9.4

Amministrazione di IBM MQ

Nota

Prima di utilizzare queste informazioni e il prodotto che supportano, leggere le informazioni in “Informazioni particolari” a pagina 579.

Questa edizione si applica alla versione 9 release 4 di IBM® MQ e a tutte le successive release e modifiche se non diversamente indicato nelle nuove edizioni.

Indice

AmministrazioneIBM MQ

Per gestire i gestori code IBM MQ e le risorse associate, scegliere il metodo preferito da una serie di attività che è possibile utilizzare per attivare e gestire tali risorse.

Informazioni su questa attività

È possibile gestire gli oggetti IBM MQ in locale o in remoto:

gestione locale
Per amministrazione locale si intende l'esecuzione di attività di amministrazione su qualsiasi gestore code definito sul sistema locale. È possibile accedere ad altri sistemi, ad esempio tramite **telnet**del programma di emulazione terminale TCP/IP, ed eseguire la gestione. In IBM MQ, è possibile considerarlo come amministrazione locale poiché non sono coinvolti canali, ovvero la comunicazione è gestita dal sistema operativo.

Per ulteriori informazioni, consultare “Utilizzo degli oggetti IBM MQ locali” a pagina 129.

Amministrazione remota
IBM MQ supporta l'amministrazione da un singolo punto di contatto tramite amministrazione remota. La gestione remota consente di immettere comandi dal sistema locale che vengono elaborati su un altro sistema e si applica anche a IBM MQ Explorer. Ad esempio, è possibile immettere un comando remoto per modificare una definizione di coda su un gestore code remoto. Non è necessario accedere a tale sistema, anche se è necessario definire i canali appropriati. Il gestore code e il server dei comandi sul sistema di destinazione devono essere in esecuzione.

Alcuni comandi non possono essere emessi in questo modo, in particolare, creando o avviando i gestori code e avviando i server dei comandi. Per eseguire questo tipo di attività, è necessario collegarsi al sistema remoto e immettere i comandi da tale ubicazione oppure creare un processo che possa emettere i comandi per conto dell'utente. Questa limitazione si applica anche a IBM MQ Explorer.

Per ulteriori informazioni, consultare “Utilizzo di oggetti IBM MQ remoti” a pagina 198.

Esistono diversi metodi che è possibile utilizzare per creare e gestire i gestori code e le risorse correlate in IBM MQ. Questi metodi includono interfacce della riga comandi, interfacce utente grafiche e un'API di gestione.

Esistono diverse serie di comandi che è possibile utilizzare per amministrare IBM MQ a seconda della piattaforma:

- “Comandi di controllo IBM MQ” a pagina 8
- “Comandi di script IBM MQ (MQSC)” a pagina 8
- “PCF (Programmable Command Format)” a pagina 8
- administrative REST API
- IBM i “CL (Control Language) IBM i” a pagina 9

Ci sono anche le seguenti opzioni per la creazione e la gestione degli oggetti IBM MQ :

- Linux Windows “Il IBM MQ Explorer” a pagina 9
- “Il IBM MQ Console” a pagina 9
- Windows “MSCS (Microsoft Cluster Service)” a pagina 10

z/OS Per informazioni sulle interfacce di gestione e le opzioni su IBM MQ for z/OS, consultare “Administering IBM MQ for z/OS” a pagina 452.

È possibile automatizzare alcune attività di amministrazione e monitoraggio per i gestori code locali e remoti utilizzando i comandi PCF. Questi comandi possono essere semplificati anche utilizzando MQAI

(IBM MQ Administration Interface) su alcune piattaforme. Per ulteriori informazioni sull'automazione delle attività di amministrazione, consultare “Automazione della gestione IBM MQ utilizzando i comandi PCF” a pagina 25.

Concetti correlati

IBM MQ - Sommario tecnico

Attività correlate

Pianificazione

Configurazione

Riferimenti correlati

Confronto serie di comandi

Modi di gestione dei gestori code IBM MQ e delle risorse associate

È possibile gestire i gestori code IBM MQ e le risorse associate utilizzando i comandi di controllo IBM MQ , MQSC (IBM MQ Script Commands), PCF (Programmable Command Format), administrative REST API, IBM MQ Consolee IBM MQ Explorer. Per IBM i è possibile utilizzare anche IBM i Control Language e per Windows è possibile utilizzare anche Microsoft Cluster Service (MSCS).

Comandi di controllo IBM MQ

Multi

I comandi di controllo forniscono un modo per eseguire una serie di attività di gestione IBM MQ . Per AIX, Linux®, and Windows, immettere questi comandi sulla riga comandi del sistema. Per IBM i, immettere questi comandi all'interno di una Qshell. Consultare “Amministrazione di IBM MQ for Multiplatforms utilizzando i comandi di controllo” a pagina 10.

Comandi di script IBM MQ (MQSC)

Utilizzare i comandi MQSC per gestire gli oggetti del gestore code, inclusi il gestore code, le code, le definizioni di processo, gli elenchi nomi, i canali, i canali di connessione client, i listener, i servizi e gli oggetti delle informazioni di autenticazione.

ALW Su AIX, Linux, and Windows, si apre un prompt dei comandi **runmqsc** , quindi si immettono i comandi MQSC a un gestore code locale o remoto da tale prompt. È possibile eseguire questa operazione in modo interattivo oppure eseguire una sequenza di comandi da un file di testo ASCII. Per ulteriori informazioni, consultare “Esecuzione interattiva dei comandi MQSC in runmqsc” a pagina 17 e “Esecuzione di comandi MQSC da file di testo in runmqsc” a pagina 22.

IBM i Su IBM i, si crea un elenco di comandi in un file di script, quindi si esegue il file utilizzando il comando **STRMQMMQSC** . Per ulteriori informazioni, consultare “Amministrazione utilizzando i comandi MQSC su IBM i” a pagina 392.

z/OS Su z/OS, i comandi MQSC possono essere emessi da un numero di origini, a seconda del comando. Per ulteriori informazioni, consultare “Sources from which you can issue MQSC and PCF commands on IBM MQ for z/OS” a pagina 453.

PCF (Programmable Command Format)

I PCF (Programmable Command Format) definiscono i messaggi di comando e risposta che possono essere scambiati tra un programma e qualsiasi gestore code (che supporta PCF) in una rete. È possibile utilizzare comandi PCF in un programma applicativo di gestione dei sistemi per la gestione di oggetti IBM MQ : oggetti delle informazioni di autenticazione, canali, listener di canali, elenchi nomi, definizioni di processo, gestori code, code, servizi e classi di memoria. L'applicazione può operare da un singolo punto nella rete per comunicare le informazioni sul comando e sulla risposta con qualsiasi gestore code, locale o remoto, utilizzando il gestore code locale.

Per ulteriori informazioni sui PCF, consultare “Introduzione a IBM MQ Programmable Command Formats” a pagina 26.

Per la definizione di PCF e strutture per comandi e risposte, consultare Programmable command formats reference.

Il administrative REST API

administrative REST API fornisce un'interfaccia RESTful che puoi utilizzare per amministrare IBM MQ. Quando si utilizza administrative REST API, si richiama un metodo HTTP su un URL che rappresenta un oggetto IBM MQ . Ad esempio, è possibile richiedere informazioni sulle installazioni di IBM MQ utilizzando il metodo HTTP GET sul seguente URL:

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

È possibile utilizzare administrative REST API con l'implementazione HTTP/REST di un linguaggio di programmazione o utilizzando strumenti come cURLo un componente aggiuntivo del browser del client REST.

Per ulteriori informazioni, consultare administrative REST API

Il IBM MQ Console

È possibile utilizzare IBM MQ Console per amministrare IBM MQ da un browser Web.

Per ulteriori informazioni, consultare “Amministrazione mediante IBM MQ Console” a pagina 91.

Il IBM MQ Explorer

Linux Windows

Utilizzando IBM MQ Explorer, è possibile eseguire le seguenti azioni:

- Definire e controllare varie risorse, come gestori code, code, definizioni di processi, elenchi nomi, canali, canali di connessione client, listener, servizi e cluster.
- Avviare o arrestare un gestore code locale e i relativi processi associati.
- Visualizzare i gestori code e i relativi oggetti associati sulla stazione di lavoro o da altre stazioni di lavoro.
- Verificare lo stato dei gestori code, dei cluster e dei canali.
- Verificare quali applicazioni, utenti o canali hanno una particolare coda aperta, dallo stato della coda.

Sui sistemi Windows e Linux for x86-64 , è possibile avviare IBM MQ Explorer utilizzando il menu di sistema o il file eseguibile `MQExplorer` .

Linux Su Linux, per avviare correttamente IBM MQ Explorer , è necessario essere in grado di scrivere un file nella propria directory home e la directory home deve esistere.

Per ulteriori informazioni, consultare “Amministrazione mediante IBM MQ Explorer” a pagina 120.

È possibile utilizzare IBM MQ Explorer per gestire i gestori code remoti su altre piattaforme incluso z/OS.

Da IBM MQ 9.3.0, IBM MQ Explorer è stato rimosso dal pacchetto di installazione di IBM MQ . Rimane disponibile come download separato e può essere installato dal download IBM MQ Explorer autonomo disponibile da Fix Central. Per ulteriori informazioni, consultare Installazione e disinstallazione di IBM MQ Explorer come applicazione autonoma su Linux e Windows.

CL (Control Language) IBM i

IBM i

Questo è il modo preferito per emettere comandi di amministrazione per IBM MQ for IBM i. I comandi possono essere emessi dalla riga comandi o scrivendo un programma CL. Questi comandi eseguono funzioni simili ai comandi PCF, ma il formato è diverso. I comandi CL sono progettati esclusivamente per

i server e le risposte CL sono leggibili, mentre i comandi PCF sono indipendenti dalla piattaforma e sia i formati di comandi che di risposta sono destinati all'utilizzo del programma.

Per informazioni dettagliate su IBM i CL (Control Language), consultare “Gestione di IBM MQ for IBM i utilizzando i comandi CL” a pagina 378 e IBM MQ for IBM i comandi CL.

MSCS (Microsoft Cluster Service)

Windows

Microsoft Cluster Service (MSCS) consente di collegare i server in un *cluster*, offrendo una maggiore disponibilità di dati e applicazioni e semplificando la gestione del sistema. MSCS è in grado di rilevare e ripristinare automaticamente gli errori del server o dell'applicazione.

È importante non confondere i cluster in senso MSCS con i cluster IBM MQ . La distinzione è la seguente:

IBM MQCluster
: Si tratta di gruppi di due o più gestori code su uno o più computer, che forniscono l'interconnessione automatica e consentono la condivisione delle code per il bilanciamento del carico e la ridondanza.

Cluster MSCS
: Si tratta di gruppi di computer, collegati tra loro e configurati in modo tale che, in caso di errore, MSCS esegua un *failover*, trasferendo i dati di stato delle applicazioni dal computer in errore a un altro computer nel cluster e inizializzando nuovamente l'operazione.

Supporto di Microsoft Cluster Service (MSCS) fornisce informazioni dettagliate su come configurare il tuo sistema IBM MQ for Windows per utilizzare MSCS.

Attività correlate

“Amministrazione di IBM MQ utilizzando i comandi MQSC” a pagina 12
È possibile utilizzare i comandi di MQSC per gestire gli oggetti del gestore code, inclusi il gestore code, le code, le definizioni di processo, canali, canali di connessione client, listener, servizi, elenchi nomi, cluster e oggetti delle informazioni di autenticazione. I comandi MQSC sono disponibili su tutte le piattaforme.

Riferimenti correlati

Riferimento di amministrazione

Multi Amministrazione di IBM MQ for Multiplatforms utilizzando i comandi di controllo

I comandi di controllo forniscono un modo per eseguire una serie di attività di gestione IBM MQ . Per AIX, Linuxe Windows, si immettono questi comandi sulla riga comandi del sistema. Per IBM i, immettere questi comandi all'interno di una Qshell.

Prima di iniziare

Quando si utilizzano i comandi di controllo che operano su un gestore code, è necessario utilizzare il comando dall'installazione associata al gestore code che si sta utilizzando.

Quando si utilizzano comandi di controllo che operano su un gestore code configurato per utilizzare l'autenticazione di connessione con CHCKLOCL (REQUIRED) e si osserva un errore di connessione,

- Fornire un ID utente e una parola d'ordine se il comando di controllo lo consente.
- Utilizzare gli equivalenti MQSC dei comandi di controllo laddove esistono.
- Avviare il gestore code utilizzando l'opzione -ns, mentre è necessario eseguire i comandi di controllo che non possono connettersi.

Nota: Piattaforme differenti possono accettare gli argomenti del comando immessi in un ordine diverso. In particolare, ciò significa che i comandi che funzionano su Linux potrebbero non funzionare su altre piattaforme. Per questo motivo, è necessario immettere sempre gli argomenti come specificato nei diagrammi di sintassi.

Per un elenco completo dei comandi di controllo, consultare il manuale IBM MQ control commands reference.

Procedura

- Linux AIX

  Utilizzare i comandi di controllo sui sistemi AIX and Linux .

  Nei sistemi IBM MQ for AIX or Linux , immettere i comandi di controllo in una finestra della shell.

  Se si desidera immettere comandi di controllo, l'ID utente deve essere un membro del gruppo mqm per la maggior parte dei comandi di controllo. Per ulteriori informazioni su questo argomento, consultare Authority to amministrare IBM MQ su AIX, Linux, and Windows. Inoltre, prendere nota delle informazioni specifiche dell'ambiente. per la piattaforma o le piattaforme utilizzate dall'azienda.

  Negli ambienti UNIX and Linux , i comandi di controllo, incluso il nome del comando stesso, gli indicatori ed eventuali argomenti, sono sensibili al maiuscolo / minuscolo. Ad esempio, nel comando:

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

  – Il nome del comando deve essere `crtmqm`, non `CRTMQM`.
  – L'indicatore deve essere `-u`, non `-U`.
  – La coda di messaggi non recapitabili è denominata `SYSTEM.DEAD.LETTER.QUEUE`.
  – L'argomento è specificato come `jupiter.queue.manager`, che è diverso da `JUPITER.queue.manager`.

  Fare attenzione a digitare i comandi esattamente come vengono visualizzati negli esempi.

- Windows

  Utilizzare i comandi di controllo sui sistemi Windows .

  In IBM MQ for Windows, immettere i comandi di controllo da un prompt dei comandi.

  Se si desidera immettere comandi di controllo, l'ID utente deve essere un membro del gruppo mqm per la maggior parte dei comandi di controllo. Per ulteriori informazioni su questo argomento, consultare Authority to amministrare IBM MQ su AIX, Linux, and Windows. Inoltre, prendere nota delle informazioni specifiche dell'ambiente. per la piattaforma o le piattaforme utilizzate dall'azienda.

  I comandi di controllo e i relativi indicatori non sono sensibili al maiuscolo / minuscolo, ma gli argomenti di questi comandi, come i nomi delle code e dei gestori code, sono sensibili al maiuscolo / minuscolo.

  Ad esempio, nel comando:

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

  – Il nome del comando può essere immesso in lettere maiuscole o minuscole oppure in una combinazione delle due. Questi sono tutti validi: `crtmqm`, `CRTMQM`e `CRTmqm`.
  – L'indicatore può essere immesso come `-u`, `-U`, `/u`o `/U`.
  – `SYSTEM.DEAD.LETTER.QUEUE` e `jupiter.queue.manager` devono essere immessi esattamente come mostrato.

- IBM i

  Utilizzare i comandi di controllo sui sistemi IBM i .

  Su IBM MQ for IBM i, si eseguono i comandi di controllo da un ambiente Qshell. Per utilizzare Qshell, immettere STRQSH sulla riga comandi IBM i . È possibile uscire e tornare alla riga comandi in qualsiasi momento premendo F3.

Un numero ridotto di comandi di controllo non è supportato su IBM i. Ad esempio, i comandi di installazione multipla non sono supportati poiché non è possibile avere più di una copia di IBM MQ su un sistema IBM i . I comandi non supportati su IBM i sono contrassegnati da ALW nel manuale IBM MQ control commands reference.

Riferimenti correlati

Riferimento ai comandi di controllo IBM MQ

Amministrazione di IBM MQ utilizzando i comandi MQSC

È possibile utilizzare i comandi di MQSC per gestire gli oggetti del gestore code, inclusi il gestore code, le code, le definizioni di processo, canali, canali di connessione client, listener, servizi, elenchi nomi, cluster e oggetti delle informazioni di autenticazione. I comandi MQSC sono disponibili su tutte le piattaforme.

Informazioni su questa attività

I comandi MQSC disponibili sono descritti in dettaglio in Riferimento ai comandi MQSC.

Il modo in cui si eseguono i comandi MQSC dipende dalla piattaforma:

- ALW Su AIX, Linux, and Windows, si immettono comandi MQSC a un gestore code dal prompt dei comandi di **runmqsc** . È possibile utilizzare questa richiesta comandi in diversi modi:
  - Interattivamente, emettendo comandi MQSC da una tastiera. Consultare “Esecuzione interattiva dei comandi MQSC in runmqsc” a pagina 17.
  - Immissione di comandi MQSC da un file di testo ASCII. Consultare “Esecuzione di comandi MQSC da file di testo in runmqsc” a pagina 22.
  - Immissione di comandi MQSC su un gestore code remoto. Consultare “Immissione di comandi MQSC su un gestore code remoto” a pagina 203.
- IBM i Su IBM i, si crea un elenco di comandi in un file di script, quindi si esegue il file utilizzando il comando **STRMQMMQSC** . Per ulteriori informazioni, consultare “Amministrazione utilizzando i comandi MQSC su IBM i” a pagina 392.
- z/OS Su z/OS, i comandi MQSC possono essere emessi da un numero di origini, a seconda del comando. Per ulteriori informazioni, consultare “Sources from which you can issue MQSC and PCF commands on IBM MQ for z/OS” a pagina 453.

Procedura

Attività correlate

Risoluzione dei problemi con i comandi MQSC

Riferimenti correlati

runmqsc (esecuzione comandi MQSC)

Sintassi del comando MQSC

È possibile utilizzare i comandi MQSC per gestire gli oggetti del gestore code. I comandi MQSC sono disponibili su tutte le piattaforme. Alcuni elementi della sintassi dei comandi sono specifici della piattaforma.

Sequenza di parametri

Ogni comando inizia con un parametro primario (un verbo) e questo è seguito da un parametro secondario (un nome). Questo è quindi seguito dal nome o dal nome generico dell'oggetto (tra parentesi) se ne esiste uno, che è presente nella maggior parte dei comandi. Successivamente, i parametri possono di solito verificarsi in qualsiasi ordine; se un parametro ha un valore corrispondente, il valore deve essere immediatamente successivo al parametro a cui si riferisce.

Nota: z/OS Su z/OS, il parametro secondario non deve essere il secondo.

Spazi e virgole

Le parole chiave, le parentesi e i valori possono essere separati da qualsiasi numero di spazi e virgole. Una virgola mostrata nei diagrammi di sintassi può essere sempre sostituita da uno o più spazi vuoti. Ogni parametro deve essere preceduto da almeno uno spazio vuoto (dopo il parametro primario) tranne che su z/OS .

Qualsiasi numero di spazi vuoti può verificarsi all'inizio o alla fine del comando e tra parametri, punteggiatura e valori. Ad esempio, è valido il seguente comando:

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

Gli spazi vuoti all'interno di una coppia di apici sono significativi.

Le virgole aggiuntive possono essere visualizzate in qualsiasi punto in cui sono consentiti spazi e vengono trattate come se fossero spazi vuoti (a meno che, naturalmente, non siano all'interno di stringhe racchiuse tra virgolette).

Parametri ripetuti

Non sono consentiti parametri ripetuti. La ripetizione di un parametro con la sua versione "NO", come in REPLACE NOREPLACE, non è consentita.

Stringhe e virgolette singole

Le stringhe che contengono spazi, caratteri minuscoli o caratteri speciali devono essere racchiuse tra virgolette singole, a meno che non si verifichi una delle seguenti condizioni:

- I caratteri speciali sono uno o più dei seguenti:
  - Punto (.)
  - Barra (/)
  - Trattino basso (_)
  - Segno percentuale (%)
- z/OS Il comando viene emesso dalle operazioni IBM MQ for z/OS e dai pannelli di controllo.
- La stringa è un valore generico che termina con un asterisco. (su IBM i devono essere racchiuse tra virgolette singole)
- La stringa è un singolo asterisco, ad esempio TRACE (*) (su IBM i devono essere racchiusi tra virgolette singole)
- La stringa è una specifica di intervallo contenente due punti, ad esempio CLASS (01:03)

Se la stringa contiene una virgoletta singola, la virgoletta singola è rappresentata da due virgolette singole.

Multi Su Multipiattaforme, una stringa che non contiene caratteri (ossia, due virgolette singole senza spazi tra di loro) viene interpretata come uno spazio vuoto racchiuso tra virgolette singole, vale a dire, interpretato nello stesso modo di (''). L'eccezione è se l'attributo utilizzato è uno dei seguenti attributi, quando due virgolette singole senza spazio vengono interpretate come una stringa di lunghezza zero:

- TOPICSTR
- SUB
- USERDATA
- SELECTOR

z/OS Su z/OS, se si desidera uno spazio vuoto racchiuso tra virgolette singole, è necessario immetterlo come tale (''). Una stringa che non contiene caratteri ('') è uguale all'immissione di ().

Gli spazi vuoti finali negli attributi stringa basati sui tipi MQCHARV, come ad esempio SELECTOR, dati utente secondari, vengono trattati come significativi, il che significa che `'abc  '` non è uguale a `'abc'`.

Parentesi vuote

Una parentesi di apertura seguita da una parentesi di chiusura, senza alcuna informazione significativa nel mezzo, non è valida se non dove specificatamente indicato. Ad esempio, la seguente stringa non è valida:

```
NAME ( )
```

Minuscolo e maiuscolo

Le parole chiave non sono sensibili al maiuscolo / minuscolo: AltER, alter e ALTER sono tutte accettabili.

Tutto ciò che non è contenuto tra virgolette viene ripiegato in maiuscolo.

Sinonimi

I sinonimi sono definiti per alcuni parametri. Ad esempio, DEF è sempre un sinonimo di DEFINE, quindi DEF QLOCAL è valido. I sinonimi non sono, tuttavia, solo stringhe minime; DEFI non è un sinonimo valido per DEFINE.

Nota: Non esiste alcun sinonimo per il parametro DELETE. Ciò per evitare l'eliminazione accidentale di oggetti quando si utilizza DEF, il sinonimo di DEFINE.

Caratteri speciali

I comandi MQSC utilizzano determinati caratteri speciali per avere determinati significati. Per ulteriori informazioni su questi caratteri speciali e su come utilizzarli, consultare “MQSC: Caratteri speciali e valori generici” a pagina 14.

Attività correlate

Risoluzione dei problemi con i comandi MQSC

Riferimenti correlati

runmqsc (esecuzione comandi MQSC)

MQSC: Caratteri speciali e valori generici

Alcuni caratteri, ad esempio barra retroversa (\) e doppi apici (") I caratteri hanno significati speciali quando vengono utilizzati con i comandi MQSC. Alcuni caratteri speciali che possono essere utilizzati con i parametri possono avere valori generici ma devono essere specificati correttamente.

Precedere la barra retroversa (\) e le virgolette (") con un \, ovvero, immettere \\ o \" se si desidera \ o " nel testo.

Ogni volta che un parametro può avere un valore generico, viene immesso con un asterisco (*), ad esempio ABC*. Un valore generico indica tutti i valori che iniziano con; quindi ABC* indica tutti i valori che iniziano con ABC. Se nel valore vengono utilizzati caratteri che richiedono le virgolette, l'asterisco deve essere inserito all'interno delle virgolette, quindi `'abc*'`. L'asterisco deve essere l'ultimo o l'unico carattere nel valore.

Il punto interrogativo (?) e i due punti (:) non sono consentiti nei valori generici.

Quando è necessario utilizzare uno di questi caratteri speciali in un campo (ad esempio, come parte di una descrizione), è necessario racchiudere l'intera stringa tra virgolette singole.

Tabella 1. Descrizioni di caratteri che hanno significati speciali

Carattere	Descrizione
	Gli spazi vengono utilizzati come separatori. Più spazi sono equivalenti ad un singolo spazio, tranne che nelle stringhe racchiuse tra apici ('). Tutti gli spazi vuoti finali in tali attributi stringa che si basano sui tipi MQCHARV vengono considerati significativi.
,	Le virgole vengono utilizzate come separatori. Più virgole sono equivalenti a una singola virgola, tranne che in stringhe racchiuse tra apici (').
'	Un apostrofo indica l'inizio o la fine di una stringa. IBM MQ lascia tutti i caratteri racchiusi tra virgolette esattamente come sono stati immessi. Gli apostrofi che contengono non vengono inclusi nel calcolo della lunghezza della stringa.
''	Le virgolette singole all'interno di una stringa vengono considerate da IBM MQ come un carattere quando si calcola la lunghezza della stringa e la stringa non viene terminata.
=	z/OS Su z/OS, un segno di uguale indica l'inizio di un valore di parametro che termina con una virgola o uno spazio.
(	Una parentesi aperta indica l'inizio di un valore di parametro o di un elenco di valori.
)	Una parentesi di chiusura indica la fine di un valore di parametro o di un elenco di valori.
:	I due punti indicano un intervallo inclusivo. Ad esempio (1: 5) indica (1,2,3,4, 5). Questa notazione può essere utilizzata solo nei comandi **TRACE** .
*	Un asterisco indica tutto. Ad esempio, **DISPLAY TRACE (*)** indica di visualizzare tutte le tracce e **DISPLAY QUEUE (PAY*)** indica di visualizzare tutte le code con nomi che iniziano con PAY.

Sintassi del file di input MQSC

Se si dispone di comandi lunghi o si utilizza una particolare sequenza di comandi ripetutamente, è possibile utilizzare un file di input per emettere comandi MQSC. Il contenuto del file di input deve seguire la sintassi descritta in questo argomento.

Panoramica

I comandi di MQSC vengono immessi tramite il dispositivo di input standard , indicato anche come `stdin`. Di solito questa è la tastiera, ma è possibile specificare che l'input deve provenire da un file di input.

È possibile utilizzare questo file di input con uno dei seguenti strumenti specifici della piattaforma:

- ALW Il comando **runmqsc** su AIX, Linux, and Windows. Vedere “Esecuzione di comandi MQSC da file di testo in runmqsc” a pagina 22
- IBM i Il comando **STRMQM** su IBM i. Vedere “Amministrazione utilizzando i comandi MQSC su IBM i” a pagina 392
- z/OS I dataset di inizializzazione CSQINP1, CSQINP2e CSQINPX o il programma di utilità batch CSQUTIL su z/OS. Vedere “Sources from which you can issue MQSC and PCF commands on IBM MQ for z/OS” a pagina 453

Sintassi

Sintassi del file di input MQSC:

- Per la portabilità tra ambienti IBM MQ , limitare la lunghezza della riga nei file di comandi MQSC a 72 caratteri.
- Ciascun comando deve iniziare su una nuova riga.
- Una riga che inizia con un asterisco (*) nella prima posizione viene ignorata. Questo può essere utilizzato per inserire commenti nel file.
- Le righe vuote vengono ignorate.
- Un segno più (+) indica che il comando continua dal primo carattere non vuoto nella riga successiva. Se si utilizza + per continuare un comando, ricordarsi di lasciare almeno uno spazio vuoto prima del parametro successivo (tranne su z/OS dove non è necessario). I commenti o le righe vuote vengono eliminati quando il comando viene riassemblato in una singola stringa.
- Un segno meno (-), indica che il comando deve essere continuato dall'inizio della riga successiva. I commenti o le righe vuote vengono eliminati quando il comando viene riassemblato in una singola stringa.
- I comandi MQSC contenuti in un comando Escape PCF (Programmable Command Format) non possono continuare con il segno più o con il segno meno. L'intero comando deve essere contenuto in un singolo comando Escape. Per informazioni sui comandi PCF, consultare “Introduzione a IBM MQ Programmable Command Formats” a pagina 26.
- Su Multipiattaformee su z/OS per i comandi emessi dal programma di utilità batch CSQUTIL, è possibile utilizzare un carattere punto e virgola (;) per terminare un comando, anche se è stato immesso un segno più (+) alla fine della riga precedente.
- Una riga non deve terminare con un carattere di controllo della tastiera (ad esempio, una tabulazione).
- Se si esegue il comando **runmqsc** in modalità client reindirizzando `stdin` da un file di testo e si fornisce l'indicatore **-u** per fornire le credenziali, il comando **runmqsc** non richiede una password e la password viene letta da `stdin`. Verificare che la prima riga di dati fornita tramite `stdin` sia la password. Questa operazione può essere eseguita utilizzando strumenti della riga comandi come "echo" o "cat" e passando la password seguita dallo script MQSC nel **runmqsc** comando `stdin`.
- Windows Su Windows, se alcuni caratteri speciali come il segno cancelletto (£) e il NOT logico (¬) vengono utilizzati in uno script di comandi (ad esempio, come parte di una descrizione dell'oggetto), vengono visualizzati in modo diverso nell'output di un comando come **DISPLAY QLOCAL**.

Consultare anche “Sintassi del comando MQSC” a pagina 12.

Esempi

Il seguente esempio è un estratto da un file di comandi di MQSC che riporta il comando **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. Estrai da un file di comandi MQSC

Quando il comando **runmqsc** viene completato, viene restituito un report. Il seguente esempio è un estratto da un report:

```
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. Estrai da un file di report di comandi MQSC

È anche possibile utilizzare i file di comando MQSC di esempio per creare il proprio file di testo:

amqscos0.tst
Definizioni di oggetti utilizzati da programmi di esempio.

amqscic0.tst
Definizioni di code per transazioni CICS .

Linux AIX Su AIX and Linux, questi file si trovano nella directory *MQ_INSTALLATION_PATH*/samp. *MQ_INSTALLATION_PATH* rappresenta la directory di alto livello in cui è installato IBM MQ .

Windows Su Windows, questi file si trovano nella directory *MQ_INSTALLATION_PATH*\tools\mqsc\samples. *MQ_INSTALLATION_PATH* rappresenta la directory di alto livello in cui è installato IBM MQ .

ALW Esecuzione interattiva dei comandi MQSC in runmqsc

Su AIX, Linux, and Windows, è possibile utilizzare il prompt dei comandi **runmqsc** per immettere i comandi MQSC per un gestore code in modo interattivo. La corsa interattiva è particolarmente indicata per test rapidi.

Prima di iniziare

È necessario utilizzare il comando **runmqsc** dall'installazione associata al gestore code che si sta utilizzando. È possibile individuare a quale installazione è associato un gestore code utilizzando il comando `dspmq -o installation` .

È possibile rendere più semplice la presenza di un ambiente MQSC e visualizzare alcuni dettagli dell'ambiente corrente impostando un prompt a scelta utilizzando la variabile di ambiente **MQPROMPT** . Per ulteriori informazioni, consultare “Impostazione del prompt dei comandi MQSC” a pagina 20.

Linux AIX Quando si eseguono comandi MQSC in modo interattivo su piattaforme AIX and Linux , il prompt dei comandi **runmqsc** supporta anche funzioni aggiuntive dell'editor della riga comandi. Consultare “Abilitazione del richiamo e del completamento dei comandi e dei tasti comando Emacs per runmqsc” a pagina 21.

Informazioni su questa attività

Il comando **runmqsc** viene usato per aprire un prompt dei comandi da cui è possibile emettere comandi MQSC. Questi comandi e la relativa sintassi sono descritti nel Riferimento ai comandi MQSC.

Quando si avvia il prompt dei comandi **runmqsc** come descritto in questa attività, si imposta il prompt in modo che venga eseguito in una delle tre modalità, a seconda degli indicatori impostati sul comando:

- *Modalità di verifica*, in cui i comandi MQSC vengono verificati su un gestore code locale, ma non vengono eseguiti.
- *Modalità diretta*, in cui i comandi MQSC vengono eseguiti su un gestore code locale.
- *Modalità indiretta*, dove i comandi MQSC vengono eseguiti su un gestore code remoto.

La procedura riportata di seguito imposta la richiesta di esecuzione in modalità diretta. Altre opzioni sono illustrate negli esempi che seguono i passi principali.

Procedura

1. Aprire una finestra di comandi o una shell e immettere il seguente comando:

   ```
   runmqsc QMgrName
   ```

   Dove *QMgrName* specifica il nome del gestore code che si desidera elaborare i comandi MQSC.
   È possibile lasciare vuoto il campo *QMgrName* per elaborare comandi MQSC sul gestore code predefinito.
2. Immettere i comandi MQSC, come richiesto. Ad esempio, per creare una coda locale denominata ORANGE.LOCAL.QUEUE immettere il seguente comando:

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

   Per i comandi che hanno troppi parametri per adattarsi su una riga, utilizzare i caratteri di continuazione per indicare che un comando continua sulla seguente riga:

   - Un segno meno (-) indica che il comando deve continuare dall'inizio della seguente riga.
   - Un segno più (+) indica che il comando deve continuare dal primo carattere non vuoto sulla riga seguente.

   L'immissione del comando termina con il carattere finale di una riga non vuota che non è un carattere di continuazione. È anche possibile terminare esplicitamente l'input del comando immettendo un punto e virgola (;).
3. Interrompere l'utilizzo dei comandi MQSC immettendo il comando seguente:

   ```
   end
   ```

   In alternativa, è possibile utilizzare il comando **exit** , il comando **quit** o il carattere EOF per il proprio sistema operativo.

Risultati

Quando si immettono i comandi MQSC, il gestore code restituisce i messaggi dell'operatore che confermano le azioni dell'utente o che indicano gli errori effettuati. Ad esempio, il seguente messaggio conferma la creazione di una coda:

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

Il seguente messaggio indica che è stato commesso un errore di sintassi:

```
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
```

Questi messaggi vengono inviati all'unità di emissione standard, che per impostazione predefinita è la visualizzazione. Se il comando non è stato immesso correttamente, fare riferimento alle informazioni di riferimento del comando per trovare la sintassi corretta. Consultare Riferimento ai comandi MQSC.

Esempio

Le seguenti sono varianti del comando `runmqsc` *QMgrName* utilizzato nei passi precedenti. Queste varianti creano diverse configurazioni del prompt dei comandi **runmqsc** .

- Il comando riportato di seguito utilizza il filtro dei comandi per passare un singolo comando MQSC all'interprete MQSC.

  Su Windows:

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

  Su Linux:

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

- Il seguente comando non specifica un nome gestore code, quindi i comandi MQSC vengono elaborati sul gestore code predefinito.

  ```
  runmqsc
  ```

- Questo comando inoltra i comandi al gestore code QMREMOTE utilizzando QMLOCAL per inoltrare i comandi.

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

- Questo comando verifica che la sintassi del comando sia corretta su un gestore code locale senza eseguire i comandi. Tenere presente che i comandi da verificare vengono letti da un file di input `myprog.in`.

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

  Per ulteriori informazioni sull'utilizzo dei file di input e output, consultare “Esecuzione di comandi MQSC da file di testo in runmqsc” a pagina 22.

Operazioni successive

Per dettagli completi sulla sintassi del comando **runmqsc** , sui parametri facoltativi e sui codici di ritorno, consultare runmqsc (esecuzione di comandi MQSC).

Attività correlate

“Esecuzione di comandi MQSC da file di testo in runmqsc” a pagina 22

Se si dispone di comandi lunghi o si sta utilizzando una particolare sequenza di comandi ripetutamente, è possibile utilizzare un file di testo per immettere comandi MQSC. È possibile reindirizzare `stdin` da un file di testo. È anche possibile reindirizzare l'emissione a un file.

Riferimenti correlati
Riferimento comandi MQSC

ALW Impostazione del prompt dei comandi MQSC

Su AIX, Linux, and Windows, utilizzare la variabile di ambiente **MQPROMPT** per impostare la richiesta visualizzata quando si esegue il comando **runmqsc** . Ciò rende più semplice vedere di essere in un ambiente MQSC e visualizzare alcuni dettagli dell'ambiente corrente.

Informazioni su questa attività

È possibile impostare la richiesta visualizzata quando si esegue il comando **runmqsc** . La richiesta viene inserita sia quando il comando **runmqsc** viene eseguito in modo interattivo, sia quando l'input viene reindirizzato in **runmqsc** da un file o dalla periferica di input standard (stdin).

È possibile includere testo semplice nel prompt dei comandi ed è anche possibile inserire variabili di ambiente utilizzando la notazione +VARNAME+ nello stesso modo delle definizioni di oggetto servizio IBM MQ . Per ulteriori informazioni, consultare “Utilizzo di inserimenti sostituibili nelle definizioni servizio” a pagina 188.

Esistono diversi altri inserimenti sostituibili aggiuntivi forniti da IBM MQ , descritti nella seguente tabella.

Inserimento sostituibile	Descrizione
MQ_HOST_NAME	Nome host del sistema
MQ_FILE_SEP	Separatore file specifico della piattaforma: • Linux AIX Su sistemi AIX and Linux , **MQ_FILE_SEP** è /. • Windows Su sistemi Windows , l'ubicazione del **MQ_FILE_SEP** è \
MQ_PATH_SEP	Separatore di percorso specifico della piattaforma: • Linux AIX Su sistemi AIX and Linux , **MQ_PATH_SEP** è :. • Windows Su sistemi Windows , l'ubicazione del **MQ_PATH_SEP** è ;
MQ_DATE_TIME	Data e ora del sistema locale in un formato YYYY-MM-DD hh:mm:ss.SSS fisso, ad esempio: `2020-12-25 17:41:37.408`

Note:

- I valori degli inserimenti sostituibili MQ si riferiscono all'installazione IBM MQ e al sistema host a cui è associato il comando **runmqsc** .
- **MQPROMPT** è limitato ad un massimo di 256 caratteri quando gli inserimenti vengono espansi. Le espansioni **MQPROMPT** su questo valore comportano il troncamento dell'intera stringa **MQPROMPT** senza le espansioni.

Esempio

Il seguente esempio imposta il prompt su MQSC:

- Linux AIX

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

- Windows

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

AIX Il seguente esempio imposta la variabile **MQPROMPT** su un sistema AIX . La richiesta viene impostata per visualizzare un nome utente (ricavato dalla variabile di ambiente di sistema associata), il nome del gestore code e il nome host IBM MQ (ricavato dagli inserimenti sostituibili 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
```

Il seguente esempio aggiunge una data / ora agli esempi **MQPROMPT** precedenti, ricavati dagli inserimenti sostituibili 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 Abilitazione del richiamo e del completamento dei comandi e dei tasti comando Emacs per `runmqsc`

Utilizzare il prompt dei comandi di **runmqsc** su AIX and Linux per abilitare il richiamo dei comandi, il completamento dei comandi e i tasti di comando Emacs.

Informazioni su questa attività

Su sistemi AIX and Linux , è possibile rendere disponibili le seguenti funzioni aggiuntive dell'editor della riga comandi dal prompt dei comandi **runmqsc** :

- Richiamo dei comandi immessi in precedenza utilizzando il tasto freccia su e il tasto freccia giù
- Completamento automatico per la parola chiave successiva di un comando utilizzando il tasto di tabulazione e la barra spaziatrice
- Tasti di comando Emacs o funzioni di tasti di comando simili

Per utilizzare queste funzioni, è necessario installare la libreria curses. Se la libreria curses non è installata sul proprio sistema, il prompt **runmqsc** non dispone delle funzioni dell'editor della riga comandi e viene visualizzato un messaggio quando viene avviato il prompt dei comandi **runmqsc** . Il nome della libreria curses da installare dipende dalla piattaforma UNIX :

- AIX Su AIX, installare `curses`.

- Linux Su Linux, installare `ncurses`.

Procedura

- Installare `ncurses` o `curses`.

  Nota: Il seguente esempio utilizza le istruzioni per Linux

  Immettere il seguente comando per individuare i package `ncurses` esistenti:

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

  I pacchetti `ncurses` richiesti sono i seguenti:

```
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
```

  È possibile installare tutti i package `ncurses` richiesti elencati nel testo precedente eseguendo il seguente comando:

```
yum install ncurses*
```

- Personalizza le associazioni tasti Emacs.

  È possibile personalizzare le chiavi associate ai comandi. Ad esempio, è possibile associare le chiavi alle associazioni vi invece delle associazioni di tasti Emacs predefinite.

  Le chiavi vengono personalizzate modificando il file `.editrc` memorizzato nella directory home. Per ulteriori informazioni, consultare editrc nelle pagine man FreeBSD .

- Disabilitare il richiamo del comando, il completamento del comando e i tasti comando Emacs.

  A tale scopo, impostare la variabile di ambiente **MQ_OVERRIDE_LIBEDIT_LOAD** su TRUE.

  Questa variabile di ambiente può essere utilizzata come soluzione temporanea quando il prompt dei comandi **runmqsc** visualizza il seguente messaggio informativo:

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

ALW Esecuzione di comandi MQSC da file di testo in `runmqsc`

Se si dispone di comandi lunghi o si sta utilizzando una particolare sequenza di comandi ripetutamente, è possibile utilizzare un file di testo per immettere comandi MQSC. È possibile reindirizzare `stdin` da un file di testo. È anche possibile reindirizzare l'emissione a un file.

Prima di iniziare

Questa attività presuppone che sia stato creato un file di testo contenente i comandi MQSC che si desidera eseguire. Per la sintassi dettagliata e gli esempi di questi file, consultare “Sintassi del file di input MQSC” a pagina 15.

È possibile impostare il prompt dei comandi MQSC su un prompt a scelta utilizzando la variabile di ambiente **MQPROMPT** . Per ulteriori informazioni, consultare “Impostazione del prompt dei comandi MQSC” a pagina 20.

Informazioni su questa attività

L'input per il comando **runmqsc** viene preso dalla *periferica di immissione standard*, indicata anche come `stdin`. In genere questa è la tastiera, ma è possibile specificare che l'input deve provenire da una porta seriale o da un file.

L'output per il comando **runmqsc** viene emesso nella *unità di output standard*, indicata anche come `stdout`. In genere si tratta di una visualizzazione, ma è possibile reindirizzare l'output a una porta seriale o a un file.

Procedura

1. Su un gestore code locale, accertarsi che la sintassi del comando nel file sia corretta senza eseguire i comandi.

   Utilizzare l'indicatore **-v** sul comando **runmqsc** , insieme ad una delle seguenti opzioni:

   - Utilizzare l'opzione **-f** per identificare il nome file del testo di input. Ad esempio:

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

   Non è possibile specificare un gestore code remoto durante la verifica dei comandi. Ciò significa che non è possibile specificare l'indicatore **-w** .

   Il report restituito è simile a quello mostrato nella Figura 2 a pagina 17.
2. Quando la sintassi del comando è corretta, rimuovere l'indicatore **-v** ed eseguire nuovamente il comando **runmqsc** .

   Si noti che è ora possibile specificare un gestore code remoto.

   - Eseguire (ad esempio) il seguente comando:

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

   Figura 1 a pagina 16 mostra un'estrazione da un file di comandi come `myprog.in` e Figura 2 a pagina 17 mostra l'estrazione corrispondente dell'output da un file di report come `results.out`.

Operazioni successive

Per dettagli completi sulla sintassi del comando **runmqsc** , sui parametri facoltativi e sui codici di ritorno, consultare runmqsc (esecuzione di comandi MQSC).

Attività correlate

“Impostazione del prompt dei comandi MQSC” a pagina 20
Su AIX, Linux, and Windows, utilizzare la variabile di ambiente **MQPROMPT** per impostare la richiesta visualizzata quando si esegue il comando **runmqsc** . Ciò rende più semplice vedere di essere in un ambiente MQSC e visualizzare alcuni dettagli dell'ambiente corrente.

“Esecuzione interattiva dei comandi MQSC in runmqsc” a pagina 17
Su AIX, Linux, and Windows, è possibile utilizzare il prompt dei comandi **runmqsc** per immettere i comandi MQSC per un gestore code in modo interattivo. La corsa interattiva è particolarmente indicata per test rapidi.

Riferimenti correlati

Riferimento comandi MQSC

Multi Configurazione automatica da uno script MQSC all'avvio

È possibile configurare il gestore code per applicare automaticamente il contenuto di uno script MQSC o di una serie di script MQSC ad ogni avvio del gestore code.

È possibile utilizzare questa funzionalità per avere una configurazione che può essere modificata e riprodotta automaticamente al successivo riavvio del gestore code. Ad esempio, se lo script o gli script

si trovano su un'unità montata, è possibile disporre di una configurazione centralizzata in cui l'ultima versione viene applicata a ogni gestore code all'avvio.

Uno scenario particolare in cui ciò può essere utile, consiste nel garantire che un cluster uniforme contenga le stesse definizioni su tutti i gestori code nel cluster, disponendo di una singola serie di configurazioni che vengono applicate. Per un esempio, consultare Creazione di un nuovo cluster uniforme.

Prima di iniziare

È possibile utilizzare:

1. Uno script singolo e creare un file di testo utilizzando i comandi MQSC.
2. Una serie di script MQSC:
   - Per identificare una directory in cui esisteranno le configurazioni e
   - In tale indirizzario, creare i file, ognuno con estensione `.mqsc`, ad esempio `queues.mqsc`.

Dal momento che questo script viene riapplicato ad ogni avvio del gestore code, è importante che i comandi possano essere riprodotti. Ad esempio, un comando **DEFINE** deve includere la stringa **REPLACE** , altrimenti il comando viene visualizzato come un errore al secondo avvio del gestore code, poiché l'oggetto già esiste.

Si noti che in uno script MQSC, qualsiasi riga con prefisso * viene considerata come un commento.

Abilitazione della configurazione automatica degli script di MQSC

Importante: Non è necessario immettere comandi per i canali di tipo MQTT, poiché non sono supportati per la configurazione automatica durante l'avvio.

È possibile configurare un nuovo gestore code utilizzando l'indicatore **-ic** per il comando **crtmqm** e puntando a un file specifico o a una directory. Il valore fornito viene memorizzato nel file `qm.ini` nella stanza AutoConfig , come attributo **MQSCConfig**.

È possibile configurare un gestore code esistente per abilitare la configurazione MQSC automatica, aggiungendo l'attributo della stanza AutoConfig **MQSCConfig**, che punta a un file o a una directory validi. Ad esempio:

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

Come funziona la configurazione automatica?

Durante l'avvio del gestore code, la configurazione identificata dall' **AutoConfig** attributo stanza **MQSCConfig** viene passata attraverso la convalida **runmqsc** , per garantire una sintassi valida e quindi memorizzata nella struttura ad albero dei dati del gestore code nella directory `autocfg` come un singolo file `cached.mqsc`.

Quando vengono elaborati più file da una directory, vengono elaborati in ordine alfabetico e se contiene un comando di fine o di uscita MQSC, il resto del contenuto di tale file viene ignorato.

Durante il primo avvio del gestore code, l'impossibilità di leggere il file o la directory o un file con sintassi MQSC non valida impedisce l'avvio del gestore code, con un messaggio di errore appropriato alla console e al log degli errori del gestore code.

Ai successivi riavvii, se il file o la directory a cui si fa riferimento è illeggibile o contiene una sintassi MQSC non valida, viene utilizzato il file precedentemente memorizzato nella cache e un messaggio scritto nel log degli errori del gestore code lo evidenzia.

Al punto in cui il contenuto di `cached.mqsc` viene applicato al gestore code, quando tutti i comandi MQSC sono stati applicati, il gestore code è abilitato per la connessione delle applicazioni. Il log **runmqsc** della configurazione applicata viene memorizzato nella directory degli errori del gestore code, come un file denominato `autocfgmqsc.LOG`.

Inoltre, tutti i comandi MQSC che non vengono completati correttamente, vengono registrati nel log degli errori del gestore code, identificando il motivo per cui il comando ha esito negativo.

Automazione della gestione IBM MQ utilizzando i comandi PCF

Si potrebbe decidere che sarebbe utile per l'installazione automatizzare alcune attività di gestione e monitoraggio. È possibile automatizzare le attività di gestione per gestori code locali e remoti utilizzando i comandi PCF (Programmable Command Format). Questa sezione presuppone che si abbia esperienza nella gestione di oggetti IBM MQ .

Comandi PCF

I comandi PCF (Programmable Command Format) IBM MQ possono essere utilizzati per programmare le attività di gestione in un programma di gestione. In tal modo, da un programma è possibile modificare gli oggetti del gestore code (code, definizioni di processi, elenchi nomi, canali, canali di connessione client, listener, servizi e oggetti delle informazioni di autenticazione) e persino modificare i gestori code stessi.

I comandi PCF coprono la stessa gamma di funzioni fornite dai comandi MQSC. È possibile scrivere un programma per emettere comandi PCF su qualsiasi gestore code nella rete da un singolo nodo. In questo modo, è possibile centralizzare e automatizzare le attività di gestione.

Ogni comando PCF è una struttura dati incorporata nella parte di dati dell'applicazione di un messaggio IBM MQ . Ciascun comando viene inviato al gestore code di destinazione utilizzando la funzione MQI MQPUT nello stesso modo di qualsiasi altro messaggio. Se il server dei comandi è in esecuzione sul gestore code che riceve il messaggio, il server dei comandi lo interpreta come messaggio di comando ed esegue il comando. Per ottenere le risposte, l'applicazione emette una chiamata MQGET e i dati della risposta vengono restituiti in un'altra struttura di dati. La domanda può quindi elaborare la risposta e agire di conseguenza.

Nota: A differenza dei comandi MQSC, i comandi PCF e le relative risposte non sono in un formato di testo leggibile.

Brevemente, queste sono alcune delle cose necessarie per creare un messaggio di comando PCF:

Descrittore messaggio
Si tratta di un descrittore di messaggi IBM MQ standard, in cui:

- Il tipo di messaggio (*MsqType*) è MQMT_REQUEST.
- Il formato del messaggio (*Format*) è MQFMT_ADMIN.

Dati applicazione
Contiene il messaggio PCF che include l'intestazione PCF, in cui:

- Il tipo di messaggio PCF (*Type*) specifica MQCFT_COMMAND.
- L'identificativo del comando specifica il comando, ad esempio, *Change Queue* (MQCMD_CHANGE_Q).

Per una descrizione completa delle strutture di dati PCF e come implementarle, consultare “Introduzione a IBM MQ Programmable Command Formats” a pagina 26.

Attributi oggetto PCF

Gli attributi degli oggetti in PCF non sono limitati a otto caratteri come per i comandi MQSC. Sono mostrati in questa guida in corsivo. Ad esempio, l'equivalente PCF di RQMNAME è *RemoteQMgrName*.

PCF di escape

I PCF di escape sono comandi PCF che contengono comandi MQSC all'interno del testo del messaggio. È possibile utilizzare i PCF per inviare comandi a un gestore code remoto. Per ulteriori informazioni sui PCF di escape, consultare Escape.

Introduzione a IBM MQ Programmable Command Formats

I PCF (Programmable Command Format) definiscono i messaggi di comando e risposta che possono essere scambiati tra un programma e qualsiasi gestore code (che supporta PCF) in una rete. I PCF semplificano la gestione dei gestori code e di altre reti. Possono essere utilizzati per risolvere il problema della gestione complessa di reti distribuite, soprattutto quando le reti crescono in dimensioni e complessità.

I formati di comando programmabili sono supportati su tutte le piattaforme IBM MQ .

Il problema che i comandi PCF risolvono

La gestione delle reti distribuite può diventare complessa. I problemi amministrativi continuano a crescere con l'aumento delle dimensioni e della complessità delle reti.

Esempi di gestione specifici per la messaggistica e l'accodamento includono:

- Gestione delle risorse.

  Ad esempio, creazione ed eliminazione della coda.
- Monitoraggio delle prestazioni.

  Ad esempio, la profondità massima della coda o la frequenza dei messaggi.
- Controllo

  Ad esempio, l'ottimizzazione dei parametri della coda come la profondità massima della coda, la lunghezza massima dei messaggi e l'abilitazione e disabilitazione delle code.
- Instradamento del messaggio.

  Definizione di percorsi alternativi attraverso una rete.

I comandi IBM MQ PCF possono essere utilizzati per semplificare la gestione del gestore code e di altre reti. I comandi PCF consentono di utilizzare una singola applicazione per eseguire la gestione della rete da un singolo gestore code nella rete.

Che cosa sono i PCF?

I PCF definiscono i messaggi di comando e di risposta che possono essere scambiati tra un programma e qualsiasi gestore code (che supporta PCF) in una rete. È possibile utilizzare comandi PCF in un programma applicativo di gestione dei sistemi per la gestione di oggetti IBM MQ : oggetti delle informazioni di autenticazione, canali, listener di canali, elenchi nomi, definizioni di processo, gestori code, code, servizi e classi di memoria. L'applicazione può operare da un singolo punto nella rete per comunicare le informazioni sul comando e sulla risposta con qualsiasi gestore code, locale o remoto, utilizzando il gestore code locale.

Ciascun gestore code dispone di una coda di amministrazione con un nome coda standard e l'applicazione può inviare messaggi di comando PCF a tale coda. Ogni gestore code ha anche un server dei comandi per gestire i messaggi di comando dalla coda di amministrazione. I messaggi di comando PCF possono quindi essere elaborati da qualsiasi gestore code nella rete e i dati di risposta possono essere restituiti all'applicazione, utilizzando la coda di risposta specificata. I comandi PCF e i messaggi di risposta vengono inviati e ricevuti utilizzando la normale MQI (Message Queue Interface).

Per un elenco dei comandi PCF disponibili, inclusi i relativi parametri, consultare Definizioni dei formati dei comandi programmabili.

Utilizzo di IBM MQ Programmable Command Format

È possibile utilizzare i PCF in un programma di gestione dei sistemi per la gestione remota IBM MQ .

Questa sezione include:

- “Messaggi di comando PCF” a pagina 27
- “Risposte PCF in IBM MQ” a pagina 29

- z/OS “Extended responses” a pagina 31
- Regole per la denominazione di oggetti IBM MQ
- “Controllo autorizzazione per comandi PCF in IBM MQ” a pagina 33

Messaggi di comando PCF

I messaggi di comando PCF sono costituiti da un'intestazione PCF, parametri identificati in tale intestazione e anche dati di messaggi definiti dall'utente. I messaggi vengono emessi utilizzando le chiamate dell'interfaccia della coda messaggi.

Ogni comando e i relativi parametri vengono inviati come un messaggio di comando separato contenente un'intestazione PCF seguita da un certo numero di strutture di parametro; per i dettagli dell'intestazione PCF, consultare MQCFH - PCF headere per un esempio di una struttura di parametro, consultare MQCFST - PCF string parameter. L'intestazione PCF identifica il comando e il numero di strutture di parametri che seguono nello stesso messaggio. Ogni struttura di parametri fornisce un parametro al comando.

Le risposte ai comandi, generate dal server dei comandi, hanno una struttura simile. C'è un'intestazione PCF, seguita da un numero di strutture di parametri. Le risposte possono essere costituite da più di un messaggio, ma i comandi sono sempre composti da un solo messaggio.

Multi Su Multipiattaforme, la coda a cui vengono inviati i comandi PCF viene sempre denominata SYSTEM.ADMIN.COMMAND.QUEUE.

z/OS Su z/OS, i comandi vengono inviati a SYSTEM.COMMAND.INPUT, sebbene SYSTEM.ADMIN.COMMAND.QUEUE può essere un alias per esso. Il server dei comandi che serve questa coda invia le risposte alla coda definita dai campi *ReplyToQ* e *ReplyToQMgr* nel descrittore del messaggio del messaggio di comando.

Come emettere i messaggi di comando PCF

Utilizzare le normali chiamate MQI (Message Queue Interface), MQPUT, MQGET e così via, per inserire e richiamare il comando PCF e i messaggi di risposta da e verso le relative code.

Nota:

Verificare che il server dei comandi sia in esecuzione sul gestore code di destinazione per il comando PCF da elaborare su tale gestore code.

Per un elenco dei file di intestazione forniti, consultare IBM MQ COPY, header, include e module files.

Descrittori di messaggi per un comando PCF

Il descrittore del messaggio IBM MQ è completamente documentato in MQMD - Descrittore messaggio.

Un messaggio di comando PCF contiene i seguenti campi nel descrittore del messaggio:

Prospetto
Qualsiasi valore valido, come richiesto.

MsgType
Questo campo deve essere MQMT_REQUEST per indicare un messaggio che richiede una risposta.

Scadenza
Qualsiasi valore valido, come richiesto.

Feedback
Imposta su MQFB_NONE

Multi **Codifica**
Se si sta inviando a un sistema IBM MQ for Multiplatforms , impostare questo campo sulla codifica utilizzata per i dati del messaggio. La conversione viene eseguita, se necessario.

Multi **CodedCharSetId**
Se si sta inviando a un sistema IBM MQ for Multiplatforms , impostare questo campo sul CCSID (coded character set identifier) utilizzato per i dati del messaggio. La conversione viene eseguita, se necessario.

Formato
Impostare su MQFMT_ADMIN.

Priorità
Qualsiasi valore valido, come richiesto.

Persistenza
Qualsiasi valore valido, come richiesto.

MsgId
L'applicazione di invio può specificare qualsiasi valore oppure è possibile specificare MQMI_NONE per richiedere al gestore code di creare un identificativo di messaggio univoco.

CorrelId
L'applicazione di invio può specificare qualsiasi valore oppure MQCI_NONE può essere specificato per indicare nessun identificativo di correlazione.

ReplyToQ
Il nome della coda per ricevere la risposta.

ReplyToQMgr
Il nome del gestore code per la risposta (o vuoto).

Campi di contesto del messaggio
Questi campi possono essere impostati su qualsiasi valore valido, come richiesto. Di solito, l'opzione di inserimento del messaggio MQPMO_DEFAULT_CONTEXT viene utilizzata per impostare i campi di contesto del messaggio sui valori predefiniti.

Se si sta utilizzando una struttura MQMD version-2 , è necessario impostare i seguenti campi aggiuntivi:

GroupId
Imposta su MQGI_NONE

MsgSeqNumber
Imposta su 1

Offset
Imposta su 0

MsgFlags
Imposta su MQMF_NONE

OriginalLength
Imposta su MQOL_UNDEFINED

Invio di dati utente

Le strutture PCF possono essere utilizzate anche per inviare dati di messaggi definiti dall'utente. In questo caso, il campo *Format* del descrittore del messaggio deve essere impostato su MQFMT_PCF.

Invio e ricezione di messaggi PCF in una coda specificata

Invio di messaggi PCF a una coda specificata

Per inviare un messaggio a una coda specificata, la chiamata Bag mqPutconverte il contenuto della borsa specificata in un messaggio PCF e invia il messaggio alla coda specificata. Il contenuto della borsa rimane invariato dopo la chiamata.

Come input per questa chiamata, è necessario fornire:

- Un handle di connessione MQI.

- Un handle di oggetto per la coda su cui deve essere inserito il messaggio.
- Un descrizione del messaggio. Per ulteriori informazioni sul descrittore del messaggio, consultare MQMD - Descrittore del messaggio.
- Inserire le opzioni del messaggio utilizzando la struttura MQPMO. Per ulteriori informazioni sulla struttura MQPM, consultare MQPMO - Put - message options.
- L'handle della borsa da convertire in un messaggio.

  Nota: Se il contenitore contiene un messaggio di gestione e la chiamata di interrogazione mqAddè stata utilizzata per inserire i valori nel contenitore, il valore dell'elemento di dati MQIASY_COMMAND deve essere un comando INQUIRE riconosciuto da MQAI.

Per una descrizione completa della chiamata Bag mqPut, consultare mqPutBag.

Ricezione di messaggi PCF da una coda specificata

Per ricevere un messaggio da una coda specificata, la chiamata al contenitore mqGetriceve un messaggio PCF da una coda specificata e converte i dati del messaggio in un contenitore dati.

Come input per questa chiamata, è necessario fornire:

- Un handle di connessione MQI.
- Una gestione oggetto della coda da cui deve essere letto il messaggio.
- Un descrizione del messaggio. Nella struttura MQMD, il parametro **Format** deve essere MQFMT_ADMIN, MQFMT_EVENT o MQFMT_PCF.

  Nota: Se il messaggio viene ricevuto all'interno di un'unità di lavoro (con l'opzione MQGMO_SYNCPOINT) e il formato del messaggio non è supportato, è possibile eseguire il backout dell'unità di lavoro. Il messaggio viene quindi reintegrato nella coda e può essere recuperato utilizzando la chiamata MQGET invece della chiamata Bag mqGet. Per ulteriori informazioni sul descrittore del messaggio, consultare MQGMO - Get - message options.
- Richiamare le opzioni del messaggio utilizzando la struttura MQGMO. Per ulteriori informazioni sulla struttura MQGMO, consultare MQMD - