs that you can use to help with system administration.

IBM MQ for z/OS supplies a set of utility programs to help you perform various administrative tasks, including the following:

- Manage message security policies.
- Perform backup, restoration, and reorganization tasks.
- Issue commands and process object definitions.
- Generate data-conversion exits.
- Modify the bootstrap data set.
- List information about the logs.
- Print the logs.
- Set up Db2 tables and other Db2 utilities.
- Process messages on the dead-letter queue.

The message security policy utility

The message security policy utility (CSQ0UTIL) runs as a stand-alone utility to manage message security policies. See The message security policy utility (CSQ0UTIL) for more information.

The CSQUTIL utility

This is a utility program provided to help you with backup, restore and reorganize tasks. See “Using the CSQUTIL utility for IBM MQ for z/OS” on page 485.

The data conversion exit utility

The IBM MQ for z/OS data conversion exit utility (**CSQUCVX**) runs as a stand-alone utility to create data conversion exit routines.

The change log inventory utility

The IBM MQ for z/OS change log inventory utility program (**CSQJU003**) runs as a stand-alone utility to change the bootstrap data set (BSDS). You can use the utility to perform the following functions:

- Add or delete active or archive log data sets.
- Supply passwords for archive logs.

The print log map utility

The IBM MQ for z/OS print log map utility program (**CSQJU004**) runs as a stand-alone utility to list the following information:

- Log data set name and log RBA association for both copies of all active and archive log data sets. If dual logging is not active, there is only one copy of the data sets.
- Active log data sets available for new log data.
- Contents of the queue of checkpoint records in the bootstrap data set (BSDS).
- Contents of the archive log command history record.
- System and utility time stamps.

The log print utility

The log print utility program (**CSQ1LOGP**) is run as a stand-alone utility. You can run the utility specifying:

- A bootstrap data set (BSDS)

- Active logs (with no BSDS)
- Archive logs (with no BSDS)

The queue sharing group utility

The queue sharing group utility program (**CSQ5PQSG**) runs as a stand-alone utility to set up Db2 tables and perform other Db2 tasks required for queue sharing groups.

The active log preformat utility

The active log preformat utility (**CSQJUFMT**) formats active log data sets before they are used by a queue manager. If the active log data sets are preformatted by the utility, log write performance is improved on the queue manager's first pass through the active logs.

The dead-letter queue handler utility

The dead-letter queue handler utility program (**CSQUDLQH**) runs as a stand-alone utility. It checks messages that are on the dead-letter queue and processes them according to a set of rules that you supply to the utility.

The queue load and unload utility

The queue load and unload utility copies or moves the contents of a queue, or its messages, to a file. The utility was originally shipped as the **QLOAD** utility in IBM MQ Supportpac MO03. From IBM MQ 8.0 it is integrated into the product as executable module **CSQUDMSG** in the SCSQLOAD library, with an alias of QLOAD for compatibility. Sample JCL is provided as member CSQ4QLOD in SCSQPROC.

The equivalent utility for Multiplatforms is called **dmpmqmsg**. For details of the available options, including the differences for z/OS, see dmpmqmsg (queue load and unload).

You can also reload messages as described in Restoring messages from a data set to a queue (LOAD) on z/OS and Restoring messages from a data set to a queue (SLOAD) on z/OS.

z/OS Using the CSQUTIL utility for IBM MQ for z/OS

The CSQUTIL utility program is provided with IBM MQ for z/OS to help you perform backup, restoration, and reorganization tasks, and to issue commands and process object definitions.

About this task

Use this utility program to invoke the following functions. For example, you can issue commands from a sequential data set using the COMMAND function of the CSQUTIL utility. This function transfers the commands, as messages, to the *system-command input queue* and waits for the response, which is printed together with the original commands in SYSPRINT.

For more information about the CSQUTIL utility program, see IBM MQ utility program (CSQUTIL).

Procedure

- COMMAND

  Use this function to issue MQSC commands, to record object definitions, and to make client-channel definition files.
- COPY

  Use this function to read the contents of a named IBM MQ for z/OS message queue or the contents of all the queues of a named page set, and put them into a sequential file and retain the original queue.

- COPYPAGE

  Use this function to copy whole page sets to larger page sets.
- EMPTY

  Use this function to delete the contents of a named IBM MQ for z/OS message queue or the contents of all the queues of a named page set, retaining the definitions of the queues.
- FORMAT

  Use this function to format IBM MQ for z/OS page sets.
- Restoring messages from a data set to a queue (LOAD) on z/OS

  Use this function to restore the contents of a named IBM MQ for z/OS message queue or the contents of all the queues of a named page set from a sequential file created by the COPY function.
- PAGEINFO

  Use this function to extract page set information from one or more page sets.
- RESETPAGE

  Use this function to copy whole page sets to other page set data sets and reset the log information in the copy.
- SCOPY

  Use this function to copy the contents of a queue to a data set while the queue manager is offline.
- SDEFS

  Use this function to produce a set of define commands for objects while the queue manager is offline.
- SLOAD

  Use this function to restore messages from the destination data set of an earlier COPY or SCOPY operation. SLOAD processes a single queue.
- SWITCH

  Use this function to switch or query the transmission queue associated with cluster-sender channels.

z/OS Using the Command Facility on z/OS

Use the editor to enter or amend MQSC commands to be passed to the queue manager.

From the primary panel, CSQOPRIA, select option **8 Command**, to start the Command Facility.

You are presented with an edit session of a sequential file, *prefix*.CSQUTIL.COMMANDS, used as input to the CSQUTIL COMMAND function; see Issuing commands to IBM MQ.

You do not need to prefix commands with the command prefix string (CPF).

You can continue MQSC commands on subsequent lines by terminating the current line with the continuation characters **+** or **-**. Alternatively, use line edit mode to provide long MQSC commands or the values of long attribute values within the command.

line edit

To use line edit, move the cursor to the appropriate line in the edit panel and use **F4** to display a single line in a scrollable panel. A single line can be up to 32 760 bytes of data.

To leave line edit:

- **F3 exit** saves changes made to the line and exits
- **F12 cancel** returns to the edit panel discarding changes made to the line.

To discard changes made in the edit session, use **F12 cancel** to terminate the edit session leaving the contents of the file unchanged. Commands are not executed.

Executing commands

When you have finished entering MQSC commands, terminate the edit session with **F3 exit** to save the contents of the file and invoke CSQUTIL to pass the commands to the queue manager. The output from command processing is held in file *prefix*.CSQUTIL.OUTPUT. An edit session opens automatically on this file so that you can view the responses. Press **F3 exit** to exit this session and return to the main menu.

z/OS Working with IBM MQ objects on z/OS

Many of the tasks described in this documentation involve manipulating IBM MQ objects. The object types are queue managers, queues, process definitions, namelists, channels, client connection channels, listeners, services, and authentication information objects.

- Defining simple queue objects
- Defining other types of objects
- Working with object definitions
- Working with namelists

Defining simple queue objects

To define a new object, use an existing definition as the basis for it. You can do this in one of three ways:

- By selecting an object that is a member of a list displayed as a result of options selected on the initial panel. You then enter action type 2 (**Define like**) in the action field next to the selected object. Your new object has the attributes of the selected object, except the disposition. You can then change any attributes in your new object as you require.
- On the initial panel, select the **Define like** action type, enter the type of object that you are defining in the **Object type** field, and enter the name of a specific existing object in the **Name** field. Your new object has the same attributes as the object you named in the **Name** field, except the disposition. You can then change any attributes in your new object definition as you require.
- By selecting the **Define like** action type, specifying an object type and then leaving the **Name** field blank. You can then define your new object and it has the default attributes defined for your installation. You can then change any attributes in your new object definition as you require.

Note: You do not enter the name of the object you are defining on the initial panel, but on the **Define** panel you are presented with.

The following example demonstrates how to define a local queue using an existing queue as a template.

Defining a local queue

To define a local queue object from the operations and control panels, use an existing queue definition as the basis for your new definition. There are several panels to complete. When you have completed all the panels and you are satisfied that the attributes are correct, press Enter to send your definition to the queue manager, which then creates the actual queue.

Use the **Define like** action either on the initial panel or against an object entry in a list displayed as a result of options selected on the initial panel.

For example, starting from the initial panel, complete these fields:

Action	2 (Define like)
Object type	QLOCAL
Name	QUEUE.YOU.LIKE. This is the name of the queue that provides the attributes for your new queue.

Press Enter to display the **Define a Local Queue** panel. The queue name field is blank so that you can supply the name for the new queue. The description is that of the queue upon which you are basing this new definition. Over type this field with your own description for the new queue.

The values in the other fields are those of the queue upon which you are basing this new queue, except the disposition. You can over type these fields as you require. For example, type Y in the **Put enabled** field (if it is not already Y) if suitably authorized applications can put messages on this queue.

You get field help by moving the cursor into a field and pressing function key F1. Field help provides information about the values that can be used for each attribute.

When you have completed the first panel, press function key F8 to display the second panel.

Hints:

1. Do not press Enter at this stage, otherwise the queue will be created before you have a chance to complete the remaining fields. (If you do press Enter prematurely, do not worry; you can always alter your definition later on.)
2. Do not press function keys F3 or F12, or the data you typed will be lost.

Press function key F8 repeatedly to see and complete the remaining panels, including the trigger definition, event control, and backout reporting panels.

When your local queue definition is complete

When your definition is complete, press Enter to send the information to the queue manager for processing. The queue manager creates the queue according to the definition you have supplied. If you do not want the queue to be created, press function key F3 to exit and cancel the definition.

Defining other types of objects

To define other types of object, use an existing definition as the base for your new definition as explained in Defining a local queue.

Use the **Define like** action either on the initial panel or against an object entry in a list displayed as a result of options selected on the initial panel.

For example, starting from the initial panel, complete these fields:

Action	2 (Define like)
Object type	QALIAS, NAMELIST, PROCESS, CHANNEL, and other resource objects.
Name	Leave blank or enter the name of an existing object of the same type.

Press Enter to display the corresponding DEFINE panels. Complete the fields as required and then press Enter again to send the information to the queue manager.

Like defining a local queue, defining another type of object generally requires several panels to be completed. Defining a namelist requires some additional work, as described in “Working with namelists” on page 489.

Working with object definitions

When an object has been defined, you can specify an action in the **Action** field, to alter, display, or manage it.

In each case, you can either:

- Select the object you want to work with from a list displayed as a result of options selected on the initial panel. For example, having entered 1 in the **Action** field to display objects, Queue in the **Object type** field, and * in the **Name** field, you are presented with a list of all queues defined in the system. You can then select from this list the queue with which you need to work.

- Start from the initial panel, where you specify the object you are working with by completing the **Object type** and **Name** fields.

Altering an object definition
To alter an object definition, specify action 3 and press Enter to see the ALTER panels. These panels are very similar to the DEFINE panels. You can alter the values you want. When your changes are complete, press Enter to send the information to the queue manager.

Displaying an object definition
If you want to see the details of an object without being able to change them, specify action 1 and press Enter to see the DISPLAY panels. Again, these panels are similar to the DEFINE panels except that you cannot change any of the fields. Change the object name to display details of another object.

Deleting an object
To delete an object, specify action 4 (Manage) and the **Delete** action is one of the actions presented on the resulting menu. Select the **Delete** action.

You are asked to confirm your request. If you press function key F3 or F12, the request is canceled. If you press Enter, the request is confirmed and passed to the queue manager. The object you specified is then deleted.

Note: You cannot delete most types of channel object unless the channel initiator is started.

Working with namelists

When working with namelists, proceed as you would for other objects.

For the actions DEFINE LIKE or ALTER, press function key F11 to add names to the list or to change the names in the list. This involves working with the ISPF editor and all the normal ISPF edit commands are available. Enter each name in the namelist on a separate line.

When you use the ISPF editor in this way, the function key settings are the normal ISPF settings, and **not** those used by the other operations and control panels.

If you need to specify lowercase names in the list, specify CAPS(OFF) on the editor panel command line. When you do this, all the namelists that you edit in the future are in lowercase until you specify CAPS(ON).

When you have finished editing the namelist, press function key F3 to end the ISPF edit session. Then press Enter to send the changes to the queue manager.

Attention: If you do not press Enter at this stage but press function key F3 instead, you lose any updates that you have typed in.

z/OS Implementing the system using multiple cluster transmission queues

It makes no difference if the channel is used in a single cluster, or an overlapping cluster. When the channel is selected and started, the channel selects the transmission queue depending on the definitions.

Procedure

- If you are using the DEFCLXQ option, see “Using the automatic definition of queues and switching” on page 489.
- If you are using a staged approach, see “Changing your cluster-sender channels using a phased approach” on page 490.

z/OS Using the automatic definition of queues and switching

Use this option if you are planning on using the DEFCLXQ option. There will be a queue created for every channel, and every new channel.

Procedure

1. Review the definition of the SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE and change the attributes if required.

   This queue is defined in member SCSQPROC(csq4insx).
2. Create the SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE model queue.
3. Apply security policies for this model queue, and the SYSTEM.CLUSTER.TRANSMIT.** queues.

   For z/OS the channel initiator started task user ID needs:

   - Control access to CLASS(MQADMIN) for

     ```
     ssid.CONTEXT.SYSTEM.CLUSTER.TRANSMIT.channelname
     ```

   - Update access to CLASS(MQQUEUE) for

     ```
     ssid.SYSTEM.CLUSTER.TRANSMIT.channelname
     ```

z/OS Changing your cluster-sender channels using a phased approach

Use this option if you are planning on using a staged approach. This process allows you to move to the new cluster-sender channels at various times to suit the needs of your enterprise.

Before you begin

- Identify your business applications, and which channels are used.
- For the queues you use, display the clusters they are in.
- Display the channels to show the connection names, the names of the remote queue managers, and which clusters the channel supports.

About this task

- Create a transmission queue. On z/OS you might want to consider which page set you use for the queue.
- Set up security policy for the queue.
- Change any queue monitoring to include this queue name.
- Decide which channels are to use this transmission queue. The channels should have a similar name, so generic characters ' * ' in the CLCHNAME identify the channel.
- When you are ready to use the new function, alter the transmission queue to specify the name of the channels to use this transmission queue. For example CLUSTER1.TOPARIS, or CLUSTER1.* or *.TOPARIS
- Start the channels

Procedure

1. Use the DIS CLUSQMGR(xxxx) XMITQ command to display the cluster sender channels defined in the cluster, where *xxxx* is the name of the remote queue manager.
2. Set up the security profile for the transmission queue and give the queue access to the channel initiator.
3. Define the transmission queue to be used, and specify USAGE(XMITQ) INDXTYPE(CORRELID) SHARE and CLCHNAME(*value*)

   The channel initiator started task user ID needs the following access:

   ```
   alter class(MQADMIN) ssid.CONTEXT.SYSTEM.CLUSTER.TRANSMIT.channel
   update class(MQQUEUE ssid.SYSTEM.CLUSTER.TRANSMIT.channel
   ```

   and the user ID using the SWITCH command needs the following access:

```
alter cl(MQADMIN) ssid.QUEUE.queuename
```

4. Stop and restart the channels.

   The channel change occurs when the channel starts using an MQSC command, or you use CSQUTIL. You can identify which channels need to be restarted using the `SWITCH CHANNEL(*)STATUS` of CSQUTIL

   If you have problems when the channel is started, stop the channel, resolve the problems, and restart the channel.

   Note that you can change the CLCHNAME attribute as often as you need to.

   The value of CLCHNAME used is the one when the channel is started, so you can change the CLCHNAME definition while the channel continues to use the definitions from the time that it started. The channel uses the new definition when it is restarted.

z/OS Undoing a change to a transmission queue on z/OS

You need to have a process to backout a change if it the results are not as you expect.

What can go wrong?

If the new transmission queue is not what you expect:

1. Check the CLCHNAME is as you expect
2. Review the job log to check if the switch process has finished. If not, wait and check the new transmission queue of the channel later.

If you are using multiple cluster transmission queues, it is important that you design the transmission queues definitions explicitly and avoid complicated overlapping configuration. In this way, you can make sure that if there are problems, you can go back to the original queues and configuration.

If you encounter problems during the move to using a different transmission queue, you must resolve any problems before you can proceed with the change.

An existing change request must complete before a new change request can be made. For example, you:

1. Define a new transmission queue with a maximum depth of one and there are 10 messages waiting to be sent.
2. Change the transmission queue to specify the channel name in the CLCHNAME parameter.
3. Stop and restart the channel. The attempt to move the messages fails and reports the problems.
4. Change the CLCHNAME parameter on the transmission queue to be blank.
5. Stop and restart the channel. The channel continues to try and complete the original request, so the channel continues to use the new transmission queue.
6. Need to resolve the problems and restart the channel so the moving of messages completes successfully.

Next time the channel is restarted it picks up any changes, so if you had set CLCHNAME to blanks, the channel will not use the specified transmission queue.

In this example, changing the CLCHNAME on the transmission queue to blanks does not necessarily mean that the channel uses the SYSTEM.CLUSTER.TRANSMIT queue, as there might be other transmission queues whose CLCHNAME parameter match the channel name. For example, a generic name, or the queue manager attribute DEFCLXQ might be set to channel, so the channel uses a dynamic queue instead of the SYSTEM.CLUSTER.TRANSMIT queue.

z/OS Writing programs to administer IBM MQ for z/OS

You can write your own application programs to administer a queue manager. Use this topic to understand the requirements for writing your own administration programs.

Start of General-use programming interface information

This set of topics contains hints and guidance to enable you to issue IBM MQ commands from an IBM MQ application program.

Note: In this topic, the MQI calls are described using C-language notation. For typical invocations of the calls in the COBOL, PL/I, and assembler languages, see Function calls.

Understanding how it all works

In outline, the procedure for issuing commands from an application program is as follows:

1. Build an IBM MQ command into a type of IBM MQ message called a *request message*. The command can be in MQSC or PCF format.
2. Send (use MQPUT) this message to a special queue called the system-command input queue. The IBM MQ command processor runs the command.
3. Retrieve (use MQGET) the results of the command as *reply messages* on the reply-to queue. These messages contain the user messages that you need to determine whether your command was successful and, if it was, what the results were.

Then it is up to your application program to process the results.

This set of topics contains:

z/OS Preparing queues for administration programs

Administration programs require a number of predefined queues for system command input and receiving responses.

This section applies to commands in the MQSC format. For the equivalent in PCF, see “Utilización de los formatos de mandato programable de IBM MQ” on page 27.

Before you can issue any MQPUT or MQGET calls, you must first define, and then open, the queues you are going to use.

Defining the system-command input queue

The system-command input queue is a local queue called SYSTEM.COMMAND.INPUT. The supplied CSQINP2 initialization data set, thlqual.SCSQPROC(CSQ4INSG), contains a default definition for the system-command input queue. For compatibility with IBM MQ on other platforms, an alias of this queue, called SYSTEM.ADMIN.COMMAND.QUEUE is also supplied. See Sample definitions supplied with IBM MQ for more information.

Defining a reply-to queue

You must define a reply-to queue to receive reply messages from the IBM MQ command processor. It can be any queue with attributes that allow reply messages to be put on it. However, for normal operation, specify these attributes:

- USAGE(NORMAL)
- NOTRIGGER (unless your application uses triggering)

Avoid using persistent messages for commands, but if you choose to do so, the reply-to queue must not be a temporary dynamic queue.

The supplied CSQINP2 initialization data set, thlqual.SCSQPROC(CSQ4INSG), contains a definition for a model queue called SYSTEM.COMMAND.REPLY.MODEL. You can use this model to create a dynamic reply-to queue.

Note: Replies generated by the command processor can be up to 15 000 bytes in length.

If you use a permanent dynamic queue as a reply-to queue, your application should allow time for all PUT and GET operations to complete before attempting to delete the queue, otherwise MQRC2055 (MQRC_Q_NOT_EMPTY) can be returned. If this occurs, try the queue deletion again after a few seconds.

Opening the system-command input queue

Before you can open the system-command input queue, your application program must be connected to your queue manager. Use the MQI call `MQCONN` or `MQCONNX` to do this.

Then use the MQI call `MQOPEN` to open the system-command input queue. To use this call:

1. Set the **Options** parameter to MQOO_OUTPUT
2. Set the MQOD object descriptor fields as follows:

   ObjectType
   MQOT_Q (the object is a queue)

   ObjectName
   SYSTEM.COMMAND.INPUT

   ObjectQMgrName
   If you want to send your request messages to your local queue manager, leave this field blank. This means that your commands are processed locally.

   If you want your IBM MQ commands to be processed on a remote queue manager, put its name here. You must also have the correct queues and links set up, as described in Distributed queuing and clusters.

Opening a reply-to queue

To retrieve the replies from an IBM MQ command, you must open a reply-to queue. One way of doing this is to specify the model queue, SYSTEM.COMMAND.REPLY.MODEL in an `MQOPEN` call, to create a permanent dynamic queue as the reply-to queue. To use this call:

1. Set the **Options** parameter to MQOO_INPUT_SHARED
2. Set the MQOD object descriptor fields as follows:

   ObjectType
   MQOT_Q (the object is a queue)

   ObjectName
   The name of the reply-to queue. If the queue name you specify is the name of a model queue object, the queue manager creates a dynamic queue.

   ObjectQMgrName
   To receive replies on your local queue manager, leave this field blank.

   DynamicQName
   Specify the name of the dynamic queue to be created.

z/OS Using the command server

The command server is an IBM MQ component that works with the command processor component. You can send formatted messages to the command server which interprets the messages, runs the administration requests, and sends responses back to your administration application.

The command server reads request messages from the system-command input queue, verifies them, and passes the valid ones as commands to the command processor. The command processor processes the commands and puts any replies as reply messages on to the reply-to queue that you specify. The first reply message contains the user message CSQN205I. See "Interpreting the reply messages from the command server" on page 497 for more information. The command server also processes channel initiator and queue sharing group commands, wherever they are issued from.

Identifying the queue manager that processes your commands

The queue manager that processes the commands you issue from an administration program is the queue manager that owns the system-command input queue that the message is put onto.

Starting the command server

Normally, the command server is started automatically when the queue manager is started. It becomes available as soon as the message CSQ9022I 'START QMGR' NORMAL COMPLETION is returned from the START QMGR command. The command server is stopped when all the connected tasks have been disconnected during the system termination phase.

You can control the command server yourself using the START CMDSERV and STOP CMDSERV commands. To prevent the command server starting automatically when IBM MQ is restarted, you can add a STOP CMDSERV command to your CSQINP1 or CSQINP2 initialization data sets. However, this is not recommended as it prevents any channel initiator or queue sharing group commands being processed.

The STOP CMDSERV command stops the command server as soon as it has finished processing the current message, or immediately if no messages are being processed.

If the command server has been stopped by a STOP CMDSERV command in the program, no other commands from the program can be processed. To restart the command server, you must issue a START CMDSERV command from the z/OS console.

If you stop and restart the command server while the queue manager is running, all the messages that are on the system-command input queue when the command server stops are processed when the command server is restarted. However, if you stop and restart the queue manager after the command server is stopped, only the persistent messages on the system-command input queue are processed when the command server is restarted. All nonpersistent messages on the system-command input queue are lost.

Sending commands to the command server

For each command, you build a message containing the command, then put it onto the system-command input queue.

Building a message that includes IBM MQ commands

You can incorporate IBM MQ commands in an application program by building request messages that include the required commands. For each such command you:

1. Create a buffer containing a character string representing the command.
2. Issue an MQPUT call specifying the buffer name in the **buffer** parameter of the call.

The simplest way to do this in C is to define a buffer using 'char'. For example:

```
char message_buffer[ ] = "ALTER QLOCAL(SALES) PUT(ENABLED)";
```

When you build a command, use a null-terminated character string. Do not specify a command prefix string (CPF) at the start of a command defined in this way. This means that you do not have to alter your command scripts if you want to run them on another queue manager. However, you must take into account that a CPF is included in any response messages that are put onto the reply-to queue.

The command server folds all lowercase characters to uppercase unless they are inside quotation marks.

Commands can be any length up to a maximum 32 762 characters.

Putting messages on the system-command input queue

Use the MQPUT call to put request messages containing commands on the system-command input queue. In this call you specify the name of the reply-to queue that you have already opened.

To use the MQPUT call:

1. Set these MQPUT parameters:

   Hconn
   The connection handle returned by the MQCONN or MQCONNX call.

   Hobj
   The object handle returned by the MQOPEN call for the system-command input queue.

   BufferLength
   The length of the formatted command.

   Buffer
   The name of the buffer containing the command.

2. Set these MQMD fields:

   MsgType
   MQMT_REQUEST

   Format
   MQFMT_STRING or MQFMT_NONE

   If you are not using the same code page as the queue manager, set *CodedCharSetId* as appropriate and set MQFMT_STRING, so that the command server can convert the message. Do not set MQFMT_ADMIN, as that causes your command to be interpreted as PCF.

   ReplyToQ
   Name of your reply-to queue.

   ReplyToQMgr
   If you want replies sent to your local queue manager, leave this field blank. If you want your IBM MQ commands to be sent to a remote queue manager, put its name here. You must also have the correct queues and links set up, as described in Distributed queuing and clusters.

3. Set any other MQMD fields, as required. You should normally use nonpersistent messages for commands.
4. Set any *PutMsgOpts* options, as required.

   If you specify MQPMO_SYNCPOINT (the default), you must follow the MQPUT call with a syncpoint call.

Using MQPUT1 and the system-command input queue

If you want to put just one message on the system-command input queue, you can use the **MQPUT1** call. This call combines the functions of an **MQOPEN**, followed by an **MQPUT** of one message, followed by an **MQCLOSE**, all in one call. If you use this call, modify the parameters accordingly. See Putting one message on a queue using the MQPUT1 call for details.

z/OS Retrieving replies to your commands

The command server sends a response to a reply queue for each request message it receives. Any administration application must receive, and handle the reply messages.

When the command processor processes your commands, any reply messages are put onto the reply-to queue specified in the MQPUT call. The command server sends the reply messages with the same persistence as the command message it received.

Waiting for a reply

Use the MQGET call to retrieve a reply from your request message. One request message can produce several reply messages. For details, see "Interpreting the reply messages from the command server" on page 497.

You can specify a time interval that an MQGET call waits for a reply message to be generated. If you do not get a reply, use the checklist beginning in topic "If you do not receive a reply" on page 498.

To use the MQGET call:

1. Set these parameters:

   Hconn
   The connection handle returned by the MQCONN or MQCONNX call.

   Hobj
   The object handle returned by the MQOPEN call for the reply-to queue.

   Buffer
   The name of the area to receive the reply.

   BufferLength
   The length of the buffer to receive the reply. This must be a minimum of 80 bytes.

2. To ensure that you only get the responses from the command that you issued, you must specify the appropriate *MsgId* and *CorrelId* fields. These depend on the report options, MQMD_REPORT, you specified in the MQPUT call:

   MQRO_NONE
   Binary zero, '00...00' (24 nulls).

   MQRO_NEW_MSG_ID
   Binary zero, '00...00' (24 nulls).

   This is the default if none of these options has been specified.

   MQRO_PASS_MSG_ID
   The *MsgId* from the **MQPUT**.

   MQRO_NONE
   The *MsgId* from the MQPUT call.

   MQRO_COPY_MSG_ID_TO_CORREL_ID
   The *MsgId* from the MQPUT call.

   This is the default if none of these options has been specified.

   MQRO_PASS_CORREL_ID
   The *CorrelId* from the MQPUT call.

   For more details on report options, see Report options and message flags.

3. Set the following *GetMsgOpts* fields:

   Options
   MQGMO_WAIT

   If you are not using the same code page as the queue manager, set MQGMO_CONVERT, and set *CodedCharSetId* as appropriate in the MQMD.

   WaitInterval
   For replies from the local queue manager, try 5 seconds. Coded in milliseconds, this becomes 5 000. For replies from a remote queue manager, and channel control and status commands, try 30 seconds. Coded in milliseconds, this becomes 30 000.

Discarded messages

If the command server finds that a request message is not valid, it discards this message and writes the message CSQN205I to the named reply-to queue. If there is no reply-to queue, the CSQN205I message is put onto the dead-letter queue. The return code in this message shows why the original request message was not valid:

00D5020F	It is not of type MQMT_REQUEST.
00D50210	It has zero length.
00D50212	It is longer than 32 762 bytes.
00D50211	It contains all blanks.
00D5483E	It needed converting, but *Format* was not MQFMT_STRING.
Other	See Command server codes

The command server reply message descriptor

For any reply message, the following MQMD message descriptor fields are set:

MsgType	MQMT_REPLY
Feedback	MQFB_NONE
Encoding	MQENC_NATIVE
Priority	As for the MQMD in the message you issued.
Persistence	As for the MQMD in the message you issued.
CorrelId	Depends on the MQPUT report options.
ReplyToQ	None.

The command server sets the *Options* field of the MQPMO structure to MQPMO_NO_SYNCPOINT. This means that you can retrieve the replies as they are created, rather than as a group at the next syncpoint.

z/OS Interpreting the reply messages from the command server

Each request message correctly processed by IBM MQ produces at least two reply messages. Each reply message contains a single IBM MQ user message.

The length of a reply depends on the command that was issued. The longest reply you can get is from a **DISPLAY NAMELIST** command, and that can be up to 15 000 bytes in length.

The first user message, CSQN205I, always contains:

- A count of the replies (in decimal), which you can use as a counter in a loop to get the rest of the replies. The count includes this first message.
- The return code from the command preprocessor.
- A reason code, which is the reason code from the command processor.

This message does not contain a CPF.

For example:

```
CSQN205I    COUNT=    4, RETURN=0000000C, REASON=00000008
```

The COUNT field is 8 bytes long and is right-justified. It always starts at position 18, that is, immediately after COUNT=. The RETURN field is 8 bytes long in character hexadecimal and is immediately after RETURN= at position 35. The REASON field is 8 bytes long in character hexadecimal and is immediately after REASON= at position 52.

If the RETURN= value is 00000000 and the REASON= value is 00000004, the set of reply messages is incomplete. After retrieving the replies indicated by the CSQN205I message, issue a further MQGET call to wait for a further set of replies. The first message in the next set of replies is again CSQN205I, indicating how many replies there are, and whether there are still more to come.

See the IBM MQ for z/OS mensajes, finalización, y códigos de razón documentation for more details about the individual messages.

If you are using a non-English language feature, the text and layout of the replies are different from those shown here. However, the size and position of the count and return codes in message CSQN205I are the same.

z/OS If you do not receive a reply

There are a series of steps you can take if you do not receive a response to request to the command server.

If you do not receive a reply to your request message, work through this checklist:

- Is the command server running?
- Is the *WaitInterval* long enough?
- Are the system-command input and reply-to queues correctly defined?
- Were the MQOPEN calls to these queues successful?
- Are both the system-command input and reply-to queues enabled for MQPUT and MQGET calls?
- Have you considered increasing the MAXDEPTH and MAXMSGL attributes of your queues?
- Are you are using the *CorrelId* and *MsgId* fields correctly?
- Is the queue manager still running?
- Was the command built correctly?
- Are all your remote links defined and operating correctly?
- Were the MQPUT calls correctly defined?
- Has the reply-to queue been defined as a temporary dynamic queue instead of a permanent dynamic queue? (If the request message is persistent, you must use a permanent dynamic queue for the reply.)

When the command server generates replies but cannot write them to the reply-to queue that you specify, it writes them to the dead-letter queue.

z/OS Passing commands using MGCRE

With appropriate authorization, an application program can make requests to multiple queue managers using a z/OS service routine.

If you have the correct authorization, you can pass IBM MQ commands from your program to multiple queue managers by the MGCRE (SVC 34) z/OS service. See the *z/OS MVS Programming: Authorized Assembler Services Guide* for more information.

The value of the CPF identifies the particular queue manager to which the command is directed. For information about CPFs, see User IDs for command security and command resource security and “Issuing queue manager commands on z/OS” on page 461.

If you use MGCRE, you can use a Command and Response Token (CART) to get the direct responses to the command.

z/OS Examples of commands and their replies

Use this topic as a series of examples of commands to the command server and the responses from the command server.

Here are some examples of commands that could be built into IBM MQ messages, and the user messages that are the replies. Unless otherwise stated, each line of the reply is a separate message.

- Messages from a DEFINE command
- Messages from a DELETE command
- Messages from DISPLAY commands
- Messages from commands with CMDSCOPE
- Messages from commands that generate commands with CMDSCOPE

Messages from a DEFINE command

The following command:

```
DEFINE QLOCAL(Q1)
```

produces these messages:

```
CSQN205I    COUNT=    2, RETURN=00000000, REASON=00000000
CSQ9022I +CSQ1 CSQMMSGP ' DEFINE QLOCAL' NORMAL COMPLETION
```

These reply messages are produced on normal completion.

Messages from a DELETE command

The following command:

```
DELETE QLOCAL(Q2)
```

produces these messages:

```
CSQN205I    COUNT=    4, RETURN=0000000C, REASON=00000008
CSQM125I +CSQ1 CSQMUQLC QLOCAL (Q2) QSGDISP(QMGR) WAS NOT FOUND
CSQM090E +CSQ1 CSQMUQLC FAILURE REASON CODE X'00D44002'
CSQ9023E +CSQ1 CSQMUQLC ' DELETE QLOCAL' ABNORMAL COMPLETION
```

These messages indicate that a local queue called Q2 does not exist.

Messages from DISPLAY commands

The following examples show the replies from some DISPLAY commands.

Finding out the name of the dead-letter queue

If you want to find out the name of the dead-letter queue for a queue manager, issue this command from an application program:

```
DISPLAY QMGR DEADQ
```

The following three user messages are returned, from which you can extract the required name:

```
CSQN205I    COUNT=    3, RETURN=00000000, REASON=00000000
CSQM409I +CSQ1 QMNAME(CSQ1) DEADQ(SYSTEM.DEAD.QUEUE          )
CSQ9022I +CSQ1 CSQMDRTS ' DISPLAY QMGR' NORMAL COMPLETION
```

Messages from the DISPLAY QUEUE command

The following examples show how the results from a command depend on the attributes specified in that command.

Example 1

You define a local queue using the command:

```
DEFINE QLOCAL(Q1) DESCR('A sample queue') GET(ENABLED) SHARE
```

If you issue the following command from an application program:

```
DISPLAY QUEUE(Q1) SHARE GET DESCR
```

these three user messages are returned:

```
CSQN205I    COUNT=    3, RETURN=00000000, REASON=00000000
CSQM401I +CSQ1 QUEUE(Q1                       ) TYPE(
QLOCAL ) QSGDISP(QMGR   )
DESCR(A sample queue
) SHARE  GET(ENABLED  )
CSQ9022I +CSQ1 CSQMDMSG ' DISPLAY QUEUE' NORMAL COMPLETION
```

Note: The second message, CSQM401I, is shown here occupying four lines.

Example 2

Two queues have names beginning with the letter A:

- A1 is a local queue with its PUT attribute set to DISABLED.
- A2 is a remote queue with its PUT attribute set to ENABLED.

If you issue the following command from an application program:

```
DISPLAY QUEUE(A*) PUT
```

these four user messages are returned:

```
CSQN205I    COUNT=    4, RETURN=00000000, REASON=00000000
CSQM401I +CSQ1 QUEUE(A1                       ) TYPE(
QLOCAL ) QSGDISP(QMGR   )
PUT(DISABLED )
CSQM406I +CSQ1 QUEUE(A2                       ) TYPE(
QREMOTE ) PUT(ENABLED  )
CSQ9022I +CSQ1 CSQMDMSG ' DISPLAY QUEUE' NORMAL COMPLETION
```

Note: The second and third messages, CSQM401I and CSQM406I, are shown here occupying three and two lines.

Messages from the DISPLAY NAMELIST command

You define a namelist using the command:

```
DEFINE NAMELIST(N1) NAMES(Q1,SAMPLE_QUEUE)
```

If you issue the following command from an application program:

```
DISPLAY NAMELIST(N1) NAMES NAMCOUNT
```

the following three user messages are returned:

```
CSQN205I    COUNT=    3, RETURN=00000000, REASON=00000000
CSQM407I +CSQ1 NAMELIST(N1                      ) QS
GDISP(QMGR   ) NAMCOUNT(     2) NAMES(Q1
,SAMPLE_QUEUE                 )
CSQ9022I +CSQ1 CSQMDMSG ' DISPLAY NAMELIST' NORMAL COMPLETION
```

Note: The second message, CSQM407I, is shown here occupying three lines.

Messages from commands with CMDSCOPE

The following examples show the replies from commands that have been entered with the CMDSCOPE attribute.

Messages from the ALTER PROCESS command

The following command:

```
ALT PRO(V4) CMDSCOPE(*)
```

produces the following messages:

```
CSQN205I  COUNT=    2, RETURN=00000000, REASON=00000004
CSQN137I !MQ25 'ALT PRO' command accepted for CMDSCOPE(*), sent to 2
CSQN205I  COUNT=    5, RETURN=00000000, REASON=00000004
CSQN121I !MQ25 'ALT PRO' command responses from MQ26
CSQM125I !MQ26 CSQMMSGP PROCESS(V4) QSGDISP(QMGR) WAS NOT FOUND
CSQM090E !MQ26 CSQMMSGP FAILURE REASON CODE X'00D44002'
CSQ9023E !MQ26 CSQMMSGP ' ALT PRO' ABNORMAL COMPLETION
CSQN205I  COUNT=    3, RETURN=00000000, REASON=00000004
CSQN121I !MQ25 'ALT PRO' command responses from MQ25
CSQ9022I !MQ25 CSQMMSGP ' ALT PRO' NORMAL COMPLETION
CSQN205I  COUNT=    2, RETURN=0000000C, REASON=00000008
CSQN123E !MQ25 'ALT PRO' command for CMDSCOPE(*) abnormal completion
```

These messages tell you that the command was entered on queue manager MQ25 and sent to two queue managers (MQ25 and MQ26). The command was successful on MQ25 but the process definition did not exist on MQ26, so the command failed on that queue manager.

Messages from the DISPLAY PROCESS command

The following command:

```
DIS PRO(V*) CMDSCOPE(*)
```

produces the following messages:

```
CSQN205I  COUNT=    2, RETURN=00000000, REASON=00000004
CSQN137I !MQ25 'DIS PRO' command accepted for CMDSCOPE(*), sent to 2
CSQN205I  COUNT=    5, RETURN=00000000, REASON=00000004
CSQN121I !MQ25 'DIS PRO' command responses from MQ26
CSQM408I !MQ26 PROCESS(V2) QSGDISP(COPY)
CSQM408I !MQ26 PROCESS(V3) QSGDISP(QMGR)
CSQ9022I !MQ26 CSQMDRTS ' DIS PROCESS' NORMAL COMPLETION
CSQN205I  COUNT=    7, RETURN=00000000, REASON=00000004
CSQN121I !MQ25 'DIS PRO' command responses from MQ25
CSQM408I !MQ25 PROCESS(V2) QSGDISP(COPY)
CSQM408I !MQ25 PROCESS(V2) QSGDISP(GROUP)
CSQM408I !MQ25 PROCESS(V3) QSGDISP(QMGR)
CSQM408I !MQ25 PROCESS(V4) QSGDISP(QMGR)
CSQ9022I !MQ25 CSQMDRTS ' DIS PROCESS' NORMAL COMPLETION
CSQN205I  COUNT=    2, RETURN=00000000, REASON=00000000
CSQN122I !MQ25 'DIS PRO' command for CMDSCOPE(*) normal completion
```

These messages tell you that the command was entered on queue manager MQ25 and sent to two queue managers (MQ25 and MQ26). Information is displayed about all the processes on each queue manager with names starting with the letter V.

Messages from the DISPLAY CHSTATUS command

The following command:

```
DIS CHS(VT) CMDSCOPE(*)
```

produces the following messages:

```
CSQN205I  COUNT=    2, RETURN=00000000, REASON=00000004
CSQN137I !MQ25 'DIS CHS' command accepted for CMDSCOPE(*), sent to 2
CSQN205I  COUNT=    4, RETURN=00000000, REASON=00000004
CSQN121I !MQ25 'DIS CHS' command responses from MQ25
CSQM422I !MQ25 CHSTATUS(VT) CHLDISP(PRIVATE) CONNAME( ) CURRENT STATUS(STOPPED)
CSQ9022I !MQ25 CSQXDRTS ' DIS CHS' NORMAL COMPLETION
CSQN205I  COUNT=    4, RETURN=00000000, REASON=00000004
CSQN121I !MQ25 'DIS CHS' command responses from MQ26
CSQM422I !MQ26 CHSTATUS(VT) CHLDISP(PRIVATE) CONNAME( ) CURRENT STATUS(STOPPED)
CSQ9022I !MQ26 CSQXDRTS ' DIS CHS' NORMAL COMPLETION
CSQN205I  COUNT=    2, RETURN=00000000, REASON=00000000
CSQN122I !MQ25 'DIS CHS' command for CMDSCOPE(*) normal completion
```

These messages tell you that the command was entered on queue manager MQ25 and sent to two queue managers (MQ25 and MQ26). Information is displayed about channel status on each queue manager.

Messages from the STOP CHANNEL command

The following command:

```
STOP CHL(VT) CMDSCOPE(*)
```

produces these messages:

```
CSQN205I  COUNT=    2, RETURN=00000000, REASON=00000004
CSQN137I !MQ25 'STOP CHL' command accepted for CMDSCOPE(*), sent to 2
CSQN205I  COUNT=    3, RETURN=00000000, REASON=00000004
CSQN121I !MQ25 'STOP CHL' command responses from MQ25
CSQM134I !MQ25 CSQMTCHL STOP CHL(VT) COMMAND ACCEPTED
SQN205I  COUNT=    3, RETURN=00000000, REASON=00000004
CSQN121I !MQ25 'STOP CHL' command responses from MQ26
CSQM134I !MQ26 CSQMTCHL STOP CHL(VT) COMMAND ACCEPTED
CSQN205I  COUNT=    3, RETURN=00000000, REASON=00000004
CSQN121I !MQ25 'STOP CHL' command responses from MQ26
CSQ9022I !MQ26 CSQXCRPS ' STOP CHL' NORMAL COMPLETION
CSQN205I  COUNT=    3, RETURN=00000000, REASON=00000004
CSQN121I !MQ25 'STOP CHL' command responses from MQ25
CSQ9022I !MQ25 CSQXCRPS ' STOP CHL' NORMAL COMPLETION
CSQN205I  COUNT=    2, RETURN=00000000, REASON=00000000
CSQN122I !MQ25 'STOP CHL' command for CMDSCOPE(*) normal completion
```

These messages tell you that the command was entered on queue manager MQ25 and sent to two queue managers (MQ25 and MQ26). Channel VT was stopped on each queue manager.

Messages from commands that generate commands with CMDSCOPE

The following command:

```
DEF PRO(V2) QSGDISP(GROUP)
```

produces these messages:

```
CSQN205I  COUNT=    3, RETURN=00000000, REASON=00000004
CSQM122I !MQ25 CSQMMSGP ' DEF PRO' COMPLETED FOR QSGDISP(GROUP)
CSQN138I !MQ25 'DEFINE PRO' command generated for CMDSCOPE(*), sent to 2
CSQN205I  COUNT=    3, RETURN=00000000, REASON=00000004
CSQN121I !MQ25 'DEFINE PRO' command responses from MQ25
CSQ9022I !MQ25 CSQMMSGP ' DEFINE PROCESS' NORMAL COMPLETION
CSQN205I  COUNT=    3, RETURN=00000000, REASON=00000004
CSQN121I !MQ25 'DEFINE PRO' command responses from MQ26
CSQ9022I !MQ26 CSQMMSGP ' DEFINE PROCESS' NORMAL COMPLETION
CSQN205I  COUNT=    2, RETURN=00000000, REASON=00000000
CSQN122I !MQ25 'DEFINE PRO' command for CMDSCOPE(*) normal completion
```

These messages tell you that the command was entered on queue manager MQ25. When the object was created on the shared repository, another command was generated and sent to all the active queue managers in the queue sharing group (MQ25 and MQ26).

z/OS Managing IBM MQ resources on z/OS

Use the links in this topic to find out how to manage the resources used by IBM MQ for z/OS, for example, managing log files, data sets, page sets, buffer pools, and coupling facility structures.

Use the following links for details of the different administrative tasks you might have to complete while using IBM MQ for z/OS:

- “Managing the logs” on page 504
- “Managing the bootstrap data set (BSDS)” on page 513
- “Managing page sets” on page 520
- “How to back up and recover page sets” on page 527
- “How to back up and restore queues using CSQUTIL” on page 530
- “Managing buffer pools” on page 530

- "Managing queue sharing groups and shared queues on z/OS" on page 532

Related concepts

IBM MQ for z/OS concepts

"Administering IBM MQ for z/OS" on page 461
IBM MQ for z/OS can be controlled and managed by MQSC and PCF commands, by a set of utilities and programs provided with the product, and by authorized applications.

"Sources from which you can issue MQSC and PCF commands on IBM MQ for z/OS" on page 461
You can issue MQSC and PCF commands from the IBM MQ for z/OS console, the initialization input data sets, the batch utility CSQUTIL, or from authorized applications. Not all commands can be issued from all these sources.

"Recovery and restart on z/OS" on page 541
Use this topic to understand the recovery and restart mechanisms used by IBM MQ.

Related tasks

Planning your IBM MQ environment on z/OS

Configuring queue managers on z/OS

IBM MQ utilities on z/OS reference

Related reference

"Using the IBM MQ for z/OS utilities" on page 483
IBM MQ for z/OS provides a set of utility programs that you can use to help with system administration.

Programmable command formats reference

z/OS Managing the logs

Use this topic to understand how to manage your IBM MQ log files, including the log archiving process, using log record compression, log record recovery, and printing log records.

This topic describes the tasks involved in managing the IBM MQ logs. It contains these sections:

z/OS *Archiving logs with the ARCHIVE LOG command*

An authorized operator can archive the current IBM MQ active log data sets whenever required using the **ARCHIVE LOG** command.

When you issue the ARCHIVE LOG command, IBM MQ truncates the current active log data sets, then runs an asynchronous offload process, and updates the BSDS with a record of the offload process.

The **ARCHIVE LOG** command has a **MODE(QUIESCE)** option. With this option, IBM MQ jobs and users are quiesced after a commit point, and the resulting point of consistency is captured in the current active log before it is offloaded.

Consider using the **MODE(QUIESCE)** option when planning a backup strategy for off site recovery. It creates a system-wide point of consistency, which minimizes the number of data inconsistencies when the archive log is used with the most current backup page set copy during recovery. For example:

```
ARCHIVE LOG MODE(QUIESCE)
```

If you issue the **ARCHIVE LOG** command without specifying a **TIME** parameter, the quiesce time period defaults to the value of the **QUIESCE** parameter of the CSQ6ARVP macro. If the time required for the `ARCHIVE LOG MODE(QUIESCE)` to complete is less than the time specified, the command completes successfully; otherwise, the command fails when the time period expires. You can specify the time period explicitly by using the **TIME** option, for example:

```
ARCHIVE LOG MODE(QUIESCE) TIME(60)
```

This command specifies a quiesce period of up to 60 seconds before **ARCHIVE LOG** processing occurs.

Attention: Using the **TIME** option when time is critical can significantly disrupt IBM MQ availability for all jobs and users that use IBM MQ resources.

By default, the command is processed asynchronously from the time you submit the command. (To process the command synchronously with other IBM MQ commands use the `WAIT(YES)` option with **QUIESCE**, but be aware that the z/OS console is locked from IBM MQ command input for the entire **QUIESCE** period.)

During the quiesce period:

- Jobs and users on the queue manager are allowed to go through commit processing, but are suspended if they try to update any IBM MQ resource after the commit.
- Jobs and users that only read data can be affected, since they might be waiting for locks held by jobs or users that were suspended.
- New tasks can start, but they cannot update data.

The output from the **DISPLAY LOG** command uses the message CSQV400I to indicate that a quiesce is in effect.

For example:

```
CSQJ322I +CSQ1 DISPLAY LOG report ...
Parameter   Initial value          SET value
----------- ---------------------- ----------------------
INBUFF      60
OUTBUFF     400
MAXRTU      2
MAXARCH     2
TWOACTV     YES
TWOARCH     YES
TWOBSDS     YES
OFFLOAD     YES
MAXCNOFF    0
WRTHRSH     20
DEALLCT     0
COMPLOG     NONE
ZHYWRITE    NO
End of LOG report
CSQJ370I +CSQ1 LOG status report ...
Copy %Full zHyperWrite Encrypted DSName
 1     68  NO          NO        VICY.CSQ1.LOGCOPY1.DS01
 2     68  NO          NO        VICY.CSQ1.LOGCOPY2.DS01
Restarted at 2019-08-15 09:49:30 using RBA=000000000891B000
Latest RBA=000000000891CCF8
Offload task is AVAILABLE
Full logs to offload - 0 of 4
CSQV400I +CSQ1 ARCHIVE LOG QUIESCE CURRENTLY ACTIVE
CSQ9022I +CSQ1 CSQJC001 ' DISPLAY LOG' NORMAL COMPLETION
```

When all updates are quiesced, the quiesce history record in the BSDS is updated with the date and time that the active log data sets were truncated, and with the last-written RBA in the current active log data sets. IBM MQ truncates the current active log data sets, switches to the next available active log data sets, and issues message CSQJ311I stating that the offload process started.

If updates cannot be quiesced before the quiesce period expires, IBM MQ issues message CSQJ317I, and **ARCHIVE LOG** processing terminates. The current active log data sets are not truncated, nor switched to the next available log data sets, and the offload process is not started.

Whether the quiesce was successful or not, all suspended users and jobs are then resumed, and IBM MQ issues message CSQJ312I, stating that the quiesce is ended and update activity is resumed.

If **ARCHIVE LOG** is issued when the current active log is the last available active log data set, the command is not processed, and IBM MQ issues the following message:

```
CSQJ319I - csect-name CURRENT ACTIVE LOG DATA SET IS THE LAST
AVAILABLE ACTIVE LOG DATA SET. ARCHIVE LOG PROCESSING
WILL BE TERMINATED
```

If **ARCHIVE LOG** is issued when another **ARCHIVE LOG** command is already in progress, the new command is not processed, and IBM MQ issues the following message:

```
CSQJ318I - ARCHIVE LOG COMMAND ALREADY IN PROGRESS
```

For information about the messages issued during archiving, see Messages for IBM MQ for z/OS.

Restarting the log archive process after a failure

If there is a problem during the log archive process (for example, a problem with allocation or tape mounts), the archiving of the active log might be suspended. You can cancel the archive process and restart it by using the following command:

```
ARCHIVE LOG CANCEL OFFLOAD
```

This command cancels any offload processing currently in progress, and restarts the archive process. It starts with the oldest log data set that has not been archived, and proceeds through all active log data sets that need offloading. Any log archive operations that have been suspended are restarted.

Use this command only if you are sure that the current log archive task is no longer functioning, or if you want to restart a previous attempt that failed. This is because the command might cause an abnormal termination of the offload task, which might result in a dump.

z/OS *Controlling archiving and logging*

You can control compression, printing, archiving, recovery and logging with using the CSQ6LOGP, CSQ6ARVP, and CSQ6SYSP macros. Note, that changes to private objects only are logged in IBM MQlogs. Changes to GROUP objects (like shared inbound channels) are also logged, because the definitions are propagated around the group and held locally.

Many aspects of archiving and logging are controlled by parameters set using the CSQ6LOGP, CSQ6ARVP and CSQ6SYSP macros of the system parameter module when the queue manager is customized. See Tailor your system parameter module for details of these macros.

Some of these parameters can be changed while a queue manager is running using the IBM MQ MQSC SET LOG, SET SYSTEM and SET ARCHIVE commands. They are shown in Table 28 on page 506:

Table 28. Archiving and logging parameters that can be changed while a queue manager is running

SET command	**Parameters**
LOG	WRTHRSH, MAXARCH, DEALLCT, MAXRTU, COMPLOG
ARCHIVE	All
SYSTEM	LOGLOAD

You can display the settings of all the parameters using the MQSC DISPLAY LOG, DISPLAY ARCHIVE and DISPLAY SYSTEM commands. These commands also show status information about archiving and logging.

Controlling log compression

You can enable and disable the compression of log records using either

- The SET and DISPLAY LOG commands in MQSC; see The MQSC commands
- Invoking PCF interface. See “Introducción a los Formatos de mandato programable de IBM MQ” on page 26
- Using the CSQ6LOGP macro in the system parameter module; see Using CSQ6LOGP

Printing log records

You can extract and print log records using the CSQ1LOGP utility. For instructions, see The log print utility.

Recovering logs

Normally, you do not need to back up and restore the IBM MQ logs, especially if you are using dual logging. However, in rare circumstances, such as an I/O error on a log, you might need to recover the logs. Use Access Method Services to delete and redefine the data set, and then copy the corresponding dual log into it.

z/OS *Discarding archive log data sets*

You can discard your archive log data sets and choose to discard the logs automatically or manually.

You must keep enough log data to be able to perform unit of work recovery, page set media recovery if a page set is lost, or CF structure media recovery if a CF structure is lost. Do not discard archive log data sets that might be required for recovery; if you discard these archive log data sets you might not be able to perform required recovery operations.

If you have confirmed that your archive log data sets can be discarded, you can do this in either of the following ways:

- Automatic archive log data set deletion
- Manually deleting archive log data sets

Automatic archive log data set deletion

You can use a DASD or tape management system to delete archive log data sets automatically. The retention period for IBM MQ archive log data sets is specified by the retention period field ARCRETN in the CSQ6ARVP installation macro (see the Using CSQ6ARVP for more information).

The default for the retention period specifies that archive logs are to be kept for 9999 days (the maximum).

Important: You can change the retention period but you must ensure that you can accommodate the number of backup cycles that you have planned for.

.

IBM MQ uses the retention period value as the value for the JCL parameter RETPD when archive log data sets are created.

The retention period set by the MVS™/DFP storage management subsystem (SMS) can be overridden by this IBM MQ parameter. Typically, the retention period is set to the smaller value specified by either IBM MQ or SMS. The storage administrator and IBM MQ administrator must agree on a retention period value that is appropriate for IBM MQ.

Note: IBM MQ does not have an automated method to delete information about archive log data sets from the BSDS, because some tape management systems provide external manual overrides of retention periods. Therefore, information about an archive log data set can still be in the BSDS long after the data set retention period has expired and the data set has been scratched by the tape management system. Conversely, the maximum number of archive log data sets might have been exceeded and the data from the BSDS might have been dropped before the data set has reached its expiration date.

If archive log data sets are deleted automatically, remember that the operation does not update the list of archive logs in the BSDS. You can update the BSDS with the change log inventory utility, as described in “Changing the BSDS” on page 514. The update is not essential. Recording old archive logs wastes space in the BSDS, but does no other harm.

Manually deleting archive log data sets

You must keep all the log records as far back as the lowest RBA identified in messages CSQI024I and CSQI025I. This RBA is obtained using the DISPLAY USAGE command that you issued when creating a point of recovery using Method 1: Full backup.

Read Creating a point of recovery for non-shared resources before discarding any logs.

Locate and discard archive log data sets

Having established the minimum log RBA required for recovery, you can find archive log data sets that contain only earlier log records by performing the following procedure:

1. Use the print log map utility to print the contents of the BSDS. For an example of the output, see The print log map utility.
2. Find the sections of the output titled `ARCHIVE LOG COPY n DATA SETS`. If you use dual logging, there are two sections. The columns labeled STARTRBA and ENDRBA show the range of RBAs contained in each volume. Find the volumes with ranges that include the minimum RBA you found with messages CSQI024I and CSQI025I. These are the earliest volumes you need to keep. If you are using dual-logging, there are two such volumes.

   If no volumes have an appropriate range, one of the following cases applies:

   - The minimum RBA has not yet been archived, and you can discard all archive log volumes.
   - The list of archive log volumes in the BSDS wrapped around when the number of volumes exceeded the number allowed by the MAXARCH parameter of the CSQ6LOGP macro. If the BSDS does not register an archive log volume, that volume cannot be used for recovery. Therefore, consider adding information about existing volumes to the BSDS. For instructions, see “Changes for archive logs” on page 516.

     Also consider increasing the value of MAXARCH. For information, see the Using CSQ6LOGP.
3. Delete any archive log data set or volume with an ENDRBA value that is less than the STARTRBA value of the earliest volume you want to keep. If you are using dual logging, delete both such copies.

   Because BSDS entries wrap around, the first few entries in the BSDS archive log section might be more recent than the entries at the end. Look at the combination of date and time and compare their ages. Do not assume that you can discard all entries before the entry for the archive log containing the minimum LOGRBA.

   Delete the data sets. If the archives are on tape, erase the tapes. If they are on DASD, run a z/OS utility to delete each data set. Then, if you want the BSDS to list only existing archive volumes, use the change log inventory utility (CSQJU003) to delete entries for the discarded volumes. See “Changes for archive logs” on page 516 for an example.

z/OS *The effect of log shunting*

Long running transactions can cause unit of work log records which span log data sets. IBM MQ handles this scenario by using log shunting, a technique which moves the log records to optimize the quantity of log data retained, and queue manager restart time.

When a unit of work is considered to be long, a representation of each log record is written further down the log. This is known as *log shunting*. It is described more fully in Log files.

The queue manager uses these shunted log records instead of the originals after a failure, to ensure unit of work integrity. There are two benefits to this:

- the quantity of log data which must be retained for unit of work coordination is reduced
- less log data must be traversed at queue manager restart time, so the queue manager is restarted more quickly

Shunted log records do not contain sufficient information for media recovery operations.

Data held in the log is used for two distinct purposes; media recovery and unit of work coordination. If a media failure occurs which affects either a CF structure or page set, the queue manager can recover the media to the point of failure by restoring a prior copy and updating this using data contained in the log.

Persistent activity performed in a unit of work is recorded on the log so that in the event of a failure, it can either be backed out or locks can be recovered on changed resources. The quantity of log data you need to retain to enable queue manager recovery is affected by these two elements.

For media recovery, you must retain sufficient log data to be able to perform media recovery from at least the most recent media copy and to be able to back out. (Your site may stipulate the ability to recover from older backups.) For unit of work integrity, you must retain the log data for your oldest in flight or indoubt units of work.

To assist you with managing the system, the queue manager detects old units of work at each log archive and reports them in messages CSQJ160 and CSQJ161. An internal task reads unit of work log information for these old units of work and rewrites it in a more succinct form to the current position in the log. Message CSQR026 indicates when this has happened. The MQSC command DISPLAY USAGE TYPE(DATASET) can also help you to manage the retention of log data. The command reports the following three pieces of recovery information:

1. How much of the log must be retained for unit of work recovery.
2. How much of the log must be retained for media recovery of page sets.
3. For a queue manager in a queue sharing group, how much of the log must be retained for media recovery of CF structures.

For each of these pieces of information, an attempt is made to map the oldest log data required into a data set. As new units of work start and stop, (1) would be expected to move to a more recent position in the log. If it is not moving, the long running UOW messages warn you that there is an issue. (2) relates to page set media recovery if the queue manager were to be shut down now and restarted. It does not know about when you last backed up your page sets, or which backup you might have to use if there was a page set failure. It normally moves to a more recent position in the log during checkpoint processing as changes held in the buffer pools are written to the page sets. In (3), the queue manager does know about CF structure backups taken either on this queue manager or on other queue managers in the queue sharing group. However, CF structure recovery requires a merge of log data from all queue managers in the queue sharing group which have interacted with the CF structure since the last backup. This means that the log data is identified by a log record sequence number, (or LRSN), which is timestamp based and so applicable across the entire queue sharing group rather than an RBA which would be different on different queue managers in the queue sharing group. It normally moves to a more recent position in the log as BACKUP CFSTRUCT commands are performed on either this or other queue managers in the queue sharing group.

z/OS *Resetting the queue manager's log*

Use this topic to understand how to reset the queue manager's log.

You must not allow the queue manager log RBA to wrap around from the end of the log RBA range to 0, as this leads to a queue manager outage and all persistent data will become unrecoverable. The end of the log RBA is either a value of FFFFFFFFFFFF (if 6-byte RBAs as in use), or FFFFFFFFFFFFFFFF (if 8-byte RBAs are in use).

The queue manager issues messages CSQI045I, CSQI046E, CSQI047E, CSQJ031D, and CSQJ032E to indicate that the used log range is significant and that you should plan to take action to avoid an unplanned outage.

The queue manager terminates with reason code 00D10257 when the RBA value reaches FFF800000000 (if 6-byte log RBAs are in use) or FFFFFFC000000000 (if 8-byte log RBAs are in use).

If 6-byte log RBAs are in use, consider converting the queue manager to use 8-byte log RBAs rather than resetting the queue manager's log, following the process described in “Implementing the larger log Relative Byte Address” on page 512. Converting a queue manager to use 8-byte log RBAs requires a shorter outage than resetting the log, and increases the period of time before you have to reset the log.

Message CSQJ034I, issued during queue manager initialization, indicates the end of the log RBA range for the queue manager as configured, and can be used to determine whether 6-byte or 8-byte log RBAs are in use.

The procedure to follow to reset the queue manager's log is as follows:

1. Resolve any unresolved units of work. The number of unresolved units of work is displayed at queue manager startup in message CSQR005I as the INDOUBT count. At each checkpoint, and at queue manager shutdown, the queue manager automatically issues the command

   DISPLAY CONN(*) TYPE(CONN) ALL WHERE(UOWSTATE EQ UNRESOLVED) to provide information about unresolved units of work.

   See How in-doubt units of recovery are resolved for information on resolving units of recovery. The ultimate recourse is to use the **RESOLVE INDOUBT** MQSC command to manually resolve indoubt units of recovery.
2. Shut down the queue manager cleanly.

   You can use either **STOP QMGR** or **STOP QMGR MODE(FORCE)** as both these commands flush any changed pages from bufferpools to the page sets.
3. If a queue manager is part of a queue sharing group, take CFSTRUCT backups on other queue managers for all structures in the queue sharing group. This ensures that the most recent backups are not in this queue manager's log, and that this queue manager's log is not required for CFSTRUCT recovery.
4. Define new logs and BSDS using CSQJU003 (see The change log inventory utility for more information on using the change log inventory utility).
5. Run **CSQUTIL RESETPAGE** against all the page sets for this queue manager (see Copying a page and resetting the log for more information on using this function). Note that page set RBAs can be reset independently, so multiple concurrent jobs (for example, one per page set) can be submitted to reduce the elapsed time for this step.
6. Restart the queue manager

Warning messages

When IBM MQ detects that the end of the log is approaching, it issues console messages in the following order, which indicate that a log reset should be planned. In this section the messages show 6-byte log RBA values. The same console messages are issued when IBM MQ is running in 8-byte log RBA mode but with different values; see “Warning thresholds” on page 511 for the 8-byte log RBA thresholds.

1. When IBM MQ detects that the end of the log is approaching in the near future, (approximately 94% full) IBM MQ issues console message CSQI045I, as in the following example:

   ```
   CSQI045I -CSQ7 CSQILCUR Log RBA has reached 0000F00000000000.
   Plan a log reset
   ```
2. IBM MQ issues the following CSQI046E error console message when the end of the log is near (approximately 97% full). This informs the IBM MQ administrator to take action soon.

   ```
   CSQI046E -CSQ7 CSQILCUR Log RBA has reached 0000F80000000000.
   Perform a log reset
   ```
3. After the CSQI046E message is issued, at the next log switch, IBM MQ issues the following CSQJ032E console message with the word WARNING:

   ```
   CSQJ032E -CSQ7 CSQJW307 WARNING - APPROACHING END OF
   THE LOG RBA RANGE OF 0000FFFFFFFFFFFF. CURRENT LOG RBA IS 0000F80000022000.
   ```
4. After the CSQI046E and CSQJ032E console messages are issued, IBM MQ issues one more error message, which does not require immediate IBM MQ administrator intervention. IBM MQ issues console message CSQI047E (when the log is approximately 99% full):

   ```
   CSQI047E -CSQ7 CSQILCUR Log RBA has reached 0000FF0000000000.
   Stop queue manager and reset logs
   ```

5. When the log RBA reaches FF8000000000, IBM MQ increases the urgency of the situation and issues console message CSQJ032E with the word CRITICAL:

```
CSQJ032E -CSQ7 CSQJW009 CRITICAL - APPROACHING END OF THE LOG RBA RANGE OF 0000FFFFFFFFFFFF.
CURRENT LOG RBA IS 0000FFF7FFFFDFFF.
```

6. If the queue manager is started when the log RBA is almost at the maximum, the following CSQJ031D console message is issued. This stage requires the input of the IBM MQ administrator:.

```
CSQJ031D -CSQ7 CSQYSTRT THE LOG RBA RANGE MUST BE RESET.
REPLY 'Y' TO CONTINUE STARTUP OR 'N' TO SHUTDOWN
```

7. IBM MQ startup remains suspended until a reply is given to message CSQJ031D.

The purpose of these messages is to give the IBM MQ administrator time to plan for a system outage to reset the logs. In an ideal configuration, there are at least two queue managers, possibly in a queue sharing group (QSG), sharing the workload. When one is down for maintenance the other can continue to receive work.

The severity of console messages that IBM MQ issues becomes greater as the RBA gets closer to the end. Ideally your IBM MQ administrator should plan to reset the log RBA when the first console message is seen.

If the warning and error console messages are ignored, IBM MQ terminates with reason code 5C6-00D10257 when the log RBA reaches FFF800000000, at which point IBM MQ determines that the available range is too small for the queue manager to continue. When this point is reached, the only option is to take an outage and either reset the log or extend the size of the log RBA.

Note: When the end of the log is reached it is not possible to resolve any in-flight units of work (UOW); these are lost during the log reset process. Enough of the RBA range should be left to start the queue manager and resolve any UOW. Because IBM MQ issues console messages several times to inform that the end of the log is approaching, a log reset should be planned.

The preferred option to avoid losing any in-flight UOW is to extend the log RBA to use 8 bytes. This means that a log RBA reset will not be necessary for a long period.

Warning thresholds

The following table lists the thresholds, based on the length of the log RBA.

Console message	6-byte log RBA	8-byte log RBA
CSQI045I	0000F00000000000	FFFF800000000000
CSQI046E	0000F80000000000	FFFFC00000000000
CSQI047E	0000FF8000000000	FFFFFC0000000000
CSQJ032E	0000FF8000000000 0000FF8000000000	FFFFFC0000000000 FFFFFC0000000000
CSQJ031D	0000FF8000000000	FFFFFC0000000000

Notes:

1. For message CSQJ032E, the first number applies to the WARNING text and the second number applies to the CRITICAL text in the console message.
2. Message CSQJ031D is issued at IBM MQ initialization only.

Related concepts

“Implementing the larger log Relative Byte Address” on page 512

Before IBM MQ for z/OS 8.0, IBM MQ for z/OS used a 6 byte log RBA to identify the location of data within the log. From IBM MQ for z/OS 8.0, the log RBA can be 8 bytes long, increasing the period of time before you have to reset the log.

z/OS *Implementing the larger log Relative Byte Address*

Before IBM MQ for z/OS 8.0, IBM MQ for z/OS used a 6 byte log RBA to identify the location of data within the log. From IBM MQ for z/OS 8.0, the log RBA can be 8 bytes long, increasing the period of time before you have to reset the log.

Attention: You only have to carry out the following procedure to enable this feature if your queue managers were created before IBM MQ 9.3.0, as queue managers created at IBM MQ 9.3.0 and later already have this feature enabled.

See Planning to increase the maximum addressable log range for considerations when planning to enable 8 byte log RBA.

Perform these instructions, in the order shown, to enable 8 byte log RBA on a single IBM MQ for z/OS queue manager. For queue managers in a queue sharing group, perform the steps on each queue manager in turn.

1. Allocate new BSDS data sets with similar attributes to the current BSDS. You can tailor sample CSQ4BSDS and delete any irrelevant statement, or you can use your existing JCL, but change the BSDS name to something like `++HLQ++.NEW.BSDS01`.

   Notes:

   a. Check the attributes of your new BSDS before submitting the job to allocate the new BSDS. The only attribute that might change is the size of the BSDS.

   b. The new BSDS contains more data that the current BSDS, therefore, you must ensure that the new data sets are allocated with sufficient available space. The sample JCL in thlqual.SCSQPROC(CSQ4BSDS) contains the recommended values when defining a new BSDS.

2. Shut down the queue manager cleanly.
3. Run the BSDS conversion utility (CSQJUCNV) to convert the existing BSDS to the new BSDS data sets. This usually takes a few seconds to run.

   Your existing BSDS will not be changed during this process, and you can use that for the initialization of the queue manager in the case of an unsuccessful conversion.

4. Rename the current BSDS to become the old BSDS, and the new BSDS to become the current BSDS, so that the new data sets are used when you next restart the queue manager. You can use the DFSMS Access Method Services ALTER command, for example:

   ```
   ALTER '++HLQ++.BSDS01' NEWNAME('++HLQ++.OLD.BSDS01')
   ALTER '++HLQ++.NEW.BSDS01' NEWNAME('++HLQ++.BSDS01')
   ```

   Ensure that you also issue commands to rename both the data and index portions of the VSAM cluster.

5. Restart the queue manager. It should start in the same amount of time as it would have done when using 6 byte log RBA.

   If the queue manager does not restart successfully due to a failure to access the converted BSDS, attempt to identify the cause of the failure, resolve the problem and retry the operation. If required, contact your IBM support center for assistance.

   If necessary, the change can be backed out at this point by:

   a. Renaming the current BSDS to become the new BSDS.

   b. Renaming the old BSDS to become the current BSDS.

   c. Restarting the queue manager.

6. Once the queue manager has been successfully restarted with the converted BSDS, do not attempt to start the queue manager using the old BSDS.
7. Message CSQJ034I is issued during queue manager initialization to indicate the end of the log RBA for the queue manager as configured. Confirm that the end of the log RBA range displayed is FFFFFFFFFFFFFFFF. This indicates that 8 byte log RBA is in use.

Related tasks

Planning to increase the maximum addressable log range

Related reference

Larger log Relative Byte Address

The BSDS conversion utility (CSQJUCNV)

z/OS Managing the bootstrap data set (BSDS)

The bootstrap data set (BSDS) is used to reference log data sets, and log records. Use this topic to understand how you can examine, change, and recover the BSDS.

For more information, see The bootstrap data set.

This topic describes the tasks involved in managing the bootstrap data set. It contains these sections:

- “Finding out what the BSDS contains” on page 513
- “Changing the BSDS” on page 514
- “Recovering the BSDS” on page 518

z/OS *Finding out what the BSDS contains*

You can use the print log map utility (CSQJU004) to examine the contents of the BSDS.

The print log map utility (CSQJU004) is a batch utility that lists the information stored in the BSDS. For instructions on running it, see The print log map utility.

The BSDS contains:

- Time stamps
- Active log data set status

Time stamps in the BSDS

The output of the print log map utility shows the time stamps, which are used to record the date and time of various system events, that are stored in the BSDS.

The following time stamps are included in the header section of the report:

SYSTEM TIMESTAMP
Reflects the date and time the BSDS was last updated. The BSDS time stamp can be updated when:

- The queue manager starts.
- The write threshold is reached during log write activities. Depending on the number of output buffers you have specified and the system activity rate, the BSDS might be updated several times a second, or might not be updated for several seconds, minutes, or even hours. For details of the write threshold, see the WRTHRSH parameter of the CSQ6LOGP macro in Using CSQ6LOGP.
- IBM MQ drops into a single BSDS mode from its normal dual BSDS mode due to an error. This can occur when a request to get, insert, point to, update, or delete a BSDS record is unsuccessful. When this error occurs, IBM MQ updates the time stamp in the remaining BSDS to force a time stamp mismatch with the disabled BSDS.

UTILITY TIMESTAMP
The date and time the contents of the BSDS were altered by the change log inventory utility (CSQJU003).

The following time stamps are included in the active and archive log data sets portion of the report:

Active log date
The date the active log entry was created in the BSDS, that is, when the CSQJU003 NEWLOG was done.

Active log time
The time the active log entry was created in the BSDS, that is, when the CSQJU003 NEWLOG was done.

Archive log date
The date the archive log entry was created in the BSDS, that is, when the CSQJU003 NEWLOG was done or the archive itself was done.

Archive log time
The time the archive log entry was created in the BSDS, that is, when the CSQJU003 NEWLOG was done or the archive itself was done.

Active log data set status

The BSDS records the status of an active log data set as one of the following:

NEW
The data set has been defined but never used by IBM MQ, or the log was truncated to a point before the data set was first used. In either case, the data set starting and ending RBA values are reset to zero.

REUSABLE
Either the data set has been defined but never used by IBM MQ, or the data set has been offloaded. In the print log map output, the start RBA value for the last REUSABLE data set is equal to the start RBA value of the last archive log data set.

NOT REUSABLE
The data set contains records that have not been offloaded.

STOPPED
The offload processor encountered an error while reading a record, and that record could not be obtained from the other copy of the active log.

TRUNCATED
Either:

- An I/O error occurred, and IBM MQ has stopped writing to this data set. The active log data set is offloaded, beginning with the starting RBA and continuing up to the last valid record segment in the truncated active log data set. The RBA of the last valid record segment is lower than the ending RBA of the active log data set. Logging is switched to the next available active log data set, and continues uninterrupted.

or

- An ARCHIVE LOG function has been called, which has truncated the active log.

The status appears in the output from the print log map utility.

z/OS *Changing the BSDS*

You do not have to take special steps to keep the BSDS updated with records of logging events because IBM MQ does that automatically.

However, you might want to change the BSDS if you do any of the following:

- Add more active log data sets.
- Copy active log data sets to newly allocated data sets, for example, when providing larger active log allocations.
- Move log data sets to other devices.
- Recover a damaged BSDS.
- Discard outdated archive log data sets.

You can change the BSDS by running the change log inventory utility (CSQJU003). Only run this utility when the queue manager is inactive, or you might get inconsistent results. The action of the utility is controlled by statements in the SYSIN data set. This section shows several examples. For complete instructions, see The change log inventory utility.

You can copy an active log data set only when the queue manager is inactive because IBM MQ allocates the active log data sets as exclusive (DISP=OLD) at queue manager startup.

z/OS *Changes for active logs*

Use this topic to understand how you can change the active logs using the BSDS.

You can add to, delete from, and record entries in the BSDS for active logs using the change log utility. Examples only are shown here; replace the data set names shown with the ones you want to use. For more details of the utility, see The change log inventory utility.

See these sections for more information:

- Adding record entries to the BSDS
- Deleting information about the active log data set from the BSDS
- Recording information about the log data set in the BSDS
- Increasing the size of the active log
- The use of CSQJUFMT

Adding record entries to the BSDS

If an active log has been flagged as "stopped", it is not reused for logging; however, it continues to be used for reading. Use the access method services to define new active log data sets, then use the change log inventory utility to register the new data sets in the BSDS. For example, use:

```
NEWLOG DSNAME=MQM111.LOGCOPY1.DS10,COPY1
NEWLOG DSNAME=MQM111.LOGCOPY2.DS10,COPY2
```

If you are copying the contents of an old active log data set to the new one, you can also give the RBA range and the starting and ending time stamps on the NEWLOG function.

Deleting information about the active log data set from the BSDS

To delete information about an active log data set from the BSDS, you could use:

```
DELETE DSNAME=MQM111.LOGCOPY1.DS99
DELETE DSNAME=MQM111.LOGCOPY2.DS99
```

Recording information about the log data set in the BSDS

To record information about an existing active log data set in the BSDS, use:

```
NEWLOG DSNAME=MQM111.LOGCOPY1.DS10,COPY2,STARTIME=19930212205198,
ENDTIME=19930412205200,STARTRBA=6400,ENDRBA=94FF
```

You might need to insert a record containing this type of information in the BSDS because:

- The entry for the data set has been deleted, but is needed again.
- You are copying the contents of one active log data set to another data set.
- You are recovering the BSDS from a backup copy.

Increasing the size of the active log

There are two methods of achieving this process.

1. When the queue manager is active:
   a. Define new larger log data sets using JCL.
   b. Add the new log data sets to the active queue manager using the MQSC DEFINE LOG command.
   c. Use the MQSC ARCHIVE LOG command to move the current active log, to be a new larger log.
   d. Wait for the archive of the smaller active log data set to complete.
   e. Shut down the queue manager, using the CSQJU003 utility to remove the old small active logs.
   f. Restart the queue manager.
2. When the queue manager is inactive:
   a. Stop the queue manager. This step is required because IBM MQ allocates all active log data sets for its exclusive use when it is active.
   b. Use Access Method Services ALTER with the NEWNAME option to rename your active log data sets.
   c. Use Access Method Services DEFINE to define larger active log data sets.

      By reusing the old data set names, you do not have to run the change log inventory utility to establish new names in the BSDSs. The old data set names and the correct RBA ranges are already in the BSDSs.
   d. Use Access Method Services REPRO to copy the old (renamed) data sets into their appropriate new data sets.

      Note: This step can take a long time, so your enterprise could be out of action for this period.
   e. Start the queue manager.

If all your log data sets are the same size, your system will be operationally more consistent and efficient. If the log data sets are not the same size, it is more difficult to track your system's logs, and so space can be wasted.

The use of CSQJUFMT

Do not run a CSQJUFMT format when increasing the size of an active log.

If you run CSQJUFMT (in order to provide a performance advantage the first time the queue manager writes to the new active log) you receive messages:

```
IEC070I 203-204,XS95GTLX,REPRO02,OUTPUT,B857,SPMG02, 358
IEC070I MG.W.MG4E.LOGCOPY1.DS02,MG.W.MG4E.LOGCOPY1.DS02.DATA,
IDC3302I ACTION ERROR ON MG.W.MG4E.LOGCOPY1.DS02
IDC3351I ** VSAM I/O RETURN CODE IS 28 - RPLFDBWD = X'2908001C'
IDC31467I MAXIMUM ERROR LIMIT REACHED.

IDC0005I NUMBER OF RECORDS PROCESSED WAS 0
```

In addition, if you use the Access Method Services REPRO, ensure that you define a new empty log.

If you use REPRO to copy the old (renamed) data set into its respective new data set, the default is NOREPLACE.

This means that REPRO does not replace a record that is already on the designated data set. When formatting is done on the data set, the RBA value is reset. The net result is a data set that is not empty after formatting.

z/OS *Changes for archive logs*

Use this topic to understand how to change the archive logs.

You can add to, delete from, and change the password of, entries in the BSDS for archive logs. Examples only are shown here; replace the data set names shown with the ones you want to use. For more details of the utility, see The change log inventory utility.

- Adding an archive log
- Deleting an archive log
- Changing the password of an archive log

Adding an archive log

When the recovery of an object depends on reading an existing archive log data set, the BSDS must contain information about that data set so that IBM MQ can find it. To register information about an existing archive log data set in the BSDS, use:

```
NEWLOG DSNAME=CSQARC1.ARCHLOG1.E00021.T2205197.A0000015,COPY1VOL=CSQV04,
UNIT=TAPE,STARTRBA=3A190000,ENDRBA=3A1F0FFF,CATALOG=NO
```

Deleting an archive log

To delete an entire archive log data set on one or more volumes, use:

```
DELETE DSNAME=CSQARC1.ARCHLOG1.E00021.T2205197.A0000015,COPY1VOL=CSQV04
```

Changing the password of an archive log

If you change the password of an existing archive log data set, you must also change the information in the BSDS.

1. List the BSDS, using the print log map utility.
2. Delete the entry for the archive log data set with the changed password, using the DELETE function of the CSQJU003 utility (see topic The change log inventory utility).
3. Name the data set as for a new archive log data set. Use the NEWLOG function of the CSQJU003 utility (see topic The change log inventory utility), and give the new password, the starting and ending RBAs, and the volume serial numbers (which can be found in the print log map utility output, see The print log map utility).

To change the password for new archive log data sets, use:

```
ARCHIVE PASSWORD= password
```

To stop placing passwords on new archive log data sets, use:

```
ARCHIVE NOPASSWD
```

Note: Only use the ARCHIVE utility function if you do not have an external security manager.

z/OS *Changing the high-level qualifier (HLQ) for the logs and BSDS*

Use this topic to understand the procedure required to change the high-level qualifier (HLQ).

Before you begin

You must end the queue manager normally before copying any of the logs or data sets to the new data sets. This is to ensure that the data is consistent and no recovery is needed during restart.

About this task

This task provides information about how to change the HLQ for the logs and BSDS. To do this, follow these steps:

Procedure

1. Run the log print utility CSQJU004 to record the log data set information. This information is needed later.
2. You can either:
   a) run DSS backup and restore with rename on the log and BSDS data sets to be renamed, or
   b) use AMS DEFINE and REPRO to create the HLQ data sets and copy the data from the old data sets.
3. Modify the MSTR and CHIN procedures to point to the new data sets.
4. Delete the old log information in the new copy of the BSDS using CSQJU003.
5. Define the new log data sets to the new BSDS using the NEWLOG function of CSQJU003.
   Keep all information about each log the same, apart from the HLQ.
6. The new BSDS should reflect the same information that was recorded for the old logs in the old BSDS.
   The HLQ should be the only thing that has changed.

What to do next

Compare the CSQJU004 output for the old and new BSDS to ensure that they look EXACTLY the same (except for the HLQs) before starting the queue manager.

Note: Care must be taken when performing these operations. Incorrect actions might lead to unrecoverable situations. Check the PRINT LOG MAP UTILITY output and make sure that all the information needed for recovery or restart has been included.

z/OS *Recovering the BSDS*

If IBM MQ is operating in dual BSDS mode and one BSDS becomes damaged, forcing IBM MQ into single BSDS mode, IBM MQ continues to operate without a problem (until the next restart).

To return the environment to dual BSDS mode:

1. Use Access Method Services to rename or delete the damaged BSDS and to define a new BSDS with the same name as the damaged BSDS. Example control statements can be found in job CSQ4BREC in thlqual.SCSQPROC.
2. Issue the IBM MQ command RECOVER BSDS to make a copy of the valid BSDS in the newly allocated data set and to reinstate dual BSDS mode.

If IBM MQ is operating in single BSDS mode and the BSDS is damaged, or if IBM MQ is operating in dual BSDS mode and both BSDSs are damaged, the queue manager stops and does not restart until the BSDS data sets are repaired. In this case:

1. Locate the BSDS associated with the most recent archive log data set. The data set name of the most recent archive log appears on the job log in the last occurrence of message CSQJ003I, which indicates that offload processing has been completed successfully. In preparation for the rest of this procedure, it is a good practice to keep a log of all successful archives noted by that message:
   - If archive logs are on DASD, the BSDS is allocated on any available DASD. The BSDS name is like the corresponding archive log data set name; change only the first letter of the last qualifier, from A to B, as in this example:

Archive log name
CSQ.ARCHLOG1. ***A*** 0000001

BSDS copy name
CSQ.ARCHLOG1. ***B*** 0000001

- If archive logs are on tape, the BSDS is the first data set of the first archive log volume. The BSDS is not repeated on later volumes.

2. If the most recent archive log data set has no copy of the BSDS (for example, because an error occurred when offloading it), locate an earlier copy of the BSDS from earlier offload processing.
3. Rename *damaged* BSDSs using the Access Method Services ALTER command with the NEWNAME option. If you want to delete a damaged BSDS, use the Access Method Services DELETE command. For each damaged BSDS, use Access Method Services to define a new BSDS as a replacement data set. Job CSQ4BREC in thlqual.SCSQPROC contains Access Method Services control statements to define a new BSDS.
4. Use the Access Method Services REPRO command to copy the BSDS from the archive log to one of the replacement BSDSs you defined in step "3" on page 519. Do not copy any data to the second replacement BSDS, you do that in step "5" on page 520.

   a. Print the contents of the replacement BSDS.

      Use the print log map utility (CSQJU004) to print the contents of the replacement BSDS. This enables you to review the contents of the replacement BSDS before continuing your recovery work.

   b. Update the archive log data set inventory in the replacement BSDS.

      Examine the output from the print log map utility and check that the replacement BSDS does not contain a record of the archive log from which the BSDS was copied. If the replacement BSDS is an old copy, its inventory might not contain all archive log data sets that were created more recently. The BSDS inventory of the archive log data sets must be updated to reflect the current subsystem inventory.

      Use the change log inventory utility (CSQJU003) NEWLOG statement to update the replacement BSDS, adding a record of the archive log from which the BSDS was copied. If the archive log data set is password-protected, use the PASSWORD option of the NEWLOG function. Also, if the archive log data set is cataloged, ensure that the CATALOG option of the NEWLOG function is properly set to CATALOG=YES. Use the NEWLOG statement to add any additional archive log data sets that were created later than the BSDS copy.

   c. Update passwords in the replacement BSDS.

      The BSDS contains passwords for the archive log data sets and for the active log data sets. To ensure that the passwords in the replacement BSDS reflect the current passwords used by your installation, use the change log inventory ARCHIVE utility function with the PASSWORD option.

   d. Update the active log data set inventory in the replacement BSDS.

      In unusual circumstances, your installation might have added, deleted, or renamed active log data sets since the BSDS was copied. In this case, the replacement BSDS does not reflect the actual number or names of the active log data sets your installation currently has in use.

      If you need to delete an active log data set from the replacement BSDS log inventory, use the change log inventory utility DELETE function.

      If you need to add an active log data set to the replacement BSDS log inventory, use the change log inventory utility NEWLOG function. Ensure that the RBA range is specified correctly on the NEWLOG function. If the active log data set is password-protected, use the PASSWORD option.

      If you need to rename an active log data set in the replacement BSDS log inventory, use the change log inventory utility DELETE function, followed by the NEWLOG function. Ensure that the RBA range is specified correctly on the NEWLOG function. If the active log data set is password-protected, use the PASSWORD option.

   e. Update the active log RBA ranges in the replacement BSDS.

Later, when the queue manager restarts, it compares the RBAs of the active log data sets listed in the BSDS with the RBAs found in the actual active log data sets. If the RBAs do not agree, the queue manager does not restart. The problem is magnified when an old copy of the BSDS is used. To solve this problem, use the change log inventory utility (CSQJU003) to adjust the RBAs found in the BSDS using the RBAs in the actual active log data sets. You do this by:

- Using the print log records utility (CSQ1LOGP) to print a summary report of the active log data set. This shows the starting and ending RBAs.
- Comparing the actual RBA ranges with the RBA ranges you have just printed, when the RBAs of all active log data sets are known.

  If the RBA ranges are equal for all active log data sets, you can proceed to the next recovery step without any additional work.

  If the RBA ranges are not equal, adjust the values in the BSDS to reflect the actual values. For each active log data set that needs to have the RBA range adjusted, use the change log inventory utility DELETE function to delete the active log data set from the inventory in the replacement BSDS. Then use the NEWLOG function to redefine the active log data set to the BSDS. If the active log data sets are password-protected, use the PASSWORD option of the NEWLOG function.

f. If only two active log data sets are specified for each copy of the active log, IBM MQ can have difficulty during queue manager restart. The problem can arise when one of the active log data sets is full and has not been offloaded, while the second active log data set is close to filling. In this case, add a new active log data set for each copy of the active log and define each new active log data set in the replacement BSDS log inventory.

   Use the Access Method Services DEFINE command to define a new active log data set for each copy of the active log and use the change log inventory utility NEWLOG function to define the new active log data sets in the replacement BSDS. You do not need to specify the RBA ranges on the NEWLOG statement. However, if the active log data sets are password-protected, use the PASSWORD option of the NEWLOG function. Example control statements to accomplish this task can be found in job CSQ4LREC in thlqual.SCSQPROC.

5. Copy the updated BSDS to the second new BSDS data set. The BSDSs are now identical.

   Use the print log map utility (CSQJU004) to print the contents of the second replacement BSDS at this point.
6. See Active log problems for information about what to do if you have lost your current active log data set.
7. Restart the queue manager using the newly constructed BSDS. IBM MQ determines the current RBA and what active logs need to be archived.

z/OS Managing page sets

Use this topic to understand how to manage the page sets associated with a queue manager.

This topic describes how to add, copy, and generally manage the page sets associated with a queue manager. It contains these sections:

- "How to back up and recover page sets" on page 527
- "How to delete page sets" on page 530
- "How to back up and restore queues using CSQUTIL" on page 530

See Page sets for a description of page sets, storage classes, buffers, and buffer pools, and some of the performance considerations that apply.

How to change the high-level qualifier (HLQ) for the page sets

This task gives information on how to change the HLQ for the page sets. To perform this task, do the following:

1. Define the new HLQ page sets.
2. If the size allocation is the same as the old page sets, copy the existing page set using REPRO to the empty new HLQ page sets.
3. If you are increasing the size of the page sets, use the FORMAT function of CSQUTIL to format the destination pages, and then the COPYPAGE function of CSQUTIL to copy all the messages from the source page set to the destination page set.

   For more information, see Formatting page sets (FORMAT), and Expanding a page set (COPYPAGE).
4. Change the CSQP00xx DD statement in the queue manager procedure to point to the new HLQ page sets.

Restart the queue manager and verify the changes to the page sets.

How to add a page set to a queue manager

This description assumes that you have a queue manager that is already running. You might need to add a page set if, for example, your queue manager has to cope with new applications using new queues.

To add a new page set, use the following procedure:

1. Define and format the new page set. You can use the sample JCL in thlqual.SCSQPROC(CSQ4PAGE) as a basis. For more information, see Formatting page sets (FORMAT).

   Take care not to format any page sets that are in use, unless this is what you intend. If so, use the FORCE option of the FORMAT utility function.
2. Use the DEFINE PSID command with the DSN option to associate the page set with a buffer pool.
3. Add the appropriate storage class definitions for your page set by issuing DEFINE STGCLASS commands.
4. Optionally, to document how your queue manager is configured:
   a. Add the new page set to the started task procedure for your queue manager.
   b. Add a definition for the new page set to your CSQINP1 initialization data set.
   c. Add a definition for the new storage class to your CSQ4INYR initialization data set member.

For details of the DEFINE PSID and DEFINE STGCLASS commands, see DEFINE PSID and DEFINE STGCLASS.

What to do when one of your page sets becomes full

You can find out about the utilization of page sets by using the IBM MQ command DISPLAY USAGE. For example, the command:

```
DISPLAY USAGE PSID(03)
```

displays the current state of the page set 03. This tells you how many free pages this page set has.

If you have defined secondary extents for your page sets, they are dynamically expanded each time they fill up. Eventually, all secondary extents are used, or no further disk space is available. If this happens, an application receives the return code MQRC_STORAGE_MEDIUM_FULL.

If an application receives a return code of MQRC_STORAGE_MEDIUM_FULL from an MQI call, this is a clear indication that there is not enough space remaining on the page set. If the problem persists or is likely to recur, you must do something to solve it.

You can approach this problem in a number of ways:

- Balance the load between page sets by moving queues from one page set to another.
- Expand the page set. See "How to increase the size of a page set" on page 524 for instructions.
- Redefine the page set so that it can expand beyond 4 GB to a maximum size of 64 GB. See Defining a page set to be larger than 4 GB for instructions.

How to balance loads on page sets

Load balancing on page sets means moving the messages associated with one or more queues from one page set to another, less used, page set. Use this technique if it is not practical to expand the page set.

To identify which queues are using a page set, use the appropriate IBM MQ commands. For example, to find out which queues are mapped to page set 02, first, find out which storage classes map to page set 02, by using the command:

```
DISPLAY STGCLASS(*) PSID(02)
```

Then use the following command to find out which queues use which storage class:

```
DISPLAY QUEUE(*) TYPE(QLOCAL) STGCLASS
```

Moving a non-shared queue

To move queues and their messages from one page set to another, use the MQSC MOVE QLOCAL command (described in MOVE QLOCAL). When you have identified the queue or queues that you want to move to a new page set, follow this procedure for each of these queues:

1. Ensure that the queue you want to move is not in use by any applications (that is, IPPROCS and OPPROCS values from the DISPLAY QSTATUS command are zero) and that it has no uncommitted messages (the UNCOM value from the DISPLAY QSTATUS command is NO).

   Note: The only way to ensure that this state continues is to change the security authorization of the queue temporarily. See Profiles for queue security for more information.

   If you cannot do this, later stages in this procedure might fail if applications start to use the queue despite precautionary steps such as setting PUT(DISABLED). However, messages can never be lost by this procedure.
2. Prevent applications from putting messages on the queue being moved by altering the queue definition to disable MQPUT s. Change the queue definition to PUT(DISABLED).
3. Define a temporary queue with the same attributes as the queue that is being moved, using the command:

```
DEFINE QL(TEMP_QUEUE) LIKE(QUEUE_TO_MOVE) PUT(ENABLED) GET(ENABLED)
```

Note: If this temporary queue already exists from a previous run, delete it before doing the define.

4. Move the messages to the temporary queue using the following command:

```
MOVE QLOCAL(QUEUE_TO_MOVE) TOQLOCAL(TEMP_QUEUE)
```

5. Delete the queue you are moving, using the command:

```
DELETE QLOCAL(QUEUE_TO_MOVE)
```

6. Define a new storage class that maps to the required page set, for example:

```
DEFINE STGCLASS(NEW) PSID(nn)
```

Add the new storage class definition to the CSQINP2 data sets ready for the next queue manager restart.

7. Redefine the queue that you are moving, by changing the storage class attribute:

```
DEFINE QL(QUEUE_TO_MOVE) LIKE(TEMP_QUEUE) STGCLASS(NEW)
```

When the queue is redefined, it is based on the temporary queue created in step “3” on page 522.

8. Move the messages back to the new queue, using the command:

```
MOVE QLOCAL(TEMP) TOQLOCAL(QUEUE_TO_MOVE)
```

9. The queue created in step “3” on page 522 is no longer required. Use the following command to delete it:

```
DELETE QL(TEMP_QUEUE)
```

10. If the queue being moved was defined in the CSQINP2 data sets, change the STGCLASS attribute of the appropriate DEFINE QLOCAL command in the CSQINP2 data sets. Add the REPLACE keyword so that the existing queue definition is replaced.

Figure 29 on page 524 shows an extract from a load balancing job.

```
//UTILITY EXEC PGM=CSQUTIL,PARM=('CSQ1')
//STEPLIB DD  DSN=thlqual.SCSQANLE,DISP=SHR
//     DD  DSN=thlqual.SCSQAUTH,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//SYSIN  DD  *
COMMAND DDNAME(MOVEQ)
/*
//MOVEQ DD  *
ALTER QL(QUEUE_TO_MOVE) PUT(DISABLED)
DELETE QL(TEMP_QUEUE) PURGE
DEFINE QL(TEMP_QUEUE) LIKE(QUEUE_TO_MOVE) PUT(ENABLED) GET(ENABLED)
MOVE QLOCAL(QUEUE_TO_MOVE) TOQLOCAL(TEMP_QUEUE)
DELETE QL(QUEUE_TO_MOVE)
DEFINE STGCLASS(NEW) PSID(2)
DEFINE QL(QUEUE_TO_MOVE) LIKE(TEMP_QUEUE) STGCLASS(NEW)
MOVE QLOCAL(TEMP_QUEUE) TOQLOCAL(QUEUE_TO_MOVE)
DELETE QL(TEMP_QUEUE)
/*
```

Figure 29. Extract from a load balancing job for a page set

How to increase the size of a page set

You can initially allocate a page set larger than 4 GB, See Defining a page set to be larger than 4 GB

A page set can be defined to be automatically expanded as it becomes full by specifying EXPAND(SYSTEM) or EXPAND(USER). If your page set was defined with EXPAND(NONE), you can expand it in either of two ways:

- Alter its definition to allow automatic expansion. See Altering a page set to allow automatic expansion
- Create a new, larger page set and copy the messages from the old page set to the new one. See Moving messages to a new, larger page set

Defining a page set to be larger than 4 GB

IBM MQ can use a page set up to 64 GB in size, provided the data set is defined with 'extended addressability' to VSAM. Extended addressability is an attribute which is conferred by an SMS data class.

Note: Page sets and active log data sets are eligible to reside in the extended addressing space (EAS) part of an extended address volumes (EAV) and, from z/OS V1.12, an archive log dataset can also reside in the EAS.

In the example shown in the following sample JCL, the management class 'EXTENDED' is defined to SMS with 'Extended addressability'. If your existing page set is not currently defined as having extended addressability, use the following method to migrate to an extended addressability format data set.

1. Stop the queue manager.
2. Use Access Method Services to rename the existing page set.
3. Define a destination page set, the same size as the existing page set, but with DATACLAS(EXTENDED).

   Note: Extended-format data sets must be SMS managed. These are the mechanisms for requesting extended format for VSAM data sets:

   - Using a data class that has a DSNTYPE value of EXT and the subparameter R or P to indicate required or preferred.
   - Coding DSNTYPE=EXTREQ (extended format is required) or DSNTYPE=EXTPREF (extended format is preferred) on the DD statement.

- Coding the LIKE= parameter on the DD statement to refer to an existing extended format data set.

For more information, see Restrictions on Defining Extended-Format Data Sets.

4. Use the COPYPAGE function of CSQUTIL to copy all the messages from the source page set to the destination page set. See Expanding a page set (COPYPAGE) for more details.
5. Restart the queue manager.
6. Alter the page set to use system expansion, to allow it to continue growing beyond its current allocation.

The following JCL shows example Access Method Services commands:

```
//S1    EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=*
//SYSIN   DD  *
ALTER 'VICY.CSQ1.PAGE01' -
NEWNAME('VICY.CSQ1.PAGE01.OLD')
ALTER 'VICY.CSQ1.PAGE01.DATA' -
NEWNAME('VICY.CSQ1.PAGE01.DATA.OLD')
DEFINE CLUSTER (NAME('VICY.CSQ1.PAGE01') -
MODEL('VICY.CSQ1.PAGE01.OLD') -
DATACLAS(EXTENDED))
/*
```

Altering a page set to allow automatic expansion

Use the ALTER PSID command with the EXPAND(USER) or EXPAND(SYSTEM) options. See ALTER PSID and Expanding a page set (COPYPAGE) for general information on expanding page sets.

Moving messages to a new, larger page set

This technique involves stopping and restarting the queue manager. This deletes any nonpersistent messages that are not on shared queues at restart time. If you have nonpersistent messages that you do not want to be deleted, use load balancing instead. For more details, see "How to balance loads on page sets" on page 522. In this description, the page set that you want to expand is referred to as the *source* page set; the new, larger page set is referred to as the *destination* page set.

Follow these steps:

1. Stop the queue manager.
2. Define the destination page set, ensuring that it is larger than the source page set, with a larger secondary extent value.
3. Use the FORMAT function of CSQUTIL to format the destination page set. See Formatting page sets (FORMAT) for more details.
4. Use the COPYPAGE function of CSQUTIL to copy all the messages from the source page set to the destination page set. See Expanding a page set (COPYPAGE) for more details.
5. Restart the queue manager using the destination page set by doing one of the following:
   - Change the queue manager started task procedure to reference the destination page set.
   - Use Access Method Services to delete the source page set and then rename the destination page set, giving it the same name as that of the source page set.

Attention:

Before you delete any IBM MQ page set, be sure that you have made the required backup copies.

How to reduce a page set

Prevent all users, other than the IBM MQ administrator, from using the queue manager. For example; by changing the access security settings.

If you have a large page set that is mostly empty (as shown by the DISPLAY USAGE command), you might want to reduce its size. The procedure to do this involves using the COPY, FORMAT, and LOAD functions of CSQUTIL (see IBM MQ utility program). This procedure does not work for page set zero (0), as it is not practical to reduce the size of this page set; the only way to do so is by reinitializing your queue manager (see “Reinitializing a queue manager” on page 548). The prerequisite of this procedure is to try and remove all users from the system so that all UOWs are complete and the page sets are consistent.

1. Use the STOP QMGR command with the QUIESCE or FORCE attribute to stop the queue manager.
2. Run the SCOPY function of CSQUTIL with the PSID option, to copy all message data from the large page set and save them in a sequential data set.
3. Define a new smaller page set data set to replace the large page set.
4. Run the FORMAT TYPE(NEW) function of CSQUTIL against the page set that you created in step “3” on page 526.
5. Restart the queue manager using the page set created in step “3” on page 526.
6. Run the LOAD function of CSQUTIL to load back all the messages saved during step “2” on page 526.
7. Allow all users access to the queue manager.
8. Delete the old large page set.

How to reintroduce a page set

In certain scenarios it is useful to be able to bring an old page set online again to the queue manager. Unless specific action is taken, when the old page set is brought online the queue manager will recognize that the page set recovery RBA stored in the page set itself and in the checkpoint records is old, and will therefore automatically start media recovery of the page set to bring it up to date.

Such media recovery can only be performed at queue manager restart, and is likely to take a considerable length of time, especially if archive logs held on tape must be read. However, normally in this circumstance, the page set has been offline for the intervening period and so the log contains no information pertinent to the page set recovery.

The following three choices are available:

Allow full media recovery to be performed.

1. Stop the queue manager.
2. Ensure definitions are available for the page set in both the started task procedure for the queue manager and in the CSQINP1 initialization data set.
3. Restart the queue manager.

Allow any messages on the page set to be destroyed.

This choice is useful where a page set has been offline for a long time (some months, for example) and it has now been decided to reuse it for a different purpose.

1. Format the page set using the FORMAT function of CSQUTIL with the TYPE(NEW) option.
2. Add definitions for the page set to both the started task procedure for the queue manager and the CSQINP1 initialization data set.
3. Restart the queue manager.

Using the TYPE(NEW) option for formatting clears the current contents of the page set and tells the queue manager to ignore any historical information in the checkpoint about the page set.

Bring the page set online avoiding the media recovery process.

Use this technique only if you are sure that the page set has been offline since a clean shutdown of the queue manager. This choice is most appropriate where the page set has been offline for a short period, typically due to operational issues such as a backup running while the queue manager is being started.

1. Format the page set using the FORMAT function of CSQUTIL with the TYPE(REPLACE) option.

2. Either add the page set back into the queue manager dynamically using the DEFINE PSID command with the DSN option or allow it to be added at a queue manager restart.

Using the TYPE(REPLACE) option for formatting checks that the page set was cleanly closed by the queue manager, and marks it so that media recovery will not be performed. No other changes are made to the contents of the page set.

z/OS How to back up and recover page sets

There are different mechanisms available for back up and recovery. Use this topic to understand these mechanisms.

This section describes the following topics:

- "Creating a point of recovery for non-shared resources" on page 527
- "Backing up page sets" on page 528
- "Recovering page sets" on page 529
- How to delete page sets

For information about how to create a point of recovery for shared resources, see "Recovering shared queues" on page 535.

Creating a point of recovery for non-shared resources

IBM MQ can recover objects and non-shared persistent messages to their current state if both:

1. Copies of page sets from an earlier point exist.
2. All the IBM MQ logs are available to perform recovery from that point.

These represent a point of recovery for non-shared resources.

Both objects and messages are held on page sets. Multiple objects and messages from different queues can exist on the same page set. For recovery purposes, objects and messages cannot be backed up in isolation, so a page set must be backed up as a whole to ensure the correct recovery of the data.

The IBM MQ recovery log contains a record of all persistent messages and changes made to objects. If IBM MQ fails (for example, due to an I/O error on a page set), you can recover the page set by restoring the backup copy and restarting the queue manager. IBM MQ applies the log changes to the page set from the point of the backup copy.

There are two ways of creating a point of recovery:

Full backup
Stop the queue manager, which forces all updates on to the page sets.

This allows you to restart from the point of recovery, using only the backed up page set data sets and the logs from that point on.

Fuzzy backup
Take *fuzzy* backup copies of the page sets without stopping the queue manager.

If you use this method, and your associated logs later become damaged or lost, you cannot to use the fuzzy page set backup copies to recover. This is because the fuzzy page set backup copies contain an inconsistent view of the state of the queue manager and are dependent on the logs being available. If the logs are not available, you need to return to the last set of backup page set copies taken while the subsystem was inactive (Method 1) and accept the loss of data from that time.

Method 1: Full backup

This method involves shutting the queue manager down. This forces all updates on to the page sets so that the page sets are in a consistent state.

1. Stop all the IBM MQ applications that are using the queue manager (allowing them to complete first). This can be done by changing the access security or queue settings, for example.
2. When all activity has completed, display and resolve any in-doubt units of recovery. (Use the commands DISPLAY CONN and RESOLVE INDOUBT, as described in DISPLAY CONN and RESOLVE INDOUBT.)

   This brings the page sets to a consistent state; if you do not do this, your page sets might not be consistent, and you are effectively doing a fuzzy backup.
3. Issue the ARCHIVE LOG command to ensure that the latest log data is written out to the log data sets.
4. Issue the STOP QMGR MODE(QUIESCE) command. Record the lowest RBA value in the CSQI024I or CSQI025I messages (see CSQI024I and CSQI025I for more information). You should keep the log data sets starting from the one indicated by the RBA value up to the current log data set.
5. Take backup copies of all the queue manager page sets (see “Backing up page sets” on page 528).

Method 2: Fuzzy backup

This method does not involve shutting the queue manager down. Therefore, updates might be in virtual storage buffers during the backup process. This means that the page sets are not in a consistent state, and can only be used for recovery with the logs.

1. Issue the DISPLAY USAGE TYPE(ALL) command, and record the RBA value in the CSQI024I or CSQI025I messages (see CSQI024I and CSQI025I for more information).
2. Take backup copies of the page sets (see “Backing up page sets” on page 528).
3. Issue the ARCHIVE LOG command, to ensure that the latest log data is written out to the log data sets. To restart from the point of recovery, you must keep the log data sets starting from the log data set indicated by the RBA value up to the current log data set.

Backing up page sets

To recover a page set, IBM MQ needs to know how far back in the log to go. IBM MQ maintains a log RBA number in page zero of each page set, called the *recovery log sequence number* (LSN). This number is the starting RBA in the log from which IBM MQ can recover the page set. When you back up a page set, this number is also copied.

If the copy is later used to recover the page set, IBM MQ must have access to all the log records from this RBA value to the current RBA. That means you must keep enough of the log records to enable IBM MQ to recover from the oldest backup copy of a page set you intend to keep.

Use ADRDSSU COPY function to copy the page sets.

For more information, see the COPY DATASET Command Syntax for Logical Data Set documentation .

For example:

```
//STEP2 EXEC PGM=ADRDSSU,REGION=6M
//SYSPRINT DD SYSOUT=H
//SYSIN DD *
 COPY -
 DATASET(INCLUDE(SCENDATA.MQPA.PAGESET.*)) -
 RENAMEU(SCENDATA.MQPA.PAGESET.**,SCENDATA.MQPA.BACKUP1.**) -
 SPHERE -
 REPUNC -
 FASTREPLICATION(PREF )-
 CANCELERROR -
 TOL(ENQF)
/*
//
```

If you copy the page set while the queue manager is running you must use a copy utility that copies page zero of the page set first. If you do not do this you could corrupt the data in your page set.

If the process of dynamically expanding a page set is interrupted, for example by power to the system being lost, you can still use ADRDSSU to take a backup of a page set.

If you perform an Access Method Services IDCAMS `LISTCAT ENT('page set data set name') ALLOC`, you will see that the `HI-ALLOC-RBA` is higher than the `HI-USED-RBA`.

The next time this page set fills up it is extended again, if possible, and the pages between the high used RBA and the highest allocated RBA are used, along with another new extent.

Backing up your object definitions

You should also back up copies of your object definitions. To do this, use the MAKEDEF feature of the CSQUTIL COMMAND function (described in Issuing commands to IBM MQ (COMMAND)).

Back up your object definitions whenever you take a backup copy of your queue manager, and keep the most current version.

Recovering page sets

If the queue manager has terminated due to a failure, the queue manager can normally be restarted with all recovery being performed during restart. However, such recovery is not possible if any of your page sets or log data sets are not available. The extent to which you can now recover depends on the availability of backup copies of page sets and log data sets.

To restart from a point of recovery you must have:

- A backup copy of the page set that is to be recovered.
- If you used the "fuzzy" backup process described in “Method 2: Fuzzy backup” on page 528, the log data set that included the recorded RBA value, the log data set that was made by the ARCHIVE LOG command, and all the log data sets between these.
- If you used full backup, but you do not have the log data sets following that made by the ARCHIVE LOG command, you do **not** need to run the FORMAT TYPE(REPLACE) function of the CSQUTIL utility against all your page sets.

To recover a page set to its current state, you must also have all the log data sets and records since the ARCHIVE LOG command.

There are two methods for recovering a page set. To use either method, the queue manager must be stopped.

Simple recovery

This is the simpler method, and is appropriate for most recovery situations.

1. Delete the page set you want to restore from backup.
2. Use the ADRDSSU COPY function to recover your page set from the backup copy..

   Alternatively, you can rename your backup copy to the original name, or change the CSQP00xx DD statement in your queue manager procedure to point to your backup page set. However, if you then lose or corrupt the page set, you will no longer have a backup copy to restore from.
3. Restart the queue manager.
4. When the queue manager has restarted successfully, you can restart your applications
5. Reinstate your normal backup procedures for the restored page.

Advanced recovery

This method provides performance advantages if you have a large page set to recover, or if there has been much activity on the page set since the last backup copy was taken. However, it requires more manual intervention than the simple method, which might increase the risk of error and the time taken to perform the recovery.

1. Delete and redefine the page set you want to restore from backup.
2. Use ADRDSSU to copy the backup copy of the page set into the new page set. Define your new page set with a secondary extent value so that it can be expanded dynamically.

   Alternatively, you can rename your backup copy to the original name, or change the CSQP00xx DD statement in your queue manager procedure to point to your backup page set. However, if you then lose or corrupt the page set, you will no longer have a backup copy to restore from.
3. Change the CSQINP1 definitions for your queue manager to make the buffer pool associated with the page set being recovered as large as possible. By making the buffer pool large, you might be able to keep all the changed pages resident in the buffer pool and reduce the amount of I/O to the page set.
4. Restart the queue manager.
5. When the queue manager has restarted successfully, stop it (using quiesce) and then restart it using the normal buffer pool definition for that page set. After this second restart completes successfully, you can restart your applications
6. Reinstate your normal backup procedures for the restored page.

What happens when the queue manager is restarted

When the queue manager is restarted, it applies all changes made to the page set that are registered in the log, beginning at the restart point for the page set. IBM MQ can recover multiple page sets in this way. The page set is dynamically expanded, if required, during media recovery.

During restart, IBM MQ determines the log RBA to start from by taking the lowest value from the following:

- Recovery LSN from the checkpoint log record for each page set.
- Recovery LSN from page zero in each page set.
- The RBA of the oldest incomplete unit of recovery in the system at the time the backup was taken.

All object definitions are stored on page set zero. Messages can be stored on any available page set.

Note: The queue manager cannot restart if page set zero is not available.

How to delete page sets

You delete a page set by using the DELETE PSID command; see DELETE PSID for details of this command.

You cannot delete a page set that is still referenced by any storage class. Use DISPLAY STGCLASS to find out which storage classes reference a page set.

The data set is deallocated from IBM MQ but is not deleted. It remains available for future use, or can be deleted using z/OS facilities.

Remove the page set from the started task procedure for your queue manager.

Remove the definition of the page set from your CSQINP1 initialization data set.

z/OS How to back up and restore queues using CSQUTIL

Use this topic as a reference for further information about back up and restore using CSQUTIL.

You can use the CSQUTIL utility functions for backing up and restoring queues. To back up a queue, use the COPY or SCOPY function to copy the messages from a queue onto a data set. To restore the queue, use the complementary function LOAD or SLOAD. For more information, see IBM MQ utility program.

z/OS Managing buffer pools

Use this topic if you want to change or delete your buffer pools.

This topic describes how to alter and delete buffer pools. It contains these sections:

- "How to change the number of buffers in a buffer pool" on page 531
- "How to delete a buffer pool" on page 531

Buffer pools are defined during queue manager initialization, using DEFINE BUFFPOOL commands issued from the initialization input data set CSQINP1. Their attributes can be altered in response to business requirements while the queue manager is running, using the processes detailed in this topic. The queue manager records the current buffer pool attributes in checkpoint log records. These are automatically restored on subsequent queue manager restart, unless the buffer pool definition in CSQINP1 includes the REPLACE attribute.

Use the DISPLAY USAGE command to display the current buffer attributes.

You can also define buffer pools dynamically using the DEFINE PSID command with the DSN option.

If you change buffer pools dynamically, you should also update their definitions in the initialization data set CSQINP1.

See Planificación en z/OS for a description of page sets, storage classes, buffers, and buffer pools, and some of the performance considerations that apply.

Note: Buffer pools use significant storage. When you increase the size of a buffer pool or define a new buffer pool ensure that sufficient storage is available. For more information, see Address space storage.

How to change the number of buffers in a buffer pool

If a buffer pool is too small, the condition can result in message CSQP020E on the console, you can allocate more buffers to it using the ALTER BUFFPOOL command as follows:

1. Determine how much space is available for new buffers by looking at the CSQY220I messages in the log. The available space is reported in MB. As a buffer has a size of 4 KB, each MB of available space allows you to allocate 256 buffers. Do not allocate all the free space to buffers, as some is required for other tasks.

   If the buffer pool uses fixed 4 KB pages, that is, its PAGECLAS attribute is FIXED4KB, ensure that there is sufficient real storage available on the LPAR.
2. If the reported free space is inadequate, release some buffers from another buffer pool using the command

   ```
   ALTER BUFFPOOL(buf-pool-id) BUFFERS(integer)
   ```

   where *buf-pool-id* is the buffer pool from which you want to reclaim space and *integer* is the new number of buffers to be allocated to this buffer pool, which must be smaller than the original number of buffers allocated to it.
3. Add buffers to the buffer pool you want to expand using the command

   ```
   ALTER BUFFPOOL(buf-pool-id) BUFFERS(integer)
   ```

   where *buf-pool-id* is the buffer pool to be expanded and *integer* is the new number of buffers to be allocated to this buffer pool, which must be larger than the original number of buffers allocated to it.

How to delete a buffer pool

When a buffer pool is no longer used by any page sets, delete it to release the virtual storage allocated to it.

You delete a buffer pool using the DELETE BUFFPOOL command. The command fails if any page sets are using this buffer pool.

See "How to delete page sets" on page 530 for information about how to delete page sets.

z/OS Managing queue sharing groups and shared queues on z/OS

IBM MQ can use different types of shared resources, for example queue sharing groups, shared queues, and the coupling facility. Use this topic to review the procedures needed to manage these shared resources.

This section contains information about the following topics:

z/OS *Managing queue sharing groups*

You can add or remove a queue manager to a queue sharing group (QSG), and manage the associated Db2 tables.

This topic has sections about the following tasks:

Setting up a queue sharing group

Each queue sharing group has a name of up to four characters. The name must be unique in your network, and must be different from any queue manager names.

Follow these steps to set up a queue sharing group:

1. If this is the first queue sharing group to use the Db2 data-sharing group, set up the Db2 environment.
2. Set up the coupling facility.
3. Add the queue sharing group to the Db2 tables. Use the ADD QSG function of the queue sharing group utility (CSQ5PQSG). This program is described in The queue sharing group utility. A sample is provided in thlqual.SCSQPROC(CSQ45AQS).
4. Add a queue manager to the queue sharing group by following the steps in "Adding a queue manager to a queue sharing group" on page 533
5. Define application structures to IBM MQ by following the steps in "Adding a coupling facility structure" on page 540.
6. If required, migrate non-shared queues to shared queues.
7. For availability, create shared channels into and out of the queue sharing group.
   - For connections into the queue sharing group:
     - Set up a VIPA socket or hardware router to distribute workload between the available queue managers in the QSG.
     - Define a receiver channel with QSGDISP(GROUP), to ensure the channel definition is available on all queue managers in the QSG.
     - Start a listener with INDISP(GROUP), on each queue manager, for MCA channel connections into the QSG. Client connections into the QSG should still connect to a listener started with INDISP(QMGR).
     - Change applications to connect using the QSG name, rather than a specific queue manager name.

  - Ensure that the channel authentication rules on all queue managers in the QSG are the same, to allow applications to connect to any queue manager in the QSG.
- For connections out of the queue sharing group:
  - Define a shared transmission queue.
  - Define the outbound channel with QSGDISP(GROUP) and DEFCDISP(SHARED).

  If you convert an existing channel to a shared channel, you might need to issue the RESET CHANNEL command before starting the channel as the synchronization queue used by the channel will have changed.

Adding a queue manager to a queue sharing group

A queue manager can be added to an existing queue sharing group.

Note that:

- The queue sharing group must exist before you can add queue managers to it.
- A queue manager can be a member of only one queue sharing group.

Follow these steps to add a queue manager to a queue sharing group:

1. Perform the tasks in implement ESM security controls for the queue sharing group to grant the appropriate access to the queue manager and channel initiator user IDs.
2. If the queue sharing group has CF structures configured to offload data to SMDS, perform the tasks in set up the SMDS environment.
3. Stop the queue manager.
4. Use the ADD QMGR function of the queue sharing group utility (CSQ5PQSG). This program is described in the queue sharing group utility. A sample is provided in thlqual.SCSQPROC(CSQ45AQM).
5. Change your system parameter module to add queue sharing group data:
   a. Modify CSQ6SYSP to specify the QSGDATA parameter. See using CSQ6SYSP for more information.
   b. Assemble and link the system parameter module. You might want to use a different name for the load module.
   c. Change your startup process to use the new module.
6. Copy and tailor sample member thlqual.SCSQPROC(CSQ4INSS), which defines required CF structures and SYSTEM queues. Add the customized member to the CSQINP2 DD in the queue manager startup JCL.
7. Restart your queue manager using the queue sharing group system parameter module.
8. Optionally, migrate to security profiles prefixed by the queue sharing group name, instead of the queue manager name.
9. If shared channels are used for connections into the QSG, create channel authentication rules that mirror those on the other queue managers in the QSG, to allow applications to connect to any queue manager in the QSG.
10. 10. Optionally, do either of the following to allow applications connected to the queue manager in the QSG to put messages to queues hosted by other queue managers in the QSG:
    - Turn on intra-group queuing by issuing the command ALTER QMGR IGQ(ENABLED).
    - Define transmission queues and channels to the other queue managers in the QSG. Defining transmission queues with the same name as the target queue managers avoids the need to define remote queues and queue manager aliases.

Note: To add a queue manager to an existing queue sharing group containing queue managers running earlier versions of IBM MQ, you must first apply the coexistence PTF for the highest version of IBM MQ in the group to every earlier version queue manager in the group.

Removing a queue manager from a queue sharing group

You can only remove a queue manager from a queue sharing group if the queue manager's logs are not needed by another process, and all SMDS owned by the queue manager are empty.

See Deleting shared message data sets and DELETE CFSTRUCT for more information.

The logs are needed if they contain:

- The latest backup of one of the coupling facility (CF) application structures used by the queue sharing group
- Data needed by a future restore process, that is, the queue manager has used a recoverable structure since the time described by the last backup exclusion interval value.

If either or both of these points apply, or an SMDS owned by the queue manager contains messages, the queue manager cannot be removed. To determine which queue managers' logs are needed for a future restore process, use the MQSC DISPLAY CFSTATUS command with the TYPE(BACKUP) option (for details of this command, see DISPLAY CFSTATUS).

Use the following steps to remove a queue manager from a queue sharing group:

1. Stop any applications connected to the queue manager that put messages to shared queues.
2. Resolve any indoubt units of work involving this queue manager.
3. Determine if there are any messages in any SMDS owned by the queue manager by issuing the command DISPLAY USAGE TYPE(SMDS).
4. If there are offloaded messages for any application structure, wait until those messages have been retrieved from the queue. The number of offloaded messages reported by DISPLAY USAGE TYPE(SMDS) should be zero before proceeding.
5. Shut the queue manager down cleanly using STOP QMGR MODE(QUIESCE).
6. Wait for an interval at least equivalent to the value of the EXCLINT parameter you will specify in the BACKUP CFSTRUCT command in the next step.
7. On another queue manager, run a CF structure backup for each recoverable CF structure by using the MQSC BACKUP CFSTRUCT command and specifying an EXCLINT value as required in the previous step.
8. Confirm that the queue manager's logs are not needed to restore any CF structures, by inspecting the output from the command DISPLAY CFSTATUS(*) TYPE(BACKUP).
9. Use the REMOVE QMGR function of the CSQ5PQSG utility to remove the queue manager from the queue sharing group. This program is described in The queue sharing group utility. A sample is provided in thlqual.SCSQPROC(CSQ45RQM).
10. Before restarting the queue manager, reset the QSGDATA system parameter to its default value, and recreate the system parameter module. See Using CSQ6SYSP for information about how to tailor your system parameters.

Note, that when removing the last queue manager in a queue sharing group, you must use the FORCE option, rather than REMOVE. This removes the queue manager from the queue sharing group, while not performing the consistency checks of the queue manager logs being required for recovery. You should only perform this operation if you are deleting the queue sharing group.

Removing a queue sharing group from the Db2 tables

To remove a queue sharing group from the Db2 tables, use the REMOVE QSG function of the queue sharing group utility (CSQ5PQSG). This program is described in The queue sharing group utility. A sample is provided in thlqual.SCSQPROC(CSQ45RQS).

You can only remove a queue sharing group from the common Db2 data-sharing group tables after you have removed all the queue managers from the queue sharing group (as described in “Removing a queue manager from a queue sharing group” on page 534).

When the queue sharing group record is deleted from the queue sharing group administration table, all objects and administrative information relating to that queue sharing group are deleted from other IBM MQ Db2 tables. This includes shared queue and group object information.

Validating the consistency of Db2 definitions

Problems for shared queues within a queue sharing group can occur if the Db2 object definitions have, for any reason, become inconsistent.

To validate the consistency of the Db2 object definitions for queue managers, CF structures, and shared queues, use the VERIFY QSG function of the queue sharing group utility (CSQ5PQSG). This program is described in The queue sharing group utility.

z/OS *Managing shared queues*

Use this topic to understand how to recover, move, and migrate shared queues.

This section describes the following tasks:

- “Recovering shared queues” on page 535
- “Moving shared queues” on page 536
- “Migrating non-shared queues to shared queues” on page 538
- Suspending a Db2 connection

Recovering shared queues

IBM MQ can recover persistent messages on shared queues if all:

- Backups of the CF structures containing the messages have been performed.
- All the logs for all queue managers in the queue sharing group are available, to perform recovery from the point the backups are taken.
- Db2 is available and the structure backup table is more recent that the most recent CF structure backup.

The messages on a shared queue are stored in a coupling facility (CF) structure. Persistent messages can be put onto shared queues, and like persistent messages on non-shared queues, they are copied to the queue manager log. The MQSC BACKUP CFSTRUCT and RECOVER CFSTRUCT commands are provided to allow the recovery of a CF structure in the unlikely event of a coupling facility failure. In such circumstances, any nonpersistent messages stored in the affected structure are lost, but persistent messages can be recovered. Any further application activity using the structure is prevented until the structure has been recovered.

To enable recovery, you must back up your coupling facility list structures frequently using the MQSC BACKUP CFSTRUCT command. The messages in the CF structure are written onto the active log data set of the queue manager making the backup. It writes a record of the backup to Db2: the name of the CF structure being backed up, the name of the queue manager doing the backup, the RBA range for this backup on that queue manager log, and the backup time. Back up CF list structures even if you are not actively using shared queues, for example, if you have set up a queue sharing group intending to use it in the future.

You can recover a CF structure by issuing an MQSC RECOVER CFSTRUCT command to the queue manager that can perform the recovery; you can use any queue manager in the queue sharing group. You can specify a single CF structure to be recovered, or you can recover several CF structures simultaneously.

As noted previously, it is important that you back up your CF list structures frequently, otherwise recovering a CF structure can take a long time. Moreover, the recovery process cannot be canceled.

The definition of a shared queue is kept in a Db2 database and can therefore be recovered if necessary using standard Db2 database procedures. See Shared queues and queue sharing groups for more information.

Moving shared queues

This section describes how to perform load balancing by moving a shared queue from one coupling facility structure to another. It also describes how to move a non-shared queue to a shared queue, and how to move a shared queue to a non-shared queue.

When you move a queue, you need to define a temporary queue as part of the procedure. This is because every queue must have a unique name, so you cannot have two queues of the same name, even if the queues have different queue dispositions. IBM MQ tolerates having two queues with the same name (as in step "2" on page 536), but you cannot use the queues.

- Moving a queue from one coupling facility structure to another
- Moving a non-shared queue to a shared queue
- Moving a shared queue to a non-shared queue

Moving a queue from one coupling facility structure to another

To move queues and their messages from one CF structure to another, use the MQSC MOVE QLOCAL command. When you have identified the queue or queues that you want to move to a new CF structure, use the following procedure to move each queue:

1. Ensure that the queue you want to move is not in use by any applications, that is, the queue attributes IPPROCS and OPPROCS are zero on all queue managers in the queue sharing group.
2. Prevent applications from putting messages on the queue being moved by altering the queue definition to disable MQPUT s. Change the queue definition to PUT(DISABLED).
3. Define a temporary queue with the same attributes as the queue that is being moved using the following command:

```
DEFINE QL(TEMP_QUEUE) LIKE(QUEUE_TO_MOVE) PUT(ENABLED) GET(ENABLED) QSGDISP(QMGR)
```

   Note: If this temporary queue exists from a previous run, delete it before doing the define.
4. Move the messages to the temporary queue using the following command:

```
MOVE QLOCAL(QUEUE_TO_MOVE) TOQLOCAL(TEMP_QUEUE)
```

5. Delete the queue you are moving, using the command:

```
DELETE QLOCAL(QUEUE_TO_MOVE)
```

6. Redefine the queue that is being moved, changing the CFSTRUCT attribute, using the following command:

```
DEFINE QL(QUEUE_TO_MOVE) LIKE(TEMP_QUEUE) CFSTRUCT(NEW) QSGDISP(SHARED)
```

   When the queue is redefined, it is based on the temporary queue created in step "3" on page 536.
7. Move the messages back to the new queue using the command:

```
MOVE QLOCAL(TEMP) TOQLOCAL(QUEUE_TO_MOVE)
```

8. The queue created in step “3” on page 536 is no longer required. Use the following command to delete it:

```
DELETE QL(TEMP_QUEUE)
```

9. If the queue being moved was defined in the CSQINP2 data sets, change the CFSTRUCT attribute of the appropriate DEFINE QLOCAL command in the CSQINP2 data sets. Add the REPLACE keyword so that the existing queue definition is replaced.

Figure 30 on page 537 shows a sample job for moving a queue from one CF structure to another.

```
//UTILITY EXEC PGM=CSQUTIL,PARM=('CSQ1')
//STEPLIB DD  DSN=thlqual.SCSQANLE,DISP=SHR
//     DD  DSN=thlqual.SCSQAUTH,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//SYSIN  DD  *
COMMAND DDNAME(MOVEQ)
/*
//MOVEQ DD  *
ALTER QL(QUEUE_TO_MOVE) PUT(DISABLED)
DELETE QL(TEMP_QUEUE) PURGE
DEFINE QL(TEMP_QUEUE) LIKE(QUEUE_TO_MOVE) PUT(ENABLED) GET(ENABLED) QSGDISP(QMGR)
MOVE QLOCAL(QUEUE_TO_MOVE) TOQLOCAL(TEMP_QUEUE)
DELETE QL(QUEUE_TO_MOVE)
DEFINE QL(QUEUE_TO_MOVE) LIKE(TEMP_QUEUE) CFSTRUCT(NEW) QSGDISP(SHARED)
MOVE QLOCAL(TEMP_QUEUE) TOQLOCAL(QUEUE_TO_MOVE)
DELETE QL(TEMP_QUEUE)
/*
```

Figure 30. Sample job for moving a queue from one CF structure to another

Moving a non-shared queue to a shared queue

The procedure for moving a non-shared queue to a shared queue is like the procedure for moving a queue from one CF structure to another (see “Moving a queue from one coupling facility structure to another” on page 536). Figure 31 on page 537 gives a sample job to do this.

Note: Remember that messages on shared queues are subject to certain restrictions on the maximum message size, message persistence, and queue index type, so you might not be able to move some non-shared queues to a shared queue.

```
//UTILITY EXEC PGM=CSQUTIL,PARM=('CSQ1')
//STEPLIB DD  DSN=thlqual.SCSQANLE,DISP=SHR
//     DD  DSN=thlqual.SCSQAUTH,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//SYSIN  DD  *
COMMAND DDNAME(MOVEQ)
/*
//MOVEQ DD  *
ALTER QL(QUEUE_TO_MOVE) PUT(DISABLED)
DELETE QL(TEMP_QUEUE) PURGE
DEFINE QL(TEMP_QUEUE) LIKE(QUEUE_TO_MOVE) PUT(ENABLED) GET(ENABLED)
MOVE QLOCAL(QUEUE_TO_MOVE) TOQLOCAL(TEMP_QUEUE)
DELETE QL(QUEUE_TO_MOVE)
DEFINE QL(QUEUE_TO_MOVE) LIKE(TEMP_QUEUE) CFSTRUCT(NEW) QSGDISP(SHARED)
MOVE QLOCAL(TEMP_QUEUE) TOQLOCAL(QUEUE_TO_MOVE)
DELETE QL(TEMP_QUEUE)
/*
```

Figure 31. Sample job for moving a non-shared queue to a shared queue

Moving a shared queue to a non-shared queue

The procedure for moving a shared queue to a non-shared queue is like the procedure for moving a queue from one CF structure to another (see “Moving a queue from one coupling facility structure to another” on page 536).

Figure 32 on page 538 gives a sample job to do this.

```
//UTILITY EXEC PGM=CSQUTIL,PARM=('CSQ1')
//STEPLIB DD  DSN=thlqual.SCSQANLE,DISP=SHR
//     DD  DSN=thlqual.SCSQAUTH,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//SYSIN  DD  *
COMMAND DDNAME(MOVEQ)
/*
//MOVEQ DD  *
ALTER QL(QUEUE_TO_MOVE) PUT(DISABLED)
DELETE QL(TEMP_QUEUE) PURGE
DEFINE QL(TEMP_QUEUE) LIKE(QUEUE_TO_MOVE) PUT(ENABLED) GET(ENABLED) QSGDISP(QMGR)
MOVE QLOCAL(QUEUE_TO_MOVE) TOQLOCAL(TEMP_QUEUE)
DELETE QL(QUEUE_TO_MOVE)
DEFINE QL(QUEUE_TO_MOVE) LIKE(TEMP_QUEUE) STGCLASS(NEW) QSGDISP(QMGR)
MOVE QLOCAL(TEMP_QUEUE) TOQLOCAL(QUEUE_TO_MOVE)
DELETE QL(TEMP_QUEUE)
/*
```

Figure 32. Sample job for moving a shared queue to a non-shared queue

Migrating non-shared queues to shared queues

There are two stages to migrating non-shared queues to shared queues:

- Migrating the first (or only) queue manager in the queue sharing group
- Migrating any other queue managers in the queue sharing group

Migrating the first (or only) queue manager in the queue sharing group

Figure 31 on page 537 shows an example job for moving a non-shared queue to a shared queue. Do this for each queue that needs migrating.

Note:

1. Messages on shared queues are subject to certain restrictions on the maximum message size, message persistence, and queue index type, so you might not be able to move some non-shared queues to a shared queue.
2. You must use the correct index type for shared queues. If you migrate a transmission queue to be a shared queue, the index type must be MSGID.

If the queue is empty, or you do not need to keep the messages that are on it, migrating the queue is simpler. Figure 33 on page 539 shows an example job to use in these circumstances.

```
//UTILITY EXEC PGM=CSQUTIL,PARM=('CSQ1')
//STEPLIB DD  DSN=thlqual.SCSQANLE,DISP=SHR
//     DD  DSN=thlqual.SCSQAUTH,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//SYSIN  DD  *
COMMAND DDNAME(MOVEQ)
/*
//MOVEQ DD  *
DELETE QL(TEMP_QUEUE) PURGE
DEFINE QL(TEMP_QUEUE) LIKE(QUEUE_TO_MOVE) PUT(ENABLED) GET(ENABLED)
DELETE QL(QUEUE_TO_MOVE)
DEFINE QL(QUEUE_TO_MOVE) LIKE(TEMP_QUEUE) CFSTRUCT(NEW) QSGDISP(SHARED)
DELETE QL(TEMP_QUEUE)
/*
```

Figure 33. Sample job for moving a non-shared queue without messages to a shared queue

Migrating any other queue managers in the queue sharing group

1. For each queue that does not have the same name as an existing shared queue, move the queue as described in Figure 31 on page 537 or Figure 33 on page 539.
2. For queues that have the same name as an existing shared queue, move the messages to the shared queue using the commands shown in Figure 34 on page 539.

```
MOVE QLOCAL(QUEUE_TO_MOVE) QSGDISP(QMGR) TOQLOCAL(QUEUE_TO_MOVE)
DELETE QLOCAL(QUEUE_TO_MOVE) QSGDISP(QMGR)
```

Figure 34. Moving messages from a non-shared queue to an existing shared queue

Suspending a connection to Db2

If you want to apply maintenance or service to the Db2 tables or package related to shared queues without stopping your queue manager, you must temporarily disconnect queue managers in the data sharing group (DSG) from Db2.

To do this:

1. Use the MQSC command SUSPEND QMGR FACILITY(Db2).
2. Do the binds.
3. Reconnect to Db2 using the MQSC command RESUME QMGR FACILITY(Db2)

Note that there are restrictions on the use of these commands.

Attention: While the Db2 connection is suspended, the following operations will not be available. Therefore, you need to do this work during a time when your enterprise is at its least busy.

- Access to Shared queue objects for administration (define, delete,alter)
- Starting shared channels
- Storing messages in Db2
- Backup or recover CFSTRUCT

z/OS *Managing group objects*

Use this topic to understand how to work with group objects.

IBM MQ automatically copies the definition of a group object to page set zero of each queue manager that uses it. You can alter the copy of the definition temporarily, and IBM MQ allows you to refresh the page set copies from the repository copy. IBM MQ always tries to refresh the page set copies from the repository

copy on start-up (for channel objects, this is done when the channel initiator restarts). This ensures that the page set copies reflect the version on the repository, including any changes that were made when the queue manager was inactive.

There are circumstances under which the refresh is not performed, for example:

- If a copy of the queue is open, a refresh that would change the usage of the queue fails.
- If a copy of a queue has messages on it, a refresh that would delete that queue fails.

In these circumstances, the refresh is not performed on that copy, but is performed on the copies on all other queue managers. Check for and correct any problems with copy objects after adding, changing, or deleting a group object, and at queue manager or channel initiator restart.

z/OS *Managing the coupling facility*

Use this topic to understand how to add or remove coupling facility (CF) structures.

This section describes the following tasks:

- "Adding a coupling facility structure" on page 540
- "Removing a coupling facility structure" on page 540

Adding a coupling facility structure

To add a coupling facility structure, use the following procedure:

1. Define the CF structure in the CFRM policy data set. The information about setting up the coupling facility in Set up the coupling facility describes the rules for naming coupling facility structures, and how to define structures in the CFRM policy data set.
2. If you want to configure the structure to offload message data to SMDS, allocate and preformat data sets. See creating a shared message data set for details.
3. Define the structure to IBM MQ using the DEFINE CFSTRUCT command.

Removing a coupling facility structure

To remove a coupling facility structure, use the following procedure:

1. Use the following command to get a list of all the queues using the coupling facility structure that you want to delete:

```
DISPLAY QUEUE(*) QSGDISP(SHARED) CFSTRUCT(structure-name)
```

2. Delete all the queues that use the structure.
3. Delete the CF structure from IBM MQ using the DELETE CFSTRUCT command.
4. If the structure was configured to offload message data to SMDS, delete the SMDS.
5. Remove the structure definition from your CFRM policy data set and run the IXCMIAPU utility. (This is the reverse of the customization task set up the coupling facility, described in Set up the coupling facility.)

z/OS *Tuning coupling facility list monitoring*

Use this topic to understand coupling facility list monitoring

Coupling facility (CF) list monitoring is used to monitor the state of list structures containing IBM MQ shared queues. When a message is added to a shared queue, and the queue's depth transitions from zero to non-zero, the CF notifies all queue managers in the queue sharing group. When notified the

queue managers might perform a number of actions, including notifying trigger monitors that are using TRIGGER(FIRST), or applications which are performing a get-wait.

By default, the CF notifies all queue managers in the queue sharing group at the same time. In certain configurations this can cause problems, such as:

- Skewed workload distribution, where a large percentage of messages go to a particular queue manager in the queue sharing group, often the queue manager running on the fastest LPAR, or which is closest to the CF, or
- A large number of failed gets, resulting in wasted CPU time.

z/OS V2R3 introduces a new coupling facility resource manager (CFRM) attribute called **KEYRNOTIFYDELAY**, which can be used with list structures containing shared queues (that is, application structures, and not the admin structure), and which can, for certain workloads, minimize the effects of workload skewing and empty MQGET calls, or empty MQGET calls.

KEYRNOTIFYDELAY can only be set on structures in a CF, running at CFLEVEL 22 or higher.

Its value must be one to seven decimal digits, in a range from 0 to 1,000,000 microseconds. If set to a non-zero value and the depth of a queue transitions from zero to non-zero, the CF selects a single queue manager from the queue sharing group, and notifies that queue manager before all the other queue managers in the group.

The queue manager is selected in a round-robin manner. If the selected queue manager does not process the message inside the time interval described by **KEYRNOTIFYDELAY** all the other queue managers in the queue sharing group will also be notified.

More information on **KEYRNOTIFYDELAY** is available here: Understanding Keyrange Monitoring Notification Delay.

Note that there are two similar CFRM attributes called **LISTNOTIFYDELAY** and **SUBNOTIFYDELAY**. Neither of these has any measurable effect on IBM MQ workload.

z/OS Recovery and restart on z/OS

Use this topic to understand the recovery and restart mechanisms used by IBM MQ.

z/OS Restarting IBM MQ

After a queue manager terminates there are different restart procedures needed depending on how the queue manager terminated. Use this topic to understand the different restart procedures that you can use.

This topic contains information about how to restart your queue manager in the following circumstances:

- “Restarting after a normal shutdown” on page 541
- “Restarting after an abnormal termination” on page 542
- “Restarting if you have lost your page sets” on page 542
- “Restarting if you have lost your log data sets” on page 542
- Restarting if you have lost your CF structures

Restarting after a normal shutdown

If the queue manager was stopped with the STOP QMGR command, the system finishes its work in an orderly way and takes a termination checkpoint before stopping. When you restart the queue manager, it uses information from the system checkpoint and recovery log to determine the system status at shutdown.

To restart the queue manager, issue the START QMGR command as described in “Using MQSC to start and stop a queue manager on z/OS” on page 467.

Restarting after an abnormal termination

IBM MQ automatically detects whether restart follows a normal shutdown or an abnormal termination.

Starting the queue manager after it has terminated abnormally is different from starting it after the STOP QMGR command has been issued. If the queue manager terminates abnormally, it terminates without being able to finish its work or take a termination checkpoint.

To restart the queue manager, issue the START QMGR command as described in "Using MQSC to start and stop a queue manager on z/OS" on page 467. When you restart a queue manager after an abnormal termination, it refreshes its knowledge of its status at termination using information in the log, and notifies you of the status of various tasks.

Normally, the restart process resolves all inconsistent states. But, in some cases, you must take specific steps to resolve inconsistencies. This is described in "Recovering units of work manually" on page 554.

Restarting if you have lost your page sets

If you have lost your page sets, you need to restore them from your backup copies before you can restart the queue manager. This is described in "How to back up and recover page sets" on page 527.

The queue manager might take a long time to restart under these circumstances because of the length of time needed for media recovery.

Restarting if you have lost your log data sets

If, after stopping a queue manager (using the STOP QMGR command), both copies of the log are lost or damaged, you can restart the queue manager providing you have a consistent set of page sets (produced using Method 1: Full backup).

Follow this procedure:

1. Define new page sets to correspond to each existing page set in your queue manager. See Task 15: Define your page sets for information about page set definition.

   Ensure that each new page set is larger than the corresponding source page set.
2. Use the FORMAT function of CSQUTIL to format the destination page set. See Formatting page sets for more details.
3. Use the RESETPAGE function of CSQUTIL to copy the existing page sets or reset them in place, and reset the log RBA in each page. See Copying a page set and resetting the log for more information about this function.
4. Redefine your queue manager log data sets and BSDS using CSQJU003 (see The change log inventory utility).
5. Restart the queue manager using the new page sets. To do this, you do one of the following:
   - Change the queue manager started task procedure to reference the new page sets. See Task 6: Create procedures for the IBM MQ queue manager for more information.
   - Use Access Method Services to delete the old page sets and then rename the new page sets, giving them the same names as the old page sets.

Attention: Before you delete any IBM MQ page set, ensure that you have made the required backup copies.

If the queue manager is a member of a queue sharing group, GROUP and SHARED object definitions are not normally affected by lost or damaged logs. However, if any shared-queue messages are involved in a unit of work that was covered by the lost or damaged logs, the effect on such uncommitted messages is unpredictable.

Note: If logs are damaged and the queue manager is a member of a queue sharing group, the ability to recover shared persistent messages might be lost. Issue a BACKUP CFSTRUCT command immediately on another active queue manager in the queue sharing group for all CF structures with the RECOVER(YES) attribute.

Restarting if you have lost your CF structures

You do not need to restart if you lose your CF structures, because the queue manager does not terminate.

z/OS *Alternative site recovery on z/OS*

You can recover a single queue manager or a queue sharing group, or consider disk mirroring.

See the following sections for more details:

- Recovering a single queue manager at an alternative site
- Recovering a queue sharing group.
  - CF structure media recovery
  - Backing up the queue sharing group at the prime site
  - Recovering a queue sharing group at the alternative site
- Using disk mirroring

Recovering a single queue manager at an alternative site

If a total loss of an IBM MQ computing center occurs, you can recover on another queue manager or queue sharing group at a recovery site. (See “Recovering a queue sharing group at the alternative site” on page 546 for the alternative site recovery procedure for a queue sharing group.)

To recover on another queue manager at a recovery site, you must regularly back up the page sets and the logs. As with all data recovery operations, the objectives of disaster recovery are to lose as little data, workload processing (updates), and time, as possible.

At the recovery site:

- The recovery queue managers **must** have the same names as the lost queue managers.
- The system parameter module (for example, CSQZPARM) used on each recovery queue manager must contain the same parameters as the corresponding lost queue manager.

When you have done this, reestablish all your queue managers as described in the following procedure. This can be used to perform disaster recovery at the recovery site for a single queue manager. It assumes that all that is available are:

- Copies of the archive logs and BSDSs created by normal running at the primary site (the active logs will have been lost along with the queue manager at the primary site).
- Copies of the page sets from the queue manager at the primary site that are the same age or older than the most recent archive log copies available.

You can use dual logging for the active and archive logs, in which case you need to apply the BSDS updates to both copies:

1. Define new page set data sets and load them with the data in the copies of the page sets from the primary site.
2. Define new active log data sets.
3. Define a new BSDS data set and use Access Method Services REPRO to copy the most recent archived BSDS into it.
4. Use the print log map utility CSQJU004 to print information from this most recent BSDS. At the time this BSDS was archived, the most recent archived log you have would have just been truncated as an active log, and does not appear as an archived log. Record the STARTRBA and ENDRBA of this log.

5. Use the change log inventory utility, CSQJU003, to register this latest archive log data set in the BSDS that you have just restored, using the STARTRBA and ENDRBA recorded in Step “4” on page 543.
6. Use the DELETE option of CSQJU003 to remove all active log information from the BSDS.
7. Use the NEWLOG option of CSQJU003 to add active logs to the BSDS, do not specify STARTRBA or ENDRBA.
8. Use CSQJU003 to add a restart control record to the BSDS. Specify CRESTART CREATE,ENDRBA=highrba, where highrba is the high RBA of the most recent archive log available (found in Step “4” on page 543), plus 1.

   The BSDS now describes all active logs as being empty, all the archived logs you have available, and no checkpoints beyond the end of your logs.
9. Restart the queue manager with the START QMGR command. During initialization, an operator reply message such as the following is issued:

   ```
   CSQJ245D +CSQ1 RESTART CONTROL INDICATES TRUNCATION AT RBA highrba.
   REPLY Y TO CONTINUE, N TO CANCEL
   ```

   Type Y to start the queue manager. The queue manager starts, and recovers data up to ENDRBA specified in the CRESTART statement.

See IBM MQ utilities on z/OS reference for information about using CSQJU003 and CSQJU004.

The following example shows sample input statements for CSQJU003 for steps 6, 7, and 8:

```
* Step 6
DELETE DSNAME=MQM2.LOGCOPY1.DS01
DELETE DSNAME=MQM2.LOGCOPY1.DS02
DELETE DSNAME=MQM2.LOGCOPY1.DS03
DELETE DSNAME=MQM2.LOGCOPY1.DS04
DELETE DSNAME=MQM2.LOGCOPY2.DS01
DELETE DSNAME=MQM2.LOGCOPY2.DS02
DELETE DSNAME=MQM2.LOGCOPY2.DS03
DELETE DSNAME=MQM2.LOGCOPY2.DS04

* Step 7
NEWLOG DSNAME=MQM2.LOGCOPY1.DS01,COPY1
NEWLOG DSNAME=MQM2.LOGCOPY1.DS02,COPY1
NEWLOG DSNAME=MQM2.LOGCOPY1.DS03,COPY1
NEWLOG DSNAME=MQM2.LOGCOPY1.DS04,COPY1
NEWLOG DSNAME=MQM2.LOGCOPY2.DS01,COPY2
NEWLOG DSNAME=MQM2.LOGCOPY2.DS02,COPY2
NEWLOG DSNAME=MQM2.LOGCOPY2.DS03,COPY2
NEWLOG DSNAME=MQM2.LOGCOPY2.DS04,COPY2

* Step 8
CRESTART CREATE,ENDRBA=063000
```

The things you need to consider for restarting the channel initiator at the recovery site are like those faced when using ARM to restart the channel initiator on a different z/OS image. See “Using ARM in an IBM MQ network” on page 552 for more information. Your recovery strategy should also cover recovery of the IBM MQ product libraries and the application programming environments that use IBM MQ (CICS , for example).

Other functions of the change log inventory utility (CSQJU003) can also be used in disaster recovery scenarios. The HIGHRBA function allows the update of the highest RBA written and highest RBA offloaded values within the bootstrap data set. The CHECKPT function allows the addition of new checkpoint queue records or the deletion of existing checkpoint queue records in the BSDS.

Attention: These functions might affect the integrity of your IBM MQ data. Only use them in disaster recovery scenarios under the guidance of IBM service personnel.

Fast copy techniques

If copies of all the page sets and logs are made while the queue manager is frozen, the copies will be a consistent set that can be used to restart the queue manager at an alternative site. They typically enable a much faster restart of the queue manager, as there is little media recovery to be performed.

Use the SUSPEND QMGR LOG command to freeze the queue manager. This command flushes buffer pools to the page sets, takes a checkpoint, and stops any further log write activity. Once log write activity has been suspended, the queue manager is effectively frozen until you issue a RESUME QMGR LOG command. While the queue manager is frozen, the page sets and logs can be copied.

By using copying tools such as FLASHCOPY or SNAPSHOT to rapidly copy the page sets and logs, the time during which the queue manager is frozen can be reduced to a minimum.

Within a queue sharing group, however, the SUSPEND QMGR LOG command might not be such a good solution. To be effective, the copies of the logs must all contain the same point in time for recovery, which means that the SUSPEND QMGR LOG command must be issued on all queue managers within the queue sharing group simultaneously, and therefore the entire queue sharing group will be frozen for some time.

Recovering a queue sharing group

In the event of a prime site disaster, you can restart a queue sharing group at a remote site using backup data sets from the prime site. To recover a queue sharing group you need to coordinate the recovery across all the queue managers in the queue sharing group, and coordinate with other resources, primarily Db2. This section describes these tasks in detail.

- CF structure media recovery
- Backing up the queue sharing group at the prime site
- Recovering a queue sharing group at the alternative site

CF structure media recovery

Media recovery of a CF structure used to hold persistent messages on a shared queue, relies on having a backup of the media that can be forward recovered by the application of logged updates. Take backups of your CF structures periodically using the MQSC BACKUP CFSTRUCT command. All updates to shared queues (MQGETs and MQPUTs) are written on the log of the queue manager where the update is performed. To perform media recovery of a CF structure you must apply logged updates to that backup from the logs of all the queue managers that have used that CF structure. When you use the MQSC RECOVER CFSTRUCT command, IBM MQ automatically merges the logs from relevant queue managers, and applies the updates to the most recent backup.

The CF structure backup is written to the log of the queue manager that processed the BACKUP CFSTRUCT command, so there are no additional data sets to be collected and transported to the alternative site.

Backing up the queue sharing group at the prime site

At the prime site you need to establish a consistent set of backups on a regular basis, which can be used in the event of a disaster to rebuild the queue sharing group at an alternative site. For a single queue manager, recovery can be to an arbitrary point in time, typically the end of the logs available at the remote site. However, where persistent messages have been stored on a shared queue, the logs of all the queue managers in the queue sharing group must be merged to recover shared queues, as any queue manager in the queue sharing group might have performed updates (MQPUT s or MQGET s) on the queue.

For recovery of a queue sharing group, you need to establish a point in time that is within the log range of the log data of all queue managers. However, as you can only **forward** recover media from the log, this point in time must be after the BACKUP CFSTRUCT command has been issued and after any page set backups have been performed. (Typically, the point in time for recovery might correspond to the end of a business day or week.)

The following diagram shows time lines for two queue managers in a queue sharing group. For each queue manager, fuzzy backups of page sets are taken (see Method 2: Fuzzy backup). On queue manager A, a BACKUP CFSTRUCT command is issued. Subsequently, an ARCHIVE LOG command is issued on each queue manager to truncate the active log, and copy it to media offline from the queue manager, which can be transported to the alternative site. End of log identifies the time at which the ARCHIVE LOG command was issued, and therefore marks the extent of log data typically available at the alternative site. The point in time for recovery must lie between the end of any page set or CF structure backups, and the earliest end of log available at the alternative site.

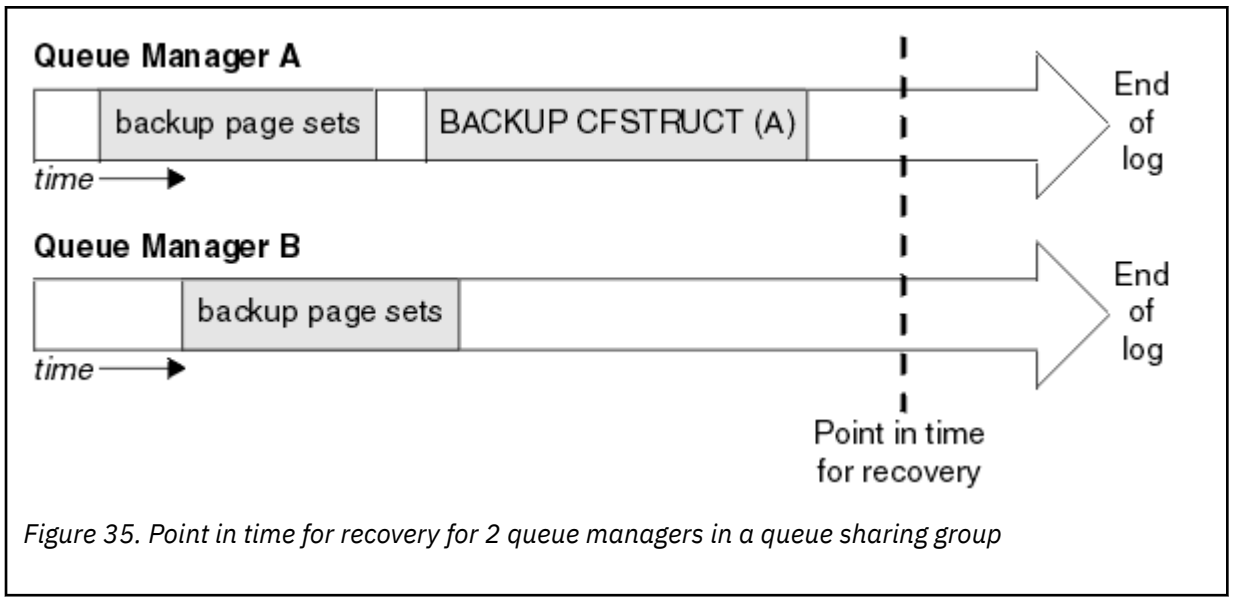


Figure 35. Point in time for recovery for 2 queue managers in a queue sharing group

IBM MQ records information associated with the CF structure backups in a table in Db2. Depending on your requirements, you might want to coordinate the point in time for recovery of IBM MQ with that for Db2, or it might be sufficient to take a copy of the IBM MQ CSQ.ADMIN_B_STRBACKUP table after the BACKUP CFSTRUCT commands have finished.

To prepare for a recovery:

1. Create page set backups for each queue manager in the queue sharing group.
2. Issue a BACKUP CFSTRUCT command for each CF structure with the RECOVER(YES) attribute. You can issue these commands from a single queue manager, or from different queue managers within the queue sharing group to balance the workload.
3. Once all the backups have completed, issue an ARCHIVE LOG command to switch the active log and create copies of the logs and BSDSs of each queue manager in the queue sharing group.
4. Transport the page set backups, the archived logs, the archived BSDS of all the queue managers in the queue sharing group, and your chosen Db2 backup information, off-site.

Recovering a queue sharing group at the alternative site

Before you can recover the queue sharing group, you need to prepare the environment:

1. If you have old information in your coupling facility from practice startups when you installed the queue sharing group, you need to clean this out first:

   Note: If you do not have old information in the coupling facility, you can omit this step.

   a. Enter the following z/OS command to display the CF structures for this queue sharing group:

```
D XCF,STRUCTURE,STRNAME= qsgname
```

b. For all structures that start with the queue sharing group name, use the z/OS command SETXCF FORCE CONNECTION to force the connection off those structures:

```
SETXCF FORCE,CONNECTION,STRNAME= strname,CONNAME=ALL
```

c. Delete all the CF structures using the following command for each structure:

```
SETXCF FORCE,STRUCTURE,STRNAME= strname
```

2. Restore Db2 systems and data-sharing groups.
3. Recover the CSQ.ADMIN_B_STRBACKUP table so that it contains information about the most recent structure backups taken at the prime site.

   Note: It is important that the STRBACKUP table contains the most recent structure backup information. Older structure backup information might require data sets that you have discarded as a result of the information given by a recent DISPLAY USAGE TYPE(DATASET) command, which would mean that your recovered CF structure would not contain accurate information.
4. Run the ADD QMGR command of the CSQ5PQSG utility for every queue manager in the queue sharing group. This will restore the XCF group entry for each queue manager.

   When you run the utility in this scenario, the following messages are normal:

```
CSQU566I Unable to get attributes for admin structure, CF not found
or not allocated
CSQU546E Unable to add QMGR queue_manager_name entry,
already exists in DB2 table CSQ.ADMIN_B_QMGR
CSQU148I CSQ5PQSG Utility completed, return code=4
```

To recover the queue managers in the queue sharing group:

1. Define new page set data sets and load them with the data in the copies of the page sets from the primary site.
2. Define new active log data sets.
3. Define a new BSDS data set and use Access Method Services REPRO to copy the most recent archived BSDS into it.
4. Use the print log map utility CSQJU004 to print information from this most recent BSDS. At the time this BSDS was archived, the most recent archived log you have would have just been truncated as an active log, and does not appear as an archived log. Record the STARTRBA, STARTLRSN, ENDRBA, and ENDLRSN values of this log.
5. Use the change log inventory utility, CSQJU003, to register this latest archive log data set in the BSDS that you have just restored, using the values recorded in Step “4” on page 547.
6. Use the DELETE option of CSQJU003 to remove all active log information from the BSDS.
7. Use the NEWLOG option of CSQJU003 to add active logs to the BSDS, do not specify STARTRBA or ENDRBA.
8. Calculate the *recoverylrsn* for the queue sharing group. The *recoverylrsn* is the lowest of the ENDLRSNs across all queue managers in the queue sharing group (as recorded in Step “4” on page 547), minus 1. For example, if there are two queue managers in the queue sharing group, and the ENDLRSN for one of them is B713 3C72 22C5, and for the other is B713 3D45 2123, the *recoverylrsn* is B713 3C72 22C4.
9. Use CSQJU003 to add a restart control record to the BSDS. Specify:

   ```
   CRESTART CREATE,ENDLRSN= recoverylrsn
   ```

   where *recoverylrsn* is the value you recorded in Step “8” on page 547.

   The BSDS now describes all active logs as being empty, all the archived logs you have available, and no checkpoints beyond the end of your logs.

You must add the CRESTART record to the BSDS for each queue manager within the queue sharing group.

10. Restart each queue manager in the queue sharing group with the START QMGR command. During initialization, an operator reply message such as the following is issued:

    ```
    CSQJ245D +CSQ1 RESTART CONTROL INDICATES TRUNCATION AT RBA highrba.
    REPLY Y TO CONTINUE, N TO CANCEL
    ```

    Reply Y to start the queue manager. The queue manager starts, and recovers data up to ENDRBA specified in the CRESTART statement.

    The first IBM MQ queue manager started can rebuild the admin structure partitions for other members of the queue sharing group as well as its own, and it is no longer necessary to restart each queue manager in the queue sharing group at this stage.

11. When the admin structure data for all queue managers has been rebuilt, issue a RECOVER CFSTRUCT command for each CF application structure.

    If you issue the RECOVER CFSTRUCT command for all structures on a single queue manager, the log merge process is only performed once, so is quicker than issuing the command on a different queue manager for each CF structure, where each queue manager has to perform the log merge step.

When conditional restart processing is used in a queue sharing group, IBM MQ queue managers, performing peer admin rebuild, check that peers BSDS contain the same CRESTART LRSN as their own. This is to ensure the integrity of the rebuilt admin structure. It is therefore important to restart other peers in the QSG, so they can process their own CRESTART information, before the next unconditional restart of any member of the group.

Using disk mirroring

Many installations now use disk mirroring technologies such as IBM Metro Mirror (formerly PPRC) to make synchronous copies of data sets at an alternative site. In such situations, many of the steps detailed become unnecessary as the IBM MQ page sets and logs at the alternative site are effectively identical to those at the prime site. Where such technologies are used, the steps to restart a queue sharing group at an alternative site may be summarized as:

- Clear IBM MQ CF structures at the alternative site. (These often contain residual information from any previous disaster recovery exercise).
- Restore Db2 systems and all tables in the database used by the IBM MQ queue sharing group.
- Restart queue managers. Before IBM WebSphere MQ 7.0.1, it is necessary to restart each queue manager defined in the queue sharing group as each queue manage recovers its own partition of the admin structure during queue manager restart. After each queue manager has been restarted, those not on their home LPAR can be shut down again. The first IBM MQ queue manager started rebuilds the admin structure partitions for other members of the queue sharing group as well as its own, and it is no longer necessary to restart each queue manager in the queue sharing group.
- After the admin structure has been rebuilt, recover the application structures.

IBM MQ for z/OS supports use of zHyperWrite when writing to active logs mirrored using Metro Mirror. zHyperWrite can help reduce the performance impact of using Metro Mirror; see Using Metro Mirror with IBM MQ for more information.

z/OS *Reinitializing a queue manager*

If the queue manager has terminated abnormally you might not be able to restart it. This could be because your page sets or logs have been lost, truncated, or corrupted. If this has happened, you might have to reinitialize the queue manager (perform a cold start).

Attention

Only perform a cold start if you cannot restart the queue manager any other way. Performing a cold start enables you to recover your queue manager and your object definitions; you will **not** be able to recover your message data. Check that none of the other restart scenarios described in this topic work for you before you do this.

When you have restarted, all your IBM MQ objects are defined and available for use, but there is no message data.

Note: Do not reinitialize a queue manager while it is part of a cluster. You must first remove the queue manager from the cluster (using RESET CLUSTER commands on the other queue managers in the cluster), then reinitialize it, and finally reintroduce it to the cluster as a new queue manager.

This is because during reinitialization, the queue manager identifier (QMID) is changed, so any cluster object with the old queue manager identifier must be removed from the cluster.

For further information see the following sections:

- Reinitializing a queue manager that is not in a queue sharing group
- Reinitializing queue managers in a queue sharing group

Reinitializing a queue manager that is not in a queue sharing group

To reinitialize a queue manager, follow this procedure:

1. Prepare the object definition statements that to be used when you restart the queue manager. To do this, either:
   - If page set zero is available, use the CSQUTIL SDEFS function (see Producing a list of IBM MQ define commands). You must get definitions for all object types (authentication information objects, CF structures, channels, namelists, processes, queues, and storage classes).
   - If page set zero is not available, use the definitions from the last time you backed up your object definitions.
2. Redefine your queue manager data sets (do not do this until you have completed step “1” on page 549).

   See creating the bootstrap and log data sets and defining your page sets for more information.
3. Restart the queue manager using the newly defined and initialized log data sets, BSDS, and page sets. Use the object definition input statements that you created in step “1” on page 549 as input in the CSQINP2 initialization input data set.

Reinitializing queue managers in a queue sharing group

In a queue sharing group, reinitializing a queue manager is more complex. It might be necessary to reinitialize one or more queue managers because of page set or log problems, but there might also be problems with Db2 or the coupling facility to deal with. Because of this, there are a number of alternatives:

Cold start
Reinitializing the entire queue sharing group involves forcing all the coupling facilities structures, clearing all object definitions for the queue sharing group from Db2, deleting or redefining the logs and BSDS, and formatting page sets for all the queue managers in the queue sharing group.

Shared definitions retained
Delete or redefine the logs and BSDS, format page sets for all queue managers in the queue sharing group, and force all the coupling facilities structures. On restart, all messages will have been deleted. The queue managers re-create COPY objects that correspond to GROUP objects that still exist in the Db2 database. Any shared queues still exist and can be used.

Single queue manager reinitialized
Delete or redefine the logs and BSDS, and format page sets for the single queue manager (this deletes all its private objects and messages). On restart, the queue manager re-creates COPY objects that correspond to GROUP objects that still exist in the Db2 database. Any shared queues still exist, as do the messages on them, and can be used.

Point in time recovery of a queue sharing group
This is the alternative site disaster recovery scenario.

Shared objects are recovered to the point in time achieved by Db2 recovery (described in A Db2 system fails). Each queue manager can be recovered to a point in time achievable from the backup copies available at the alternative site.

Persistent messages can be used in queue sharing groups, and can be recovered using the MQSC RECOVER CFSTRUCT command. Note that this command recovers to the time of failure. However, there is no recovery of nonpersistent shared queue messages; they are lost unless you have made backup copies independently using the COPY function of the CSQUTIL utility program.

It is not necessary to try to restore each queue manager to the same point in time because there are no interdependencies between the local objects on different queue managers (which are what is actually being recovered), and the queue manager resynchronization with Db2 on restart creates or deletes COPY objects as necessary on a queue manager by queue manager basis.

z/OS Using the z/OS Automatic Restart Manager (ARM)

Use this topic to understand how you can use ARM to automatically restart your queue managers.

This section contains information about the following topics:

- “What is the ARM?” on page 550
- “ARM policies” on page 551
- “Using ARM in an IBM MQ network” on page 552

What is the ARM?

The z/OS Automatic Restart Manager (ARM) is a z/OS recovery function that can improve the availability of your queue managers. When a job or task fails, or the system on which it is running fails, ARM can restart the job or task without operator intervention.

If a queue manager or a channel initiator has failed, ARM restarts it on the same z/OS image. If z/OS, and hence a whole group of related subsystems and applications have failed, ARM can restart all the failed systems automatically, in a predefined order, on another z/OS image within the sysplex. This is called a *cross-system restart*.

Restart the channel initiator by ARM only in exceptional circumstances. If the queue manager is restarted by ARM, restart the channel initiator from the CSQINP2 initialization data set (see “Using ARM in an IBM MQ network” on page 552).

You can use ARM to restart a queue manager on a different z/OS image within the sysplex in the event of z/OS failure. The network implications of IBM MQ ARM restart on a different z/OS image are described in “Using ARM in an IBM MQ network” on page 552.

To enable automatic restart:

- Set up an ARM couple data set.
- Define the automatic restart actions that you want z/OS to perform in an *ARM policy*.
- Start the ARM policy.

Also, IBM MQ must register with ARM at startup (this happens automatically).

Note: If you want to restart queue managers in different z/OS images automatically, you must define every queue manager as a subsystem in each z/OS image on which that queue manager might be restarted, with a sysplex wide unique four character subsystem name.

ARM couple data sets

Ensure that you define the couple data sets required for ARM, and that they are online and active before you start any queue manager for which you want ARM support. IBM MQ automatic ARM registration fails if the couple data sets are not available at queue manager startup. In this situation, IBM MQ assumes that the absence of the couple data set means that you do not want ARM support, and initialization continues.

See *z/OS MVS Setting up a Sysplex* for information about ARM couple data sets.

z/OS *ARM policies*

The Automatic Restart Manager policies are user-defined rules that control ARM functions that can control any restarts of a queue manager.

ARM functions are controlled by a user-defined *ARM policy*. Each z/OS image running a queue manager instance that is to be restarted by ARM must be connected to an ARM couple data set with an active ARM policy.

IBM provides a default ARM policy. You can define new policies, or override the policy defaults by using the *administrative data utility* (IXCMIAPU) provided with z/OS. *z/OS MVS Setting up a Sysplex* describes this utility, and includes full details of how to define an ARM policy.

Figure 36 on page 551 shows an example of an ARM policy. This sample policy restarts any queue manager within a sysplex, if either the queue manager failed, or a whole system failed.

```
//IXCMIAPU EXEC PGM=IXCMIAPU,REGION=2M
//SYSPRINT DD SYSOUT=*
//SYSIN  DD *
DATA TYPE(ARM)
DEFINE POLICY NAME(ARMPOL1) REPLACE(YES)
RESTART_GROUP(DEFAULT)
ELEMENT(*)
RESTART_ATTEMPTS(0)  /* Jobs not to be restarted by ARM  */
RESTART_GROUP(GROUP1)
ELEMENT(SYSMQMGRMQ*)  /* These jobs to be restarted by ARM */
/*
```

Figure 36. Sample ARM policy

For more information see:

- Defining an ARM policy
- Activating an ARM policy
- Registering with ARM

Defining an ARM policy

Set up your ARM policy as follows:

- Define RESTART_GROUPs for each queue manager instance that also contain any CICS or IMS subsystems that connect to that queue manager instance. If you use a subsystem naming convention, you might be able to use the '?' and '*' wild-card characters in your element names to define RESTART_GROUPs with minimum definition effort.
- Specify TERMTYPE(ELEMTERM) for your channel initiators to indicate that they will be restarted only if the channel initiator has failed and the z/OS image has not failed.

- Specify TERMTYPE(ALLTERM) for your queue managers to indicate that they will be restarted if either the queue manager has failed or the z/OS image has failed.
- Specify RESTART_METHOD(BOTH, PERSIST) for both queue managers and channel initiators. This tells ARM to restart using the JCL it saved (after resolution of system symbols) during the last startup. It tells ARM to do this irrespective of whether the individual element failed, or the z/OS image failed.
- Accept the default values for all the other ARM policy options.

Activating an ARM policy

To start your automatic restart management policy, issue the following z/OS command:

```
SETXCF START,POLICY,TYPE=ARM,POLNAME= mypol
```

When the policy is started, all systems connected to the ARM couple data set use the same active policy.

Use the SETXCF STOP command to disable automatic restarts.

Registering with ARM

IBM MQ registers automatically as an *ARM element* during queue manager startup (subject to ARM availability). It deregisters during its shutdown phase, unless requested not to.

At startup, the queue manager determines whether ARM is available. If it is, IBM MQ registers using the name SYSMQMGR *ssid*, where *ssid* is the four character queue manager name, and SYSMQMGR is the element type.

The STOP QMGR MODE(QUIESCE) and STOP QMGR MODE(FORCE) commands deregister the queue manager from ARM (if it was registered with ARM at startup). This prevents ARM restarting this queue manager. The STOP QMGR MODE(RESTART) command does not deregister the queue manager from ARM, so it is eligible for immediate automatic restart.

Each channel initiator address space determines whether ARM is available, and if so registers with the element name SYSMQCH *ssid*, where *ssid* is the queue manager name, and SYSMQCH is the element type.

The channel initiator is always deregistered from ARM when it stops normally, and remains registered only if it ends abnormally. The channel initiator is always deregistered if the queue manager fails.

z/OS *Using ARM in an IBM MQ network*

You can set up your queue manager so that the channel initiators and associated listeners are started automatically when the queue manager is restarted.

To ensure fully automatic queue manager restart on the same z/OS image for both LU 6.2 and TCP/IP communication protocols:

- Start your listeners automatically by adding the appropriate START LISTENER command to the CSQINPX data set.
- Start your channel initiator automatically by adding the appropriate START CHINIT command to the CSQINP2 data set.

For restarting a queue manager with TCP/IP or LU6.2, see

- “Restarting on a different z/OS image with TCP/IP” on page 553
- “Restarting on a different z/OS image with LU 6.2” on page 554

See Task 13: Customize the initialization input data sets for information about the CSQINP2 and CSQINPX data sets.

Restarting on a different z/OS image with TCP/IP

If you are using TCP/IP as your communication protocol, and you are using virtual IP addresses, you can configure these to recover on other z/OS images, allowing channels connecting to that queue manager to reconnect without any changes. Otherwise, you can reallocate a TCP/IP address after moving a queue manager to a different z/OS image only if you are using clusters or if you are connecting to a queue sharing group using a WLM dynamic Domain Name System (DNS) logical group name.

- When using clustering
- When connecting to a queue sharing group

When using clustering

z/OS ARM responds to a system failure by restarting the queue manager on a different z/OS image in the same sysplex; this system has a different TCP/IP address to the original z/OS image. The following explains how you can use IBM MQ clusters to reassign a queue manager's TCP/IP address after it has been moved by ARM restart to a different z/OS image.

When a client queue manager detects the queue manager failure (as a channel failure), it responds by reallocating suitable messages on its cluster transmission queue to a different server queue manager that hosts a different instance of the target cluster queue. However, it cannot reallocate messages that are bound to the original server by affinity constraints, or messages that are in doubt because the server queue manager failed during end-of-batch processing. To process these messages, do the following:

1. Allocate a different cluster-receiver channel name and a different TCP/IP port to each z/OS queue manager. Each queue manager needs a different port so that two systems can share a single TCP/IP stack on a z/OS image. One of these is the queue manager originally running on that z/OS image, and the other is the queue manager that ARM will restart on that z/OS image following a system failure. Configure each port on each z/OS image, so that ARM can restart any queue manager on any z/OS image.
2. Create a different channel initiator command input file (CSQINPX) for each queue manager and z/OS image combination, to be referenced during channel initiator startup.

   Each CSQINPX file must include a START LISTENER PORT(port) command specific to that queue manager, and an ALTER CHANNEL command for a cluster-receiver channel specific to that queue manager and z/OS image combination. The ALTER CHANNEL command needs to set the connection name to the TCP/IP name of the z/OS image on which it is restarted. It must include the port number specific to the restarted queue manager as part of the connection name.

   The start-up JCL of each queue manager can have a fixed data set name for this CSQINPX file, and each z/OS image must have a different version of each CSQINPX file on a non-shared DASD volume.

If an ARM restart occurs, IBM MQ advertises the changed channel definition to the cluster repository, which in turn publishes it to all the client queue managers that have expressed an interest in the server queue manager.

The client queue manager treats the server queue manager failure as a channel failure, and tries to restart the failed channel. When the client queue manager learns the new server connection-name, the channel restart reconnects the client queue manager to the restarted server queue manager. The client queue manager can then resynchronize its messages, resolve any in-doubt messages on the client queue manager's transmission queue, and normal processing can continue.

When connecting to a queue sharing group

When connecting to a queue sharing group through a TCP/IP dynamic Domain Name System (DNS) logical group name, the connection name in your channel definition specifies the logical group name of your queue sharing group, not the host name or IP address of a physical machine. When this channel starts, it connects to the dynamic DNS and is then connected to one of the queue managers in the queue sharing group. This process is explained in Setting up communication for IBM MQ for z/OS using queue sharing groups.

In the unlikely event of an image failure, one of the following occurs:

- The queue managers on the failing image de-register from the dynamic DNS running on your sysplex. The channel responds to the connection failure by entering RETRYING state and then connects to the dynamic DNS running on the sysplex. The dynamic DNS allocates the inbound request to one of the remaining members of the queue sharing group that is still running on the remaining images.
- If no other queue manager in the queue sharing group is active and ARM restarts the queue manager and channel initiator on a different image, the group listener registers with dynamic DNS from this new image. This means that the logical group name (from the connection name field of the channel) connects to the dynamic DNS and is then connected to the same queue manager, now running on a different image. No change was required to the channel definition.

For this type of recovery to occur, the following points must be noted:

- On z/OS, the dynamic DNS runs on one of the z/OS images in the sysplex. If this image were to fail, the dynamic DNS needs to be configured so that there is a secondary name server active in the sysplex, acting as an alternative to the primary name server. Information about primary and secondary dynamic DNS servers can be found in the *OS/390® SecureWay CS IP Configuration* manual
- The TCP/IP group listener might have been started on a particular IP address that might not be available on this z/OS image. If so, the listener might need to be started on a different IP address on the new image. If you are using virtual IP addresses, you can configure these to recover on other z/OS images so that no change to the START LISTENER command is required.

Restarting on a different z/OS image with LU 6.2

If you use only LU 6.2 communication protocols, carry out the following procedure to enable network reconnect after automatic restart of a queue manager on a different z/OS image within the sysplex:

- Define each queue manager within the sysplex with a unique subsystem name.
- Define each channel initiator within the sysplex with a unique LUNAME. This is specified in both the queue manager attributes and in the START LISTENER command.

  Note: The LUNAME names an entry in the APPC side table, which in turn maps this to the actual LUNAME.
- Set up a shared APPC side table, which is referenced by each z/OS image within the sysplex. This should contain an entry for each channel initiator's LUNAME. See *z/OS MVS Planning: APPC/MVS Management* for information about this.
- Set up an APPCPM *xx* member of SYS1.PARMLIB for each channel initiator within the sysplex to contain an LUADD to activate the APPC side table entry for that channel initiator. These members should be shared by each z/OS image. The appropriate SYS1.PARMLIB member is activated by a z/OS command SET APPC= *xx*, which is issued automatically during ARM restart of the queue manager (and its channel initiator) on a different z/OS image, as described in the following text.
- Use the LU62ARM queue manager attribute to specify the *xx* suffix of this SYS1.PARMLIB member for each channel initiator. This causes the channel initiator to issue the required z/OS command SET APPC= *xx* to activate its LUNAME.

Define your ARM policy so that it restarts the channel initiator only if it fails while its z/OS image stays up; the user ID associated with the XCFAS address space must be authorized to issue the IBM MQ command START CHINIT. Do not restart the channel initiator automatically if its z/OS image also fails, instead use commands in the CSQINP2 and CSQINPX data sets to start the channel initiator and listeners.

z/OS Recovering units of work manually

You can manually recover units of work CICS, IMS, RRS, or other queue managers in a queue sharing group. You can use queue manager commands to display the status of the units of work associated with each connection to the queue manager.

This topic contains information about the following subjects:

Displaying connections and threads

You can use the DISPLAY CONN command to get information about connections to queue managers and their associated units of work. You can display active units of work to see what is currently happening, or to see what needs to be terminated to allow the queue manager to shut down, and you can display unresolved units of work to help with recovery.

Active units of work

To display only active units of work, use

```
DISPLAY CONN(*) WHERE(UOWSTATE EQ ACTIVE)
```

Unresolved units of work

An unresolved unit of work, also known as an "in-doubt thread", is one that is in the second pass of the two-phase commit operation. Resources are held in IBM MQ on its behalf. To display unresolved units of work, use

```
DISPLAY CONN(*) WHERE(UOWSTATE EQ UNRESOLVED)
```

External intervention is needed to resolve the status of unresolved units of work. This might only involve starting the recovery coordinator (CICS, IMS, or RRS) or might involve more, as described in the following sections.

z/OS *Recovering CICS units of recovery manually*

Use this topic to understand what happens when the CICS adapter restarts, and then explains how to deal with any unresolved units of recovery that arise.

What happens when the CICS adapter restarts

Whenever a connection is broken, the adapter has to go through a *restart phase* during the *reconnect process*. The restart phase resynchronizes resources. Resynchronization between CICS and IBM MQ enables in-doubt units of work to be identified and resolved.

Resynchronization can be caused by:

- An explicit request from the distributed queuing component
- An implicit request when a connection is made to IBM MQ

If the resynchronization is caused by connecting to IBM MQ, the sequence of events is:

1. The connection process retrieves a list of in-doubt units of work (UOW) IDs from IBM MQ.
2. The UOW IDs are displayed on the console in CSQC313I messages.
3. The UOW IDs are passed to CICS.
4. CICS initiates a resynchronization task (CRSY) for each in-doubt UOW ID.

5. The result of the task for each in-doubt UOW is displayed on the console.

You need to check the messages that are displayed during the connect process:

CSQC313I
Shows that a UOW is in doubt.

CSQC400I
Identifies the UOW and is followed by one of these messages:

- CSQC402I or CSQC403I shows that the UOW was resolved successfully (committed or backed out).
- CSQC404E, CSQC405E, CSQC406E, or CSQC407E shows that the UOW was not resolved.

CSQC409I
Shows that all UOWs were resolved successfully.

CSQC408I
Shows that not all UOWs were resolved successfully.

CSQC314I
Warns that UOW IDs highlighted with a * are not resolved automatically. These UOWs must be resolved explicitly by the distributed queuing component when it is restarted.

Figure 37 on page 556 shows an example set of restart messages displayed on the z/OS console.

```
CSQ9022I +CSQ1 CSQYASCP ' START QMGR' NORMAL COMPLETION
+CSQC323I VICIC1 CSQCQCON CONNECT received from TERMID=PB62 TRANID=CKCN
+CSQC303I VICIC1 CSQCCON CSQCSERV loaded. Entry point is 850E8918
+CSQC313I VICIC1 CSQCCON UOWID=VICIC1.A6E5A6F0E2178D25 is in doubt
+CSQC313I VICIC1 CSQCCON UOWID=VICIC1.A6E5A6F055B2AC25 is in doubt
+CSQC313I VICIC1 CSQCCON UOWID=VICIC1.A6E5A6EFFD60D425 is in doubt
+CSQC313I VICIC1 CSQCCON UOWID=VICIC1.A6E5A6F07AB56D22 is in doubt
+CSQC307I VICIC1 CSQCCON Successful connection to subsystem VC2
+CSQC472I VICIC1 CSQCSERV Server subtask (TCB address=008BAD18) connect
successful
+CSQC472I VICIC1 CSQCSERV Server subtask (TCB address=008BAA10) connect
successful
+CSQC472I VICIC1 CSQCSERV Server subtask (TCB address=008BA708) connect
successful
+CSQC472I VICIC1 CSQCSERV Server subtask (TCB address=008CAE88) connect
successful
+CSQC472I VICIC1 CSQCSERV Server subtask (TCB address=008CAB80) connect
successful
+CSQC472I VICIC1 CSQCSERV Server subtask (TCB address=008CA878) connect
successful
+CSQC472I VICIC1 CSQCSERV Server subtask (TCB address=008CA570) connect
successful
+CSQC472I VICIC1 CSQCSERV Server subtask (TCB address=008CA268) connect
successful
+CSQC403I VICIC1 CSQCTRUE Resolved BACKOUT for
+CSQC400I VICIC1 CSQCTRUE UOWID=VICIC1.A6E5A6F0E2178D25
+CSQC403I VICIC1 CSQCTRUE Resolved BACKOUT for
+CSQC400I VICIC1 CSQCTRUE UOWID=VICIC1.A6E5A6F055B2AC25
+CSQC403I VICIC1 CSQCTRUE Resolved BACKOUT for
+CSQC400I VICIC1 CSQCTRUE UOWID=VICIC1.A6E5A6F07AB56D22
+CSQC403I VICIC1 CSQCTRUE Resolved BACKOUT for
+CSQC400I VICIC1 CSQCTRUE UOWID=VICIC1.A6E5A6EFFD60D425
+CSQC409I VICIC1 CSQCTRUE Resynchronization completed successfully
```

Figure 37. Example restart messages

The total number of CSQC313I messages should equal the total number of CSQC402I plus CSQC403I messages. If the totals are not equal, there are UOWs that the connection process cannot resolve. Those UOWs that cannot be resolved are caused by problems with CICS (for example, a cold start) or with IBM MQ, or by distributing queuing. When these problems have been fixed, you can initiate another resynchronization by disconnecting and then reconnecting.

Alternatively, you can resolve each outstanding UOW yourself using the RESOLVE INDOUBT command and the UOW ID shown in message CSQC400I. You must then initiate a disconnect and a connect to clean

up the *unit of recovery descriptors* in CICS. You need to know the correct outcome of the UOW to resolve UOWs manually.

All messages that are associated with unresolved UOWs are locked by IBM MQ and no Batch, TSO, or CICS task can access them.

If CICS fails and an emergency restart is necessary, *do not* vary the GENERIC APPLID of the CICS system. If you do and then reconnect to IBM MQ, data integrity with IBM MQ cannot be guaranteed. This is because IBM MQ treats the new instance of CICS as a different CICS (because the APPLID is different). In-doubt resolution is then based on the wrong CICS log.

How to resolve CICS units of recovery manually

If the adapter ends abnormally, CICS and IBM MQ build in-doubt lists either dynamically or during restart, depending on which subsystem caused the abend.

Note: If you use the DFH$INDB sample program to show units of work, you might find that it does not always show IBM MQ UOWs correctly.

When CICS connects to IBM MQ, there might be one or more units of recovery that have not been resolved.

One of the following messages is sent to the console:

- CSQC404E
- CSQC405E
- CSQC406E
- CSQC407E
- CSQC408I

For details of what these messages mean, see the CICS adapter and Bridge messages messages.

CICS retains details of units of recovery that were not resolved during connection startup. An entry is purged when it no longer appears on the list presented by IBM MQ.

Any units of recovery that CICS cannot resolve must be resolved manually using IBM MQ commands. This manual procedure is rarely used within an installation, because it is required only where operational errors or software problems have prevented automatic resolution. *Any inconsistencies found during in-doubt resolution must be investigated.*

To resolve the units of recovery:

1. Obtain a list of the units of recovery from IBM MQ using the following command:

```
+CSQ1  DISPLAY CONN( * ) WHERE(UOWSTATE EQ UNRESOLVED)
```

   You receive the following message:

```
CSQM201I +CSQ1 CSQMDRTC DISPLAY CONN DETAILS
CONN(BC85772CBE3E0001)
EXTCONN(C3E2D8C3C7D9F0F94040404040404040)
TYPE(CONN)
CONNOPTS(
MQCNO_STANDARD_BINDING
)
UOWLOGDA(2005-02-04)
UOWLOGTI(10.17.44)
UOWSTDA(2005-02-04)
UOWSTTI(10.17.44)
UOWSTATE(UNRESOLVED)
NID(IYRCSQ1 .BC8571519B60222D)
EXTURID(BC8571519B60222D)
QMURID(0000002BDA50)
URTYPE(CICS)
USERID(MQTEST)
APPLTAG(IYRCSQ1)
ASID(0000)
APPLTYPE(CICS)
TRANSID(GP02)
TASKNO(0000096)
END CONN DETAILS
```

For CICS connections, the NID consists of the CICS applid and a unique number provided by CICS at the time the syncpoint log entries are written. This unique number is stored in records written to both the CICS system log and the IBM MQ log at syncpoint processing time. This value is referred to in CICS as the *recovery token*.

2. Scan the CICS log for entries related to a particular unit of recovery.

   Look for a PREPARE record for the task-related installation where the recovery token field (JCSRMTKN) equals the value obtained from the network ID. The network ID is supplied by IBM MQ in the DISPLAY CONN command output.

   The PREPARE record in the CICS log for the units of recovery provides the CICS task number. All other entries on the log for this CICS task can be located using this number.

   You can use the CICS journal print utility DFHJUP when scanning the log. For details of using this program, see the *CICS Operations and Utilities Guide*.

3. Scan the IBM MQ log for records with the NID related to a particular unit of recovery. Then use the URID from this record to obtain the rest of the log records for this unit of recovery.

   When scanning the IBM MQ log, note that the IBM MQ startup message CSQJ001I provides the start RBA for this session.

   The print log records program (CSQ1LOGP) can be used for that purpose.

4. If you need to, do in-doubt resolution in IBM MQ.

   IBM MQ can be directed to take the recovery action for a unit of recovery using an IBM MQ RESOLVE INDOUBT command.

   To recover all threads associated with a specific *connection-name*, use the NID(*) option.

   The command produces one of the following messages showing whether the thread is committed or backed out:

```
CSQV414I +CSQ1 THREAD network-id COMMIT SCHEDULED
CSQV415I +CSQ1 THREAD network-id ABORT SCHEDULED
```

When performing in-doubt resolution, CICS and the adapter are not aware of the commands to IBM MQ to commit or back out units of recovery, because only IBM MQ resources are affected. However, CICS keeps details about the in-doubt threads that could not be resolved by IBM MQ. This information is purged

either when the list presented is empty, or when the list does not include a unit of recovery of which CICS has details.

z/OS *Recovering IMS units of recovery manually*

Use this topic to understand what happens when the IMS adapter restarts, and then explains how to deal with any unresolved units of recovery that arise.

What happens when the IMS adapter restarts

Whenever the connection to IBM MQ is restarted, either following a queue manager restart or an IMS / START SUBSYS command, IMS initiates the following resynchronization process:

1. IMS presents the list of unit of work (UOW) IDs that it believes are in doubt to the IBM MQ IMS adapter one at a time with a resolution parameter of Commit or Backout.
2. The IMS adapter passes the resolution request to IBM MQ and reports the result back to IMS.
3. Having processed all the IMS resolution requests, the IMS adapter gets from IBM MQ a list of all UOWs that IBM MQ still holds in doubt that were initiated by the IMS system. These are reported to the IMS master terminal in message CSQQ008I.

Note: While a UOW is in doubt, any associated IBM MQ message is locked by IBM MQ and is not available to any application.

How to resolve IMS units of recovery manually

When IMS connects to IBM MQ, IBM MQ might have one, or more in-doubt units of recovery that have not been resolved.

If IBM MQ has in-doubt units of recovery that IMS did not resolve, the following message is issued at the IMS master terminal:

```
CSQQ008I nn units of recovery are still in doubt in queue manager qmgr-name
```

If this message is issued, IMS was either cold-started or it was started with an incomplete log tape. This message can also be issued if IBM MQ or IMS terminates abnormally because of a software error or other subsystem failure.

After receiving the CSQQ008I message:

- The connection remains active.
- IMS applications can still access IBM MQ resources.
- Some IBM MQ resources remain locked out.

If the in-doubt thread is not resolved, IMS message queues can start to build up. If the IMS queues fill to capacity, IMS terminates. You must be aware of this potential difficulty, and you must monitor IMS until the in-doubt units of recovery are fully resolved.

Recovery procedure

Use the following procedure to recover the IMS units of work:

1. Force the IMS log closed, using /SWI OLDS, and then archive the IMS log. Use the utility, DFSERA10, to print the records from the previous IMS log tape. Type X'3730' log records indicate a phase-2 commit request and type X'38' log records indicate an abort request. Record the requested action for the last transaction in each dependent region.
2. Run the DL/I batch job to back out each PSB involved that has not reached a commit point. The process might take some time because transactions are still being processed. It might also lock

up a number of records, which could affect the rest of the processing and the rest of the message queues.

3. Produce a list of the in-doubt units of recovery from IBM MQ using the following command:

```
+CSQ1 DISPLAY CONN(*) WHERE(UOWSTATE EQ UNRESOLVED)
```

You receive the following message:

```
CSQM201I +CSQ1 CSQMDRTC DISPLAY CONN DETAILS
CONN(BC45A794C4290001)
EXTCONN(C3E2D8C3E2C5C3F24040404040404040)
TYPE(CONN)
CONNOPTS(
MQCNO_STANDARD_BINDING
)
UOWLOGDA(2005-02-15)
UOWLOGTI(16.39.43)
UOWSTDA(2005-02-15)
UOWSTTI(16.39.43)
UOWSTATE(UNRESOLVED)
NID(IM8F  .BC45A794D3810344)
EXTURID(
0000052900000000
)
QMURID(00000354B76E)
URTYPE(IMS)
USERID(STCPI)
APPLTAG(IM8F)
ASID(0000)
APPLTYPE(IMS)
PSTID(0004)
PSBNAME(GP01MPP)
```

For IMS, the NID consists of the IMS connection name and a unique number provided by IMS. The value is referred to in IMS as the *recovery token*. For more information, see the IMS documentation.

4. Compare the NIDs (IMSID plus OASN in hexadecimal) displayed in the DISPLAY THREAD messages with the OASNs (4 bytes decimal) shown in the DFSERA10 output. Decide whether to commit or back out.
5. Perform in-doubt resolution in IBM MQ with the RESOLVE INDOUBT command, as follows:

```
RESOLVE INDOUBT( connection-name )
ACTION(COMMIT|BACKOUT)
NID( network-id )
```

To recover all threads associated with *connection-name*, use the NID(*) option. The command results in one of the following messages to indicate whether the thread is committed or backed out:

```
CSQV414I  THREAD network-id COMMIT SCHEDULED
CSQV415I  THREAD network-id BACKOUT SCHEDULED
```

When performing in-doubt resolution, IMS and the adapter are not aware of the commands to IBM MQ to commit or back out in-doubt units of recovery because only IBM MQ resources are affected.

z/OS *Recovering RRS units of recovery manually*

Use this topic to understand the how to determine if there are in-doubt RRS units of recovery, and how to manually resolve those units of recovery.

When RRS connects to IBM MQ, IBM MQ might have one, or more in-doubt units of recovery that have not been resolved. If IBM MQ has in-doubt units of recovery that RRS did not resolve, one of the following messages is issued at the z/OS console:

- CSQ3011I
- CSQ3013I

- CSQ3014I
- CSQ3016I

Both IBM MQ and RRS provide tools to display information about in-doubt units of recovery, and techniques for manually resolving them.

In IBM MQ, use the DISPLAY CONN command to display information about in-doubt IBM MQ threads. The output from the command includes RRS unit of recovery IDs for those IBM MQ threads that have RRS as a coordinator. This can be used to determine the outcome of the unit of recovery.

Use the RESOLVE INDOUBT command to resolve the IBM MQ in-doubt thread manually. This command can be used to either commit or back out the unit of recovery after you have determined what the correct decision is.

z/OS *Recovering units of recovery on another queue manager in the queue sharing group*

Use this topic to identify, and manually recover units of recovery on other queue managers in a queue sharing group.

If a queue manager that is a member of a queue sharing group fails and cannot be restarted, other queue managers in the group can perform peer recovery, and take over from it. However, the queue manager might have in-doubt units of recovery that cannot be resolved by peer recovery because the final disposition of that unit of recovery is known only to the failed queue manager. These units of recovery are resolved when the queue manager is eventually restarted, but until then, they remain in doubt.

This means that certain resources (for example, messages) might be locked, making them unavailable to other queue managers in the group. In this situation, you can use the DISPLAY THREAD command to display these units of work on the inactive queue manager. If you want to resolve these units of recovery manually to make the messages available to other queue managers in the group, you can use the RESOLVE INDOUBT command.

When you issue the DISPLAY THREAD command to display units of recovery that are in doubt, you can use the QMNAME keyword to specify the name of the inactive queue manager. For example, if you issue the following command:

```
+CSQ1 DISPLAY THREAD(*) TYPE(INDOUBT) QMNAME(QM01)
```

You receive the following messages:

```
CSQV436I +CSQ1 INDOUBT THREADS FOR QM01 -
NAME   THREAD-XREF    URID NID
USER1  00000000000000000000000000 CSQ:0001.0
USER2  00000000000000000000000000 CSQ:0002.0
DISPLAY THREAD REPORT COMPLETE
```

If the queue manager specified is active, IBM MQ does not return information about in-doubt threads, but issues the following message:

```
CSQV435I CANNOT USE QMNAME KEYWORD, QM01 IS ACTIVE
```

Use the IBM MQ command RESOLVE INDOUBT to resolve the in-doubt threads manually. Use the QMNAME keyword to specify the name of the inactive queue manager in the command.

This command can be used to commit or back out the unit of recovery. The command resolves the shared portion of the unit of recovery only; any local messages are unaffected and remain locked until the queue manager restarts, or reconnects to CICS, IMS, or RRS batch.

z/OS IBM MQ and IMS

IBM MQ provides two components to interface with IMS, the IBM MQ - IMS adapter, and the IBM MQ - IMS bridge. These components are commonly called the IMS adapter, and the IMS bridge.

z/OS Operating the IMS adapter

Use this topic to understand how to operate the IMS adapter, which connects IBM MQ to IMS systems.

Note: The IMS adapter does not incorporate any operations and control panels.

This topic contains the following sections:

z/OS *Controlling IMS connections*

Use this topic to understand the IMS operator commands which control and monitor the connection to IBM MQ.

IMS provides the following operator commands to control and monitor the connection to IBM MQ:

/CHANGE SUBSYS
Deletes an in-doubt unit of recovery from IMS.

/DISPLAY OASN SUBSYS
Displays outstanding recovery elements.

/DISPLAY SUBSYS
Displays connection status and thread activity.

/START SUBSYS
Connects the IMS control region to a queue manager.

/STOP SUBSYS
Disconnects IMS from a queue manager.

/TRACE
Controls the IMS trace.

For more information about these commands, see the *IMS/ESA® Operator's Reference* manual for the level of IMS that you are using.

IMS command responses are sent to the terminal from which the command was issued. Authorization to issue IMS commands is based on IMS security.

z/OS *Connecting from the IMS control region*

Use this topic to understand the mechanisms available to connect from IMS to IBM MQ.

IMS makes one connection from its control region to each queue manager that uses IMS. IMS must be enabled to make the connection in one of these ways:

- Automatically during either:
  - A cold start initialization.
  - A warm start of IMS, if the IBM MQ connection was active when IMS was shut down.
- In response to the IMS command:

```
/START SUBSYS sysid
```

where *sysid* is the queue manager name.

The command can be issued regardless of whether the queue manager is active.

The connection is not made until the first IBM MQ API call to the queue manager is made. Until that time, the IMS command /DIS SUBSYS shows the status as 'NOT CONN'.

The order in which you start IMS and the queue manager is not significant.

IMS cannot re-enable the connection to the queue manager automatically if the queue manager is stopped with a STOP QMGR command, the IMS command /STOP SUBSYS, or an abnormal end. Therefore, you must make the connection by using the IMS command /START SUBSYS.

If an IMS command is seen in the queue manager console log similar to this:

```
MODIFY IMS*,SS*
```

check the IMS master log and ensure that IBM MQ has RACF authority to issue IMS Adapter MODIFY commands.

Initializing the adapter and connecting to the queue manager

The adapter is a set of modules loaded into the IMS control and dependent regions, using the IMS external Subsystem Attach Facility.

This procedure initializes the adapter and connects to the queue manager:

1. Read the subsystem member (SSM) from IMS.PROCLIB. The SSM chosen is an IMS EXEC parameter. There is one entry in the member for each queue manager to which IMS can connect. Each entry contains control information about an IBM MQ adapter.
2. Load the IMS adapter.

   Note: IMS loads one copy of the adapter modules for each IBM MQ instance that is defined in the SSM member.
3. Attach the external subsystem task for IBM MQ.
4. Run the adapter with the CTL EXEC parameter (IMSID) as the connection name.

The process is the same whether the connection is part of initialization or a result of the IMS command / START SUBSYS.

If the queue manager is active when IMS tries to make the connection, the following messages are sent:

- to the z/OS console:

```
DFS3613I ESS TCB INITIALIZATION COMPLETE
```

- to the IMS master terminal:

```
CSQQ000I IMS/TM imsid connected to queue manager ssnm
```

When IMS tries to make the connection and *the queue manager is not active*, the following messages are sent to the IMS master terminal each time an application makes an MQI call:

```
CSQQ001I IMS/TM imsid not connected to queue manager ssnm.
Notify message accepted
DFS3607I MQM1   SUBSYSTEM ID EXIT FAILURE, FC = 0286, RC = 08,
JOBNAME = IMSEMPR1
```

If you get DFS3607I messages when you start the connection to IMS or on system startup, this indicates that the queue manager is not available. To prevent a large number of messages being generated, you must do one of the following:

1. Start the relevant queue manager.
2. Issue the IMS command:

   ```
   /STOP SUBSYS
   ```

   so that IMS does not expect to connect to the queue manager.

If you do neither, a DFS3607I message and the associated CSQQ001I message are issued each time a job is scheduled in the region and each time a connection request to the queue manager is made by an application.

Thread attachment

In an MPP or IFP region, IMS makes a thread connection when the first application program is scheduled into that region, even if that application program does not make an IBM MQ call. In a BMP region, the thread connection is made when the application makes its first IBM MQ call (`MQCONN` or `MQCONNX`). This thread is retained for the duration of the region or until the connection is stopped.

For both the message driven and non-message driven regions, the recovery thread cross-reference identifier, *Thread-xref*, associated with the thread is:

```
PSTid + PSBname
```

where:

PSTid
Partition specification table region identifier

PSBname
Program specification block name

You can use connection IDs as unique identifiers in IBM MQ commands, in which case IBM MQ automatically inserts these IDs into any operator message that it generates.

z/OS *Displaying in-doubt units of recovery*

You can display in-doubt of units of recovery and attempt to recover them.

The operational steps used to list and recover in-doubt units of recovery in this topic are for relatively simple cases only. If the queue manager ends abnormally while connected to IMS, IMS might commit or back out work without IBM MQ being aware of it. When the queue manager restarts, that work is termed *in doubt*. A decision must be made about the status of the work.

To display a list of in-doubt units of recovery, issue the command:

```
+CSQ1  DISPLAY CONN(*) WHERE(UOWSTATE EQ UNRESOLVED)
```

IBM MQ responds with a message like the following:

```
CSQM201I +CSQ1 CSQMDRTC DIS CONN DETAILS
CONN(BC0F6125F5A30001)
EXTCONN(C3E2D8C3C3E2D8F14040404040404040)
TYPE(CONN)
CONNOPTS(
MQCNO_STANDARD_BINDING
)
UOWLOGDA(2004-11-02)
UOWLOGTI(12.27.58)
UOWSTDA(2004-11-02)
UOWSTTI(12.27.58)
UOWSTATE(UNRESOLVED)
NID(CSQ1CHIN.BC0F5F1C86FC0766)
EXTURID(000000000000001F000000007472616E5F6964547565204E6F762020...)
QMURID(000000026232)
URTYPE(XA)
USERID( )
APPLTAG(CSQ1CHIN)
ASID(0000)
APPLTYPE(CHINIT)
CHANNEL( )
CONNAME( )
END CONN DETAILS
```

For an explanation of the attributes in this message, see the description of the DISPLAY CONN command.

Recovering in-doubt units of recovery

To recover in-doubt units of recovery, issue this command:

```
+CSQ1  RESOLVE INDOUBT( connection-name ) ACTION(COMMIT|BACKOUT)
NID( net-node.number )
```

where:

connection-name
The IMS system ID.

ACTION
Indicates whether to commit (COMMIT) or back out (BACKOUT) this unit of recovery.

net-node.number
The associated net-node.number.

When you have issued the RESOLVE INDOUBT command, one of the following messages is displayed:

```
CSQV414I +CSQ1 THREAD network-id COMMIT SCHEDULED

CSQV415I +CSQ1 THREAD network-id BACKOUT SCHEDULED
```

Resolving residual recovery entries

At given times, IMS builds a list of residual recovery entries (RREs). RREs are units of recovery about which IBM MQ might be in doubt. They arise in several situations:

- If the queue manager is not active, IMS has RREs that cannot be resolved until the queue manager is active. These RREs are not a problem.
- If the queue manager is active and connected to IMS, and if IMS backs out the work that IBM MQ has committed, the IMS adapter issues message CSQQ010E. If the data in the two systems must be consistent, there is a problem. For information about resolving this problem, see "Recovering IMS units of recovery manually" on page 559.
- If the queue manager is active and connected to IMS, there might still be RREs even though no messages have informed you of this problem. After the IBM MQ connection to IMS has been established, you can issue the following IMS command to find out if there is a problem:

```
/DISPLAY OASN SUBSYS sysid
```

To purge the RRE, issue one of the following IMS commands:

```
/CHANGE SUBSYS sysid RESET
/CHANGE SUBSYS sysid RESET OASN nnnn
```

where *nnnn* is the originating application sequence number listed in response to your +CSQ1 DISPLAY command. This is the schedule number of the program instance, giving its place in the sequence of invocations of that program since the last IMS cold start. IMS cannot have two in-doubt units of recovery with the same schedule number.

These commands reset the status of IMS ; they do not result in any communication with IBM MQ.

z/OS *Controlling IMS dependent region connections*

You can control, monitor, and, when necessary, terminate connections between IMS and IBM MQ.

Controlling IMS dependent region connections involves the following activities:

- Connecting from dependent regions
- Region error options
- Monitoring the activity on connections
- Disconnecting from dependent regions

Connecting from dependent regions

The IMS adapter used in the control region is also loaded into dependent regions. A connection is made from each dependent region to IBM MQ. This connection is used to coordinate the commitment of IBM MQ and IMS work. To initialize and make the connection, IMS does the following:

1. Reads the subsystem member (SSM) from IMS.PROCLIB.

   A subsystem member can be specified on the dependent region EXEC parameter. If it is not specified, the control region SSM is used. If the region is never likely to connect to IBM MQ, to avoid loading the adapter, specify a member with no entries.
2. Loads the IBM MQ adapter.

For a batch message program, the load is not done until the application issues its first messaging command. At that time, IMS tries to make the connection.

For a message-processing program region or IMS fast-path region, the attempt is made when the region is initialized.

Region error options

If the queue manager is not active, or if resources are not available when the first messaging command is sent from application programs, the action taken depends on the error option specified on the SSM entry. The options are:

R
The appropriate return code is sent to the application.

Q
The application ends abnormally with abend code U3051. The input message is re-queued.

A
The application ends abnormally with abend code U3047. The input message is discarded.

Monitoring the activity on connections

A thread is established from a dependent region when an application makes its first successful IBM MQ request. You can display information about connections and the applications currently using them by issuing the following command from IBM MQ:

```
+CSQ1 DISPLAY CONN(*) ALL
```

The command produces a message like the following:

```
CONN(BC45A794C4290001)
EXTCONN(C3E2D8C3C3E2D8F14040404040404040)
TYPE(CONN)
CONNOPTS(
MQCNO_STANDARD_BINDING
)
UOWLOGDA(2004-12-15)
UOWLOGTI(16.39.43)
UOWSTDA(2004-12-15)
UOWSTTI(16.39.43)
UOWSTATE(ACTIVE)
NID( )
EXTURID(
0000052900000000
)
QMURID(00000354B76E)
URTYPE(IMS)
USERID(STCPI)
APPLTAG(IM8F)
ASID(0049)
APPLTYPE(IMS)
PSTID(0004)
PSBNAME(GP01MPP)
```

For the control region, *thread-xref* is the special value CONTROL. For dependent regions, it is the PSTid concatenated with the PSBname. *auth-id* is either the user field from the job card, or the ID from the z/OS started procedures table.

For an explanation of the displayed list, see the description of message CSQV402I in the IBM MQ for z/OS mensajes, finalización, y códigos de razón documentation.

IMS provides a display command to monitor the connection to IBM MQ. It shows which program is active on each dependent region connection, the LTERM user name, and the control region connection status. The command is:

```
/DISPLAY SUBSYS name
```

The status of the connection between IMS and IBM MQ is shown as one of:

```
CONNECTED
NOT CONNECTED
CONNECT IN PROGRESS
STOPPED
STOP IN PROGRESS
INVALID SUBSYSTEM NAME= name
SUBSYSTEM name NOT DEFINED BUT RECOVERY OUTSTANDING
```

The thread status from each dependent region is one of the following:

```
CONN
CONN, ACTIVE (includes LTERM of user)
```

Disconnecting from dependent regions

To change values in the SSM member of IMS.PROCLIB, you disconnect a dependent region. To do this, you must:

1. Issue the IMS command:

   ```
   /STOP REGION
   ```

2. Update the SSM member.
3. Issue the IMS command:

   ```
   /START REGION
   ```

z/OS *Disconnecting from IMS*

The connection is ended when either IMS or the queue manager terminates. Alternatively, the IMS master terminal operator can explicitly break the connection.

To terminate the connection between IMS and IBM MQ, use the following IMS command:

```
/STOP SUBSYS sysid
```

The command sends the following message to the terminal that issued it, typically the master terminal operator (MTO):

```
DFS058I STOP COMMAND IN PROGRESS
```

The IMS command:

```
/START SUBSYS sysid
```

is required to reestablish the connection.

Note: The IMS command /STOP SUBSYS is not completed if an IMS trigger monitor is running.

z/OS *Controlling the IMS trigger monitor*

You can use the CSQQTRMN transaction to stop, and start the IMS trigger monitor.

The IMS trigger monitor (the CSQQTRMN transaction) is described in the Setting up the IMS trigger monitor.

To control the IMS trigger monitor see:

- Starting CSQQTRMN
- Stopping CSQQTRMN

Starting CSQQTRMN

1. Start a batch-oriented BMP that runs the program CSQQTRMN for each initiation queue you want to monitor.
2. Modify your batch JCL to add a DDname of CSQQUT1 that points to a data set containing the following information:

```
QMGRNAME=q_manager_name    Comment: queue manager name
INITQUEUENAME=init_q_name   Comment: initiation queue name
LTERM=lterm          Comment: LTERM to remove error messages
CONSOLEMESSAGES=YES      Comment: Send error messages to console
```

   where:

q_manager_name	The name of the queue manager (if this is blank, the default nominated in CSQQDEFV is assumed)
init_q_name	The name of the initiation queue to be monitored
lterm	The IMS LTERM name for the destination of error messages (if this is blank, the default value is MASTER).
CONSOLEMESSAGES=YES	Requests that messages sent to the nominated IMS LTERM are also sent to the z/OS console. If this parameter is omitted or misspelled, the default is NOT to send messages to the console.

3. Add a DD name of CSQQUT2 if you want a printed report of the processing of CSQQUT1 input.

Note:

1. The data set CSQQUT1 is defined with LRECL=80. Other DCB information is taken from the data set. The DCB for data set CSQQUT2 is RECFM=VBA and LRECL=125.

2. You can put only one keyword on each record. The keyword value is delimited by the first blank following the keyword; this means that you can include comments. An asterisk in column 1 means that the whole input record is a comment.
3. If you misspell either of the QMGRNAME or LTERM keywords, CSQQTRMN uses the default for that keyword.
4. Ensure that the subsystem is started in IMS (by the /START SUBSYS command) before submitting the trigger monitor BMP job. If it is not started, your trigger monitor job terminates with abend code U3042.

Stopping CSQQTRMN

Once started, CSQQTRMN runs until either the connection between IBM MQ and IMS is broken due to one of the following events:

- the queue manager ending
- IMS ending

or a z/OS STOP **jobname** command is entered.

z/OS Controlling the IMS bridge

Use this topic to understand the IMS commands that you can use to control the IMS bridge.

There are no IBM MQ commands to control the IBM MQ-IMS bridge. However, you can stop messages being delivered to IMS in the following ways:

- For non-shared queues, by using the ALTER QLOCAL(xxx) GET(DISABLED) command for all bridge queues.
- For clustered queues, by using the SUSPEND QMGR CLUSTER(xxx) command. This is effective only when another queue manager is also hosting the clustered bridge queue.
- For clustered queues, by using the SUSPEND QMGR FACILITY(IMSBRIDGE) command. No further messages are sent to IMS, but the responses for any outstanding transactions are received from IMS.

  To start sending messages to IMS again, issue the RESUME QMGR FACILITY(IMSBRIDGE) command.

You can also use the MQSC command DISPLAY SYSTEM to display whether the bridge is suspended.

See MQSC commands for details of these commands.

For further information see:

- “Starting and stopping the IMS bridge” on page 570
- “Controlling IMS connections” on page 571
- Controlling bridge queues
- “Resynchronizing the IMS bridge” on page 572
- Working with tpipe names
- Deleting messages from IMS
- Deleting tpipes
- “IMS Transaction Expiration” on page 574

Starting and stopping the IMS bridge

Start the IBM MQ bridge by starting OTMA. Either use the IMS command:

```
/START OTMA
```

or start it automatically by specifying OTMA=YES in the IMS system parameters. If OTMA is already started, the bridge starts automatically when queue manager startup has completed. An IBM MQ event message is produced when OTMA is started.

Use the IMS command:

```
/STOP OTMA
```

to stop OTMA communication. When this command is issued, an IBM MQ event message is produced.

Controlling IMS connections

IMS provides these operator commands to control and monitor the connection to IBM MQ:

/DEQUEUE TMEMBER *tmember* TPIPE *tpipe*
Removes messages from a Tpipe. Specify PURGE to remove all messages or PURGE1 to remove the first message only.

/DISPLAY OTMA
Displays summary information about the OTMA server and clients, and client status.

/DISPLAY TMEMBER *name*
Displays information about an OTMA client.

/DISPLAY TRACE TMEMBER *name*
Displays information about what is being traced.

/SECURE OTMA
Sets security options.

/START OTMA
Enables communications through OTMA.

/START TMEMBER *tmember* TPIPE *tpipe*
Starts the named Tpipe.

/STOP OTMA
Stops communications through OTMA.

/STOP TMEMBER *tmember* TPIPE *tpipe*
Stops the named Tpipe.

/TRACE
Controls the IMS trace.

For more information about these commands, see the *IMS/ESA Operators Reference* manual for the level of IMS that you are using.

IMS command responses are sent to the terminal from which the command was issued. Authorization to issue IMS commands is based on IMS security.

Controlling bridge queues

To stop communicating with the queue manager with XCF member name *tmember* through the bridge, issue the following IMS command:

```
/STOP TMEMBER tmember TPIPE ALL
```

To resume communication, issue the following IMS command:

```
/START TMEMBER tmember TPIPE ALL
```

The Tpipes for a queue can be displayed using the MQ DISPLAY QUEUE command.

To stop communication with the queue manager on a single Tpipe, issue the following IMS command:

```
/STOP TMEMBER tmember TPIPE tpipe
```

One or two Tpipes are created for each active bridge queue, so issuing this command stops communication with the IBM MQ queue. To resume communication, use the following IMS command :

```
/START TMEMBER tmember TPIPE tpipe
```

Alternatively, you can alter the attributes of the IBM MQ queue to make it get inhibited.

Resynchronizing the IMS bridge

The IMS bridge is automatically restarted whenever the queue manager, IMS, or OTMA are restarted.

The first task undertaken by the IMS bridge is to resynchronize with IMS. This involves IBM MQ and IMS checking sequence numbers on every synchronized Tpipe. A synchronized Tpipe is used when persistent messages are sent to IMS from an IBM MQ - IMS bridge queue using commit mode zero (commit-then-send).

If the bridge cannot resynchronize with IMS, the IMS sense code is returned in message CSQ2023E and the connection to OTMA is stopped. If the bridge cannot resynchronize with an individual IMS Tpipe, the IMS sense code is returned in message CSQ2025E and the Tpipe is stopped. If a Tpipe has been cold started, the recoverable sequence numbers are automatically reset to 1.

If the bridge discovers mismatched sequence numbers when resynchronizing with a Tpipe, message CSQ2020E is issued. Use the IBM MQ command RESET TPIPE to initiate resynchronization with the IMS Tpipe. You need to provide the XCF group and member name, and the name of the Tpipe; this information is provided by the message.

You can also specify:

- A new recoverable sequence number to be set in the Tpipe for messages sent by IBM MQ, and to be set as the partner's receive sequence number. If you do not specify this, the partner's receive sequence number is set to the current IBM MQ send sequence number.
- A new recoverable sequence number to be set in the Tpipe for messages received by IBM MQ, and to be set as the partner's send sequence number. If you do not specify this, the partner's send sequence number is set to the current IBM MQ receive sequence number.

If there is an unresolved unit of recovery associated with the Tpipe, this is also notified in the message. Use the IBM MQ command RESET TPIPE to specify whether to commit the unit of recovery, or back it out. If you commit the unit of recovery, the batch of messages has already been sent to IMS, and is deleted from the bridge queue. If you back the unit of recovery out, the messages are returned to the bridge queue, to be later sent to IMS.

Commit mode 1 (send-then-commit) Tpipes are not synchronized.

Considerations for Commit mode 1 transactions

In IMS, commit mode 1 (CM1) transactions send their output replies before sync point.

A CM1 transaction might not be able to send its reply, for example because:

- The Tpipe on which the reply is to be sent is stopped
- OTMA is stopped
- The OTMA client (that is, the queue manager) has gone away
- The reply-to queue and dead-letter queue are unavailable

For these reasons, the IMS application sending the message pseudo-abends with code U0119. The IMS transaction and program are not stopped in this case.

These reasons often prevent messages being sent into IMS, as well as replies being delivered from IMS. A U0119 abend can occur if:

- The Tpipe, OTMA, or the queue manager is stopped while the message is in IMS
- IMS replies on a different Tpipe to the incoming message, and that Tpipe is stopped
- IMS replies to a different OTMA client, and that client is unavailable.

Whenever a U0119 abend occurs, both the incoming message to IMS and the reply messages to IBM MQ are lost. If the output of a CM0 transaction cannot be delivered for any of these reasons, it is queued on the Tpipe within IMS.

Working with tpipe names

Many of the commands used to control the IBM MQ - IMS bridge require the *tpipe* name. Use this topic to understand how you can find further details of the tpipe name.

You need *tpipe* names for many of the commands that control the IBM MQ - IMS bridge. You can get the tpipe names from DISPLAY QUEUE command and note the following points:

- tpipe names are assigned when a local queue is defined
- a local queue is given two tpipe names, one for sync and one for non-sync
- tpipe names will not be known to IMS until after some communication between IMS and IBM MQ specific to that particular local queue takes place
- For a tpipe to be available for use by the IBM MQ - IMS bridge its associated queue must be assigned to a Storage Class that has the correct XCF group and member name fields completed

Deleting messages from IMS

A message that is destined for IBM MQ through the IMS bridge can be deleted if the Tmember/Tpipe is stopped. To delete one message for the queue manager with XCF member name *tmember*, issue the following IMS command:

```
/DEQUEUE TMEMBER tmember TPIPE tpipe PURGE1
```

To delete all the messages on the Tpipe, issue the following IMS command:

```
/DEQUEUE TMEMBER tmember TPIPE tpipe PURGE
```

Deleting tpipes

You cannot delete IMS tpipes yourself. They are deleted by IMS at the following times:

- Synchronized tpipes are deleted when IMS is cold started.

- Non-synchronized tpipes are deleted when IMS is restarted.

IMS Transaction Expiration

An expiration time is associated with a transaction; any IBM MQ message can have an expiration time associated with it. The expiration interval is passed from the application, to IBM MQ, using the MQMD.Expiry field. The time is the duration of a message before it expires, expressed as a value in tenths of a second. An attempt to perform the MQGET of a message, later than it has expired, results in the message being removed from the queue and expiry processing performed. The expiration time decreases as a message flows between queue managers on an IBM MQ network. When an IMS message is passed across the IMS bridge to OTMA, the remaining message expiry time is passed to OTMA as a transaction expiration time.

If a transaction has an expiration time specified, OTMA expires the input transactions in three different places in IMS:

- input message receiving from XCF
- input message enqueuing time
- application GU time

No expiration is performed after the GU time.

The transaction EXPRTIME can be provided by:

- IMS transaction definition
- IMS OTMA message header
- IMS DFSINSX0 user exit
- IMS CREATE or UPDATE TRAN commands

IMS indicates that it has expired a transaction by abending a transaction with 0243, and issuing a message. The message issued is either DFS555I in the non-shared-queues environment, or DFS2224I in the shared-queues environment.

z/OS Operating Advanced Message Security on z/OS

The Advanced Message Security address space accepts commands using the z/OS MODIFY command.

Procedure

- Modify Advanced Message Security on z/OS.

  To enter commands for the Advanced Message Security (AMS) address space, use the z/OS MODIFY command.

  For example:

  ```
  F qmgrAMSM, cmd
  ```

  where *qmgr* is the prefix of the started task name.

  The following table describes the MODIFY commands that are accepted:

Table 29. Advanced Message Security address space MODIFY commands

Command	Option	Description
DISPLAY		Display version information

Table 29. Advanced Message Security address space MODIFY commands (continued)		
Command	**Option**	**Description**
REFRESH	KEYRING POLICY ALL	Refresh the key ring certificates, security policies, or both.
SMFAUDIT	SUCCESS FAILURE ALL	Set whether SMF auditing is required when AMS successfully protects or unprotects messages, when AMS fails to protect or unprotect messages, or both.
SMFTYPE	0 - 255	Set the SMF record type to be generated when AMS protects or unprotects messages. To disable SMF auditing specify a record type of 0.

Note: To specify an option it must be separated by a comma. For example:

```
F qmgrAMSM,REFRESH KEYRING
F qmgrAMSM,SMFAUDIT ALL
F qmgrAMSM,SMFTYPE 180
```

- Refresh Advanced Message Security on z/OS.

  Changes that are made effective by issuing the **REFRESH** command apply to applications that issue MQOPEN after the **REFRESH** command has completed. Existing applications that have a queue open, continue to use the options from when the application opened the queue. To use the new values, the application has to close and reopen the queue.
- Start and stop AMS on z/OS.

  You do not need to enter a command to start or stop the Advanced Message Security address space. The AMS address space is started automatically when the queue manager is started if AMS has been enabled with the **SPLCAP** parameter of CSQ6SYSP, and is stopped when the queue manager is stopped.

Administración de IBM MQ Internet Pass-Thru

En esta sección se describe cómo administrar IBM MQ Internet Pass-Thru (MQIPT).

Configure MQIPT realizando cambios en el archivo de configuración `mqipt.conf` tal como se describe en Configuración de IBM MQ Internet Pass-Thru. Para administrar MQIPT, incluida la renovación de MQIPT para que se apliquen los cambios de configuración sin reiniciar MQIPT, utilice el mandato **mqiptAdmin**. Para obtener información sobre cómo administrar MQIPT mediante el mandato **mqiptAdmin**, consulte “administrar MQIPT utilizando la línea de mandatos” en la página 578.

Iniciar y detener MQIPT

Puede iniciar MQIPT desde la línea de mandatos o hacer que se inicie automáticamente cuando se inicie el sistema. Puede detener MQIPT mediante el mandato **mqiptAdmin**.

Inicio de MQIPT desde la línea de mandatos

MQIPT se instala en un directorio de instalación, por ejemplo:

- Windows `C:\MQIPT` en sistemas Windows, con scripts ejecutables en `C:\MQIPT\bin`
- Linux AIX `/opt/mqipt` en sistemas AIX and Linux, con scripts ejecutables en `/opt/mqipt/bin`

MQIPT también utiliza un directorio inicial, que contiene el archivo de configuración `mqipt.conf` y los archivos de salida de MQIPT cuando se está ejecutando. Los subdirectorios siguientes del directorio de inicio de MQIPT se crean automáticamente cuando se invoca MQIPT por primera vez:

- Un directorio `errors` en el que se escriben los archivos First Failure Support Technology (FFST) y de rastreo
- Un directorio `logs` en el que se mantiene el registro de conexión

El ID de usuario con el que se ejecuta MQIPT debe tener permiso para crear estos directorios o bien los directorios ya deben existir y el ID de usuario debe tener permiso para crear, leer y escribir archivos en ellos. Además, si utiliza una política de Java security manager la política de seguridad debe conceder los permisos necesarios para estos directorios. Para obtener más información sobre los valores de política de Security Manager, consulte Java security manager.

Puede utilizar el directorio de instalación como un directorio de inicio. Si utiliza este directorio, debe asegurarse de que el ID de usuario con el que se ejecuta MQIPT tiene los permisos adecuados y que todas las políticas de Security Manager se hayan configurado correctamente.

Para iniciar MQIPT, utilice el mandato **mqipt**, que se encuentra en el directorio `bin` del directorio de instalación de MQIPT. Por ejemplo, el mandato siguiente inicia una instancia de MQIPT que utiliza el directorio `C:\mqiptHome` como directorio inicial:

```
mqipt C:\mqiptHome
```

Para obtener más información sobre el mandato **mqipt**, consulte mqipt (start MQIPT).

Puede utilizar el mandato **mqipt** para especificar un nombre que se proporcionará a la instancia de MQIPT que se está iniciando. El nombre de la instancia de MQIPT se utiliza para administrar las instancias locales de MQIPT con el mandato **mqiptAdmin** sin necesidad de utilizar un puerto de mandatos. Si no se especifica este parámetro, el nombre del directorio de inicio de MQIPT se utiliza como nombre de la instancia de MQIPT.

Los mensajes de la consola muestran el estado de MQIPT. Si se produce un error, consulte Resolución de problemas de IBM MQ Internet Pass-Thru. Los mensajes siguientes son un ejemplo de la salida cuando MQIPT se inicia correctamente:

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is C:\mqiptHome
MQCPI021 Password checking has been enabled on the command port
MQCPI144 MQ Advanced capabilities not enabled
MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
MQCPI006 Route 1414 is starting and will forward messages to :
MQCPI034 ....examplehost(1414)
MQCPI035 ....using MQ protocols
MQCPI057 ....trace level 5 enabled
MQCPI078 Route 1414 ready for connection requests
```

Inicio automático de MQIPT

Puede instalar MQIPT como un servicio del sistema que se inicia automáticamente cuando se inicia el sistema. Utilice el mandato **mqiptService** para instalar y desinstalar el servicio MQIPT.

- Windows En sistemas Windows, el mandato **mqiptService** instala MQIPT como un servicio Windows.
- Linux AIX En sistemas AIX and Linux , el mandato **mqiptService** instala MQIPT como un servicio de inicialización de System V que se inicia cuando se arranca el sistema. En los sistemas Linux que no dan soporte a System V init, utilice otro método, como `systemd`, para gestionar MQIPT como servicio.

Cuando se inicia el servicio MQIPT, se inician todas las rutas de MQIPT activas. Cuando se detiene el servicio, todas las rutas están sujetas a la conclusión inmediata.

Puede instalar solo un servicio MQIPT en un sistema, aun cuando haya más de una instalación de MQIPT en el sistema.

Para obtener más información sobre el mandato **mqiptService**, consulte mqiptService (gestionar el servicio MQIPT).

Detener MQIPT

Puede detener MQIPT mediante el mandato **mqiptAdmin** con el parámetro **-stop**.

Por ejemplo, el mandato siguiente detiene una instancia de MQIPT con el nombre `mqipt1` que se ejecuta localmente con el mismo ID de usuario que el mandato **mqiptAdmin**:

```
mqiptAdmin -stop -n ipt1
```

El mandato **mqiptAdmin** se conecta a la instancia activa de MQIPT para administrar utilizando uno de los métodos siguientes:

- conectando a una instancia local de MQIPT sin utilizar el puerto de mandatos.
- realizando una conexión de red a un puerto de mandatos.

Se debe habilitar primero el cierre remoto estableciendo la propiedad **RemoteShutDown** en `true` para utilizar el mandato **mqiptAdmin** para detener MQIPT enviando un mandato a un puerto de mandatos.

Para obtener más información sobre cómo administrar MQIPT mediante el mandato **mqiptAdmin**, consulte “administrar MQIPT utilizando la línea de mandatos” en la página 578.

Especificación de la clave de cifrado de contraseña

Si la configuración de MQIPT contiene contraseñas que están cifradas utilizando una clave de cifrado distinta de la clave predeterminada, debe proporcionar la clave de cifrado de contraseña en un archivo que MQIPT pueda leer cuando se inicie.

Archivo de claves de cifrado de contraseña

Las contraseñas cifradas para almacenarlas y utilizarlas en MQIPT se pueden cifrar mediante una clave de cifrado que proporcione. Si no proporciona ninguna clave de cifrado, se utilizará la clave de cifrado predeterminada. No tiene que especificar una clave de cifrado de contraseña, pero es más seguro hacerlo. Si no especifica ninguna clave de cifrado, se utilizará la clave de cifrado predeterminada.

Si proporciona una clave de cifrado de contraseña, se debe almacenar en un archivo al que se pueda acceder mediante el mandato **mqiptPW** que se utiliza para cifrar las contraseñas e MQIPT. Las únicas restricciones en el contenido del archivo son que debe contener al menos un carácter y solo una línea de texto.

Nota: debe asegurarse de que se han establecido los permisos de archivo adecuados en el archivo de claves de cifrado de contraseña para impedir que los usuarios no autorizados lean la clave de cifrado. Solo el usuario que ejecuta el mandato **mqiptPW** y el usuario con el que se ejecuta MQIPT necesitan autorización para leer la clave de cifrado de contraseña.

La misma clave de cifrado de contraseña se utiliza para cifrar y descifrar todas las contraseñas almacenadas para una instancia de MQIPT. Por lo tanto, solo necesita un único archivo de claves de cifrado de contraseña para cada instalación de MQIPT.

Si cambia la clave de cifrado de contraseña de una instalación de MQIPT, se deben volver a cifrar todas las contraseñas cifradas utilizando la nueva clave de cifrado.

Iniciar MQIPT

El nombre predeterminado del archivo de claves de cifrado de contraseña es *MQIPT_HOME_DIR*/`mqipt_cred.key`, donde *MQIPT_HOME_DIR* es el directorio donde se almacena el archivo de configuración de `mqipt.conf`. Si tiene previsto ejecutar MQIPT como un servicio que se inicia

automáticamente, debe crear el archivo de claves de cifrado de contraseña con el nombre predeterminado.

Si el archivo de claves de cifrado de contraseña se crea con un nombre que no sea el nombre predeterminado, se debe proporcionar el nombre del archivo a MQIPT al iniciarse. El nombre del archivo de claves de cifrado de contraseña se puede especificar utilizando cualquiera de los métodos siguientes, en orden de preferencia:

1. El parámetro **-sf** en el mandato **mqipt** utilizado para iniciar MQIPT.
2. la variable de entorno `MQS_MQIPTCRED_KEYFILE`.
3. la propiedad `com.ibm.mq.ipt.cred.keyfile` Java.

Si no se proporciona ningún nombre de archivo de claves de cifrado de contraseña, se utilizará el nombre de archivo predeterminado, si el archivo existe. Si no existe el archivo de claves de cifrado de contraseña predeterminado, se utilizará la clave de cifrado de contraseña predeterminada.

administrar MQIPT utilizando la línea de mandatos

Puede utilizar el mandato **mqiptAdmin** en la línea de mandatos para administrar MQIPT.

Puede utilizar el mandato **mqiptAdmin** para realizar las siguientes funciones administrativas:

- Listar instancias locales activas de MQIPT.
- Renovar una instancia de MQIPT después de realizar cambios en el archivo de configuración.
- Detener una instancia de MQIPT.

El mandato `mqiptAdmin` se encuentra en el subdirectorio `bin` del directorio de instalación de MQIPT.

El mandato **mqiptAdmin** se conecta a la instancia activa de MQIPT para administrar utilizando uno de los métodos siguientes:

- realizando una conexión de red a un puerto de mandatos.
- conectando a una instancia local de MQIPT sin utilizar el puerto de mandatos.

El mandato **mqiptAdmin** es compatible con las versiones anteriores de MQIPT, pero no se puede utilizar para administrar versiones de MQIPT con una versión posterior a la versión del mandato **mqiptAdmin**. En un entorno que incluye distintas versiones de MQIPT, debe utilizar la última versión del mandato **mqiptAdmin**.

Para obtener más información sobre la sintaxis del mandato **mqiptAdmin**, consulte mqiptAdmin (administrar MQIPT).

Administración local sin un puerto de mandatos

Las instancias locales de MQIPT se pueden administrar sin utilizar un puerto de mandatos. La administración local le permite administrar MQIPT utilizando el mandato **mqiptAdmin** sólo cuando este se ejecuta en el mismo sistema que la instancia de MQIPT que desea administrar.

Para que **mqiptAdmin** esté autorizado a administrar una instancia local de MQIPT sin utilizar el puerto de mandatos, la instancia de MQIPT debe estar en ejecución en el mismo sistema y con el mismo ID de usuario que **mqiptAdmin**. Como alternativa, en AIX and Linux, **mqiptAdmin** se puede ejecutar como `root`.

De forma predeterminada, la administración local está habilitada. Para inhabilitar la administración local, utilice la propiedad de configuración **LocalAdmin**. Para obtener más información sobre la propiedad **LocalAdmin**, consulte LocalAdmin.

Para administrar las instancias locales de MQIPT, debe proporcionar un nombre a cada una. Puede asignar un nombre a una instancia de MQIPT utilizando el parámetro **-n** al iniciar MQIPT con el mandato **mqipt**. Si no especifica un nombre al iniciar MQIPT, se utiliza el nombre del directorio de inicio como el

nombre de la instancia de MQIPT. Por ejemplo, el siguiente mandato inicia MQIPT y asigna el nombre `ipt1` a la instancia:

```
mqipt /opt/mqipt1 -n ipt1
```

Una vez que la instancia tiene un nombre, puede administrar esta instancia especificando el nombre en el mandato **mqiptAdmin** con el parámetro **-n**. Por ejemplo, el mandato siguiente detiene la instancia local de MQIPT con el nombre `ipt1`:

```
mqiptAdmin -stop -n ipt1
```

Puede listar todas las instancias activas locales de MQIPT que el mandato **mqiptAdmin** está autorizado a administrar sin utilizar un puerto de mandatos utilizando el mandato **mqiptAdmin** con el parámetro **-list**. Por ejemplo, el siguiente mandato lista todas las instancias activas locales de MQIPT que el usuario que ha iniciado el mandato **mqiptAdmin** está autorizado a administrar:

```
mqiptAdmin -list
```

Administración utilizando un puerto de mandatos

Puede configurar MQIPT con un puerto de mandatos que no esté protegido y un puerto de mandatos que esté protegido con TLS. Puede utilizar estos puertos de mandatos para administrar MQIPT como cualquier usuario que está en el mismo sistema que la instancia de MQIPT que desea administrar, o desde un sistema remoto.

Las versiones anteriores de MQIPT sólo aceptan mandatos administrativos emitidos a un puerto de mandatos no seguro.

Nota: Las conexiones al puerto de mandatos no seguro no se cifran, por lo que otros usuarios de la red pueden ver los datos enviados mediante la red al puerto de mandatos no seguro, incluida la contraseña de acceso de MQIPT.

Para que MQIPT esté a la escucha en un puerto de mandatos de los mandatos emitidos por el mandato **mqiptAdmin**, se debe especificar un valor para las propiedades **CommandPort** o **SSLCommandPort** en la sección global del archivo de configuración `mqipt.conf`.

Revise las consideraciones de seguridad de Otras consideraciones de seguridad antes de habilitar los puertos de mandatos de MQIPT. Considere habilitar la autenticación para los mandatos recibidos por los puertos de mandatos. Para obtener más información sobre la autenticación de puerto de mandatos, consulte “Autenticación de puerto de mandatos” en la página 583.

Para administrar una instancia de MQIPT utilizando un puerto de mandatos, especifique la dirección de red del host donde se ejecuta MQIPT y el número de puerto de mandatos como parámetros del mandato **mqiptAdmin**. Por ejemplo, para renovar la instancia de MQIPT en ejecución en `mqipt.example.com` y que tiene el puerto de mandatos no seguro configurado para estar a la escucha en el puerto 1890, emita el mandato siguiente:

```
mqiptAdmin -refresh -r mqipt.example.com:1890
```

Si no especifica el nombre de host y el número de puerto, **mqiptAdmin** intenta conectarse a `localhost`, puerto 1881.

Para obtener más información sobre cómo administrar MQIPT mediante el puerto de mandatos TLS, consulte “Administración de MQIPT utilizando el puerto de mandatos TLS” en la página 579.

Administración de MQIPT utilizando el puerto de mandatos TLS

MQIPT se puede configurar para utilizar un puerto de mandatos TLS para escuchar los mandatos administrativos emitidos por el mandato **mqiptAdmin** . Mediante el puerto de mandatos TLS se protegen datos confidenciales como por ejemplo la contraseña de acceso de MQIPT en la red entre **mqiptAdmin** y MQIPT. Utilice este procedimiento para configurar el puerto de mandatos TLS y administrar MQIPT utilizando el puerto de mandatos TLS.

Acerca de esta tarea

El puerto de mandatos TLS debe estar configurado con un certificado de servidor almacenado en un almacén de claves PKCS #12 o en hardware criptográfico que dé soporte a la interfaz de señales criptográficas PKCS #11 . El certificado de servidor de puerto de mandatos se envía al mandato **mqiptAdmin** durante el reconocimiento TLS. Esta tarea presupone que solicita un nuevo certificado de servidor de una entidad emisora de certificados (CA) de confianza y que el certificado se le devuelve en un archivo. El mandato **mqiptAdmin** valida el certificado de puerto de mandatos utilizando el certificado de CA de la CA que ha firmado el certificado de servidor. El certificado de CA debe almacenarse en un almacén de claves PKCS #12 al que se pueda acceder mediante el mandato **mqiptAdmin** .

El puerto de mandatos TLS no admite la autenticación de certificados de cliente. Para habilitar la autenticación para los mandatos administrativos emitidos al puerto de mandatos, consulte "Autenticación de puerto de mandatos" en la página 583.

Este procedimiento describe cómo gestionar los almacenes de claves y certificados digitales necesarios para utilizar el puerto de mandatos TLS utilizando el mandato V 9.4.0 V 9.4.0 **mqiptKeytool** . Para obtener más información sobre la gestión de almacenes de claves que MQIPT utiliza, consulte Gestión de almacenes de claves de MQIPT.

Procedimiento

1. Siga estos pasos para configurar el puerto de mandatos TLS para la instancia de MQIPT.
   a) Cree un par de claves pública y privada y un certificado de servidor de puerto de mandatos TLS asociado en un almacén de claves PKCS #12 .

   V 9.4.0 V 9.4.0 Para crear el almacén de claves que contiene el certificado de servidor de puerto de mandatos TLS, especifique el mandato siguiente:

   ```
   mqiptKeytool -genkeypair -keystore filename -storetype pkcs12 -storepass password
                -dname distinguished_name -alias label
                -keyalg key_algorithm -keysize key_size -sigalg sig_algorithm
   ```

   donde:

   -keystore *nombre_archivo*
   Especifica el nombre del almacén de claves.

   -storepass *contraseña*
   Especifica la contraseña del almacén de claves.

   -alias *etiqueta*
   Especifica la etiqueta de certificado.

   -keyalg *algoritmo_clave*
   Especifica el algoritmo que se utiliza para crear el par de claves.

   -keysize *tamaño_clave*
   Especifica el tamaño de clave.

   -sigalg *algoritmo*
   Especifica el algoritmo que se utiliza para firmar el certificado.

   -dname *nombre_distinguido*
   Especifica el nombre distinguido X.500 especificado entre comillas dobles.

   b) Cree una solicitud de certificado para el certificado de servidor de puerto de mandatos TLS firmado por una CA.

   V 9.4.0 V 9.4.0 Para crear una solicitud de certificado, especifique el mandato siguiente:

   ```
   mqiptKeytool -certreq -keystore filename -storetype pkcs12 -storepass password
                -alias label -file certreq_filename
   ```

   donde:

-keystore *nombre_archivo*
Especifica el nombre del almacén de claves.

-storepass *contraseña*
Especifica la contraseña del almacén de claves.

-alias *etiqueta*
Especifica la etiqueta de certificado.

-file *nombrearchivo_solcert*
Especifica el nombre de archivo para la solicitud de certificado.

c) Envía el archivo de solicitud de certificado creado en el paso "1.b" en la página 580 a la CA para que la firme.

d) Después de que la CA le envíe el certificado firmado, reciba el certificado firmado en el almacén de claves.

V 9.4.0 V 9.4.0 Para recibir el certificado firmado en el almacén de claves, especifique el mandato siguiente:

```
mqiptKeytool -importcert -keystore cert_filename -storetype pkcs12 -storepass password
             -file cert_filename
```

donde *nombre_archivo_cert* es el nombre del archivo que contiene el certificado, *nombre_archivo* es el nombre del almacén de claves y *contraseña* es la contraseña del almacén de claves.

e) Cifre la contraseña del almacén de claves utilizando el mandato **mqiptPW** .

Escriba el mandato siguiente:

```
mqiptPW -sf encryption_key_file
```

donde *archivo_claves_cifrado* es el nombre de un archivo que contiene la clave de cifrado de contraseña para la instalación de MQIPT. No es necesario que especifique el parámetro **-sf** si la instalación de MQIPT utiliza la clave de cifrado de contraseña predeterminada. Escriba la contraseña del almacén de claves para cifrar cuando se le solicite.

Para obtener más información sobre el mandato **mqiptPW**, consulte Cifrado de una contraseña de conjunto de claves.

f) Edite el archivo de configuración `mqipt.conf` y especifique las siguientes propiedades para configurar el puerto de mandatos TLS:

i) Establezca el valor de la propiedad **SSLCommandPort** en el número de puerto del mandato TLS.

ii) Establezca el valor de la propiedad **SSLCommandPortKeyRing** en el nombre de archivo del almacén de claves creado en el paso "1.a" en la página 580.

iii) Establezca el valor de **SSLCommandPortKeyRingPW** en la salida de script del mandato **mqiptPW** en el paso "1.e" en la página 581.

iv) Establezca el valor de la propiedad **SSLCommandPortSiteLabel** en el nombre de etiqueta del certificado de puerto de mandato TLS, especificado al crear la solicitud de certificado en el paso "1.b" en la página 580.

v) Si desea restringir las conexiones de entrada al puerto de mandatos TLS a las de una interfaz de red determinada, establezca el valor de la propiedad **SSLCommandPortListenerAddress** en una dirección de red que pertenezca a una de las interfaces de red del sistema en el que se ejecuta MQIPT. Por ejemplo, para restringir las conexiones de entrada al puerto de mandatos TLS a las de la máquina local, establezca el valor de la propiedad **SSLCommandPortListenerAddress** en `localhost`.

g) Inicie o renueve MQIPT para habilitar el puerto de mandatos TLS.

MQIPT emite mensajes de la consola como los siguientes para mostrar la configuración de puerto de mandatos TLS en vigor:

```
MQCPI155 Listening for control commands on port 1882 on local address * using TLS
MQCPI139 ......secure socket protocols <NULL>
MQCPI031 ......cipher suites <NULL>
```

```
MQCPI032 ......key ring file c:\\iptHome\\ssl\\commandport.p12
MQCPI072 ......and certificate label mqiptadmin
```

2. En el sistema donde se utiliza el mandato **mqiptAdmin** para administrar MQIPT, siga estos pasos para habilitar **mqiptAdmin** para conectarse al puerto de mandatos TLS.
   a) Importe el certificado de CA de la CA que ha firmado el certificado de puerto de mandatos TLS en un almacén de claves PKCS #12 que utilizará como almacén de confianza el mandato **mqiptAdmin** .

   V 9.4.0 V 9.4.0 Para importar el certificado de CA, especifique el mandato siguiente:

   ```
   mqiptKeytool -importcert -keystore filename -storetype pkcs12 -storepass password
                -file cert_filename -alias certlabel
   ```

   donde:

   nombre_archivo
   Especifica el nombre del almacén de claves que se va a crear

   Contraseña
   Especifica la contraseña del almacén de claves

   etiqueta_cert
   Especifica la etiqueta que se proporcionará al certificado de CA

   nombrearchivo_cert
   Especifica el nombre del archivo que contiene el certificado de CA

   b) Cifre la contraseña del almacén de claves utilizando el mandato **mqiptPW** .

   Escriba el mandato siguiente:

   ```
   mqiptPW -sf encryption_key_file
   ```

   donde *archivo_claves_cifrado* es el nombre del archivo que contiene la clave de cifrado de contraseña. El archivo de claves de cifrado de contraseña puede ser distinto al utilizado para cifrar contraseñas en la configuración MQIPT. La clave de cifrado de contraseña predeterminada se utiliza si no especifica un archivo de claves de cifrado con el parámetro **-sf**. Escriba la contraseña del almacén de claves para cifrar cuando se le solicite.

   Para obtener más información sobre el mandato **mqiptPW**, consulte Cifrado de una contraseña de conjunto de claves.

   c) Cree un archivo de propiedades que utilizará el mandato **mqiptAdmin** y especifique las propiedades siguientes:

   ```
   SSLClientCAKeyRing=key_ring_file_name
   SSLClientCAKeyRingPW=key_ring_password
   PasswordProtectionKeyFile=encryption_key_file
   ```

   donde:

   nombre_archivo_conjunto_claves
   es el nombre del almacén de claves creado en el paso “2.a” en la página 582.

   contraseña_conjunto_claves
   es la salida de contraseña cifrada del mandato **mqiptPW** del paso “2.b” en la página 582.

   archivo_claves_cifrado
   es el nombre del archivo que contiene la clave de cifrado de contraseña. Debe especificar la propiedad **PasswordProtectionKeyFile** sólo si se ha utilizado un archivo de claves de cifrado para cifrar la contraseña de almacén de claves en el paso “2.b” en la página 582.

   d) Emita el mandato **mqiptAdmin** para administrar MQIPT, especificando el parámetro **-s** para indicar que se requiere una conexión TLS y el parámetro **-p** para especificar el nombre del archivo de propiedades creado en el paso “2.c” en la página 582.

Por ejemplo, especifique el mandato siguiente para renovar una instancia de MQIPT enviando un mandato de renovación al puerto de mandatos TLS:

```
mqiptAdmin -refresh -r hostname:port -s -p properties_file
```

El mandato **mqiptAdmin** emite un mensaje como el siguiente para confirmar que la conexión a MQIPT está protegida con TLS:

```
MQCAI109 The connection to MQIPT is secured with TLSv1.2.
```

Qué hacer a continuación

Para habilitar la autenticación para los mandatos recibidos por el puerto de mandatos TLS, siga los pasos de "Autenticación de puerto de mandatos" en la página 583.

Autenticación de puerto de mandatos

MQIPT se puede configurar para autenticar los mandatos recibidos por el puerto de mandatos no seguro y el puerto de mandatos TLS utilizando una contraseña. Utilice este procedimiento para habilitar la autenticación de puerto de mandatos.

Acerca de esta tarea

El mandato **mqiptAdmin** solicita a los usuarios que especifiquen una contraseña cuando el mandato se conecta al puerto de mandatos de una instancia de MQIPT que tiene la autenticación de puerto de mandatos habilitada. MQIPT valida la contraseña especificada en el mandato **mqiptAdmin** con la contraseña de acceso especificada en la configuración de MQIPT.

Las propiedades que establece para la autenticación de puerto de mandatos se aplican tanto al puerto de mandatos TLS como al puerto de mandatos no seguro.

Procedimiento

1. Cifre la contraseña de acceso de MQIPT mediante el mandato **mqiptPW**.

   Escriba el mandato siguiente:

   ```
   mqiptPW -sf encryption_key_file
   ```

   donde *archivo_claves_cifrado* es el nombre del archivo que contiene la clave de cifrado de contraseña para la instalación de MQIPT. No es necesario que especifique el parámetro **-sf** si la instalación de MQIPT utiliza la clave de cifrado de contraseña predeterminada. Cuando se le solicite, especifique la contraseña de acceso para el cifrado.

   Para obtener más información sobre el cifrado de contraseñas en la configuración de MQIPT, consulte Cifrado de contraseñas almacenadas.
2. Edite el archivo de configuración `mqipt.conf` y especifique las siguientes propiedades:

   ```
   AccessPW=encrypted_password
   RemoteCommandAuthentication=auth_setting
   ```

   donde:

   contraseña_cifrada
   es la salida de contraseña cifrada del mandato **mqiptPW** del paso "1" en la página 583.

   valor_autenticación
   es el requisito de autenticación. La autenticación de puerto de mandatos está habilitada si esta propiedad está establecida en uno de los valores siguientes:

   opcional
   No se requiere ninguna contraseña, pero si se proporciona una contraseña esta debe ser válida. Esta opción puede resultar de utilidad durante la migración, por ejemplo.

necesario
Se debe proporcionar una contraseña válida con cada mandato recibido por un puerto de mandatos.

Para obtener más información sobre estas propiedades, consulte Propiedades globales de MQIPT.

3. Inicie o renueve MQIPT para que se apliquen los cambios.

   MQIPT emite un mensaje que indica si la autenticación de puerto de mandatos está habilitada. Por ejemplo, si MQIPT está configurado para requerir que se especifique una contraseña válida cada vez que se ejecuta el mandato **mqiptAdmin**, se emite el mandato siguiente:

```
MQCPI021 Password checking has been enabled on the command port
```

Realización de copias de seguridad

Debe hacer copia de seguridad de una serie de archivos de MQIPT como parte de los procedimientos de copia de seguridad habituales.

Realice copia de seguridad de los archivos siguientes de forma regular:

- El archivo de configuración, `mqipt.conf`
- Los archivos de conjunto de claves SSL/TLS especificados por las siguientes propiedades en `mqipt.conf`:
  - **SSLClientKeyRing**
  - **SSLClientCAKeyRing**
  - **SSLServerKeyRing**
  - **SSLServerCAKeyRing**
  - **SSLCommandPortKeyRing**
- Los archivos de contraseña del conjunto de claves SSL/TLS especificados por las siguientes propiedades en `mqipt.conf`:
  - **SSLClientKeyRingPW**
  - **SSLClientCAKeyRingPW**
  - **SSLServerKeyRingPW**
  - **SSLServerCAKeyRingPW**
- El archivo de claves de cifrado de contraseña, si la configuración de MQIPT contiene contraseñas que están cifradas con una clave de cifrado distinta de la clave predeterminada.
- El archivo de política especificado en **SecurityManagerPolicy**, si se ha establecido esa propiedad.
- Los archivos de salida de seguridad y los archivos de salida de certificado especificados por las siguientes propiedades en `mqipt.conf`:
  - **SecurityExitName**
  - **SSLExitName**
- Archivos de registro de conexión en el subdirectorio `log` del directorio inicial de MQIPT, si son necesarios para fines de auditoría.

Ajuste de rendimiento

Puede ajustar el rendimiento relativo de cada ruta de MQIPT utilizando una combinación de una agrupación de hebras y una especificación de tiempo de espera inactivo.

Hebras de conexión

A cada ruta de MQIPT se le asigna una agrupación de trabajo de hebras que se ejecutan simultáneamente y gestionan las solicitudes de comunicación entrantes. Durante la inicialización, se crea una agrupación

de hebras (del tamaño especificado en el atributo `MinConnectionThreads` de la ruta) y se asigna una hebra para gestionar la primera solicitud entrante. Cuando llega esta solicitud, se asigna otra hebra, lista para la siguiente solicitud entrante. Cuando se han asignado todas las hebras para el trabajo, se crea una nueva hebra, se añade a la agrupación de trabajo y se asigna para el trabajo.

De este modo, la agrupación aumenta hasta que se alcanza el número máximo de hebras (especificado en **MaxConnectionThreads**). Las hebras se liberan de nuevo a la agrupación cuando finaliza una conversación o transcurre el período de tiempo de espera inactivo especificado. Cuando se alcanza el número máximo de hebras de trabajo, la solicitud entrante siguiente espera hasta que se libere una hebra de nuevo a la agrupación de trabajo.

Puede reducir el tiempo que las solicitudes podrían tener que esperar aumentando el número de hebras disponibles. Sin embargo, debe equilibrar este aumento con los recursos del sistema que están disponibles.

Tiempo de espera inactivo

De forma predeterminada, las hebras de trabajo no terminan debido a la inactividad. Cuando se ha asignado una hebra a una conversación, permanece asignada a esa conversación hasta que se cierra normalmente, se desactiva la ruta o se cierra MQIPT. Opcionalmente, puede especificar un intervalo de intervalo de tiempo de espera inactivo (en minutos) en la propiedad **IdleTimeout** para que las hebras que han estado inactivas durante el periodo de tiempo especificado se reciclen. Las hebras se reciclan para su uso, colocándolas de nuevo en la agrupación de trabajo.

Si la actividad de IBM MQ es intermitente, establezca su intervalo de pulsaciones en un valor menor que el del tiempo de espera de MQIPT para que las hebras no se reciclen constantemente.

Avisos

Esta información se ha desarrollado para productos y servicios ofrecidos en los Estados Unidos.

Es posible que IBM no ofrezca los productos, servicios o las características que se tratan en este documento en otros países. Consulte al representante local de IBM para obtener información sobre los productos y servicios que actualmente pueden adquirirse en su zona. Las referencias a programas, productos o servicios de IBM no pretenden establecer ni implicar que sólo puedan utilizarse dichos productos, programas o servicios de IBM. En su lugar podrá utilizarse cualquier producto, programa o servicio equivalente que no infrinja ninguno de los derechos de propiedad intelectual de IBM. No obstante, es responsabilidad del usuario evaluar y verificar el funcionamiento de cualquier producto, programa o servicio no IBM.

IBM puede tener patentes o solicitudes de patentes pendientes que cubran el tema principal descrito en este documento. El suministro de este documento no le otorga ninguna licencia sobre estas patentes. Puede enviar consultas sobre licencias, por escrito, a:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Para consultas sobre licencias relacionadas con información de doble byte (DBCS), póngase en contacto con el Departamento de propiedad intelectual de IBM de su país o envíe las consultas por escrito a:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokio 103-8510, Japón

El párrafo siguiente no se aplica al Reino Unido ni a ningún otro país donde estas disposiciones contradigan la legislación vigente: INTERNATIONAL BUSINESS MACHINES CORPORATION PROPORCIONA ESTA PUBLICACIÓN "TAL CUAL" SIN NINGÚN TIPO DE GARANTÍA, YA SEA EXPLÍCITA O IMPLÍCITA, INCLUYENDO, PERO SIN LIMITARSE A, LAS GARANTÍAS IMPLÍCITAS DE NO INCUMPLIMIENTO, COMERCIABILIDAD O IDONEIDAD PARA UNA FINALIDAD DETERMINADA. Algunas legislaciones no contemplan la exclusión de garantías, ni implícitas ni explícitas, en determinadas transacciones, por lo que puede haber usuarios a los que no les afecte dicha norma.

Esta información puede contener imprecisiones técnicas o errores tipográficos. La información aquí contenida está sometida a cambios periódicos; tales cambios se irán incorporando en nuevas ediciones de la publicación. IBM puede realizar mejoras y/o cambios en los productos y/o programas descritos en esta publicación en cualquier momento sin previo aviso.

Las referencias hechas en esta publicación a sitios web que no son de IBM se proporcionan sólo para la comodidad del usuario y no constituyen de modo alguno un aval de esos sitios web. Los materiales de estos sitios web no forman parte de los materiales para este producto IBM, por lo que la utilización de dichos sitios web es a cuenta y riesgo del usuario.

IBM puede utilizar o distribuir cualquier información que el usuario le proporcione del modo que considere apropiado sin incurrir por ello en ninguna obligación con respeto al usuario.

Los titulares de licencias de este programa que deseen información del mismo con el fin de permitir: (i) el intercambio de información entre los programas creados de forma independiente y otros programas (incluido este) y (ii) el uso mutuo de la información intercambiada, deben ponerse en contacto con:

IBM Corporation
Software Interoperability Coordinator, Department 49XA

3605 Highway 52 N
Rochester, MN 55901
U.S.A.

Dicha información puede estar disponible, sujeta a los términos y condiciones apropiados, incluyendo, en algunos casos, el pago de una cantidad.

El programa bajo licencia que se describe en esta información y todo el material bajo licencia disponible para el mismo lo proporciona IBM bajo los términos del Acuerdo de cliente de IBM, el Acuerdo de licencia de programas internacional de IBM o cualquier acuerdo equivalente entre las partes.

Los datos de rendimiento incluidos en este documento se han obtenido en un entorno controlado. Por consiguiente, los resultados obtenidos en otros entornos operativos pueden variar de manera significativa. Es posible que algunas mediciones se hayan realizado en sistemas en nivel de desarrollo y no existe ninguna garantía de que estas mediciones serán las mismas en sistemas disponibles generalmente. Además, es posible que algunas mediciones se hayan estimado mediante extrapolación. Los resultados reales pueden variar. Los usuarios de este documento deben verificar los datos aplicables a su entorno específico.

La información relativa a productos que no son de IBM se obtuvo de los proveedores de esos productos, sus anuncios publicados u otras fuentes de disponibilidad pública. IBM no ha comprobado estos productos y no puede confirmar la precisión de su rendimiento, compatibilidad o alguna reclamación relacionada con productos que no sean de IBM. Todas las preguntas sobre las prestaciones de productos que no son de IBM deben dirigirse a los proveedores de dichos productos.

Todas las declaraciones relacionadas con una futura intención o tendencia de IBM están sujetas a cambios o se pueden retirar sin previo aviso y sólo representan metas y objetivos.

Este documento contiene ejemplos de datos e informes que se utilizan diariamente en la actividad de la empresa. Para ilustrar los ejemplos de la forma más completa posible, éstos incluyen nombres de personas, empresas, marcas y productos. Todos estos nombres son ficticios y cualquier similitud con los nombres y direcciones utilizados por una empresa real es puramente casual.

LICENCIA DE DERECHOS DE AUTOR:

Esta información contiene programas de aplicación de ejemplo en lenguaje fuente que ilustran técnicas de programación en diversas plataformas operativas. Puede copiar, modificar y distribuir estos programas de ejemplo de cualquier forma sin pagar ninguna cuota a IBM para fines de desarrollo, uso, marketing o distribución de programas de aplicación que se ajusten a la interfaz de programación de aplicaciones para la plataforma operativa para la que se han escrito los programas de ejemplo. Los ejemplos no se han probado minuciosamente bajo todas las condiciones. IBM, por tanto, no puede garantizar la fiabilidad, servicio o funciones de estos programas.

Puede que si visualiza esta información en copia software, las fotografías e ilustraciones a color no aparezcan.

Información acerca de las interfaces de programación

La información de interfaz de programación, si se proporciona, está pensada para ayudarle a crear software de aplicación para su uso con este programa.

Este manual contiene información sobre las interfaces de programación previstas que permiten al cliente escribir programas para obtener los servicios de IBM MQ.

Sin embargo, esta información puede contener también información de diagnóstico, modificación y ajustes. La información de diagnóstico, modificación y ajustes se proporciona para ayudarle a depurar el software de aplicación.

Importante: No utilice esta información de diagnóstico, modificación y ajuste como interfaz de programación porque está sujeta a cambios.

Marcas registradas

IBM, el logotipo de IBM , ibm.com, son marcas registradas de IBM Corporation, registradas en muchas jurisdicciones de todo el mundo. Hay disponible una lista actual de marcas registradas de IBM en la web en "Copyright and trademark information"www.ibm.com/legal/copytrade.shtml. Otros nombres de productos y servicios pueden ser marcas registradas de IBM o de otras empresas.

Microsoft y Windows son marcas registradas de Microsoft Corporation en Estados Unidos y/o otros países.

UNIX es una marca registrada de Open Group en Estados Unidos y en otros países.

Linux es una marca registrada de Linus Torvalds en Estados Unidos y en otros países.

Este producto incluye software desarrollado por Eclipse Project (https://www.eclipse.org/).

Java y todas las marcas registradas y logotipos son marcas registradas de Oracle o sus afiliados.

Número Pieza:

(1P) P/N: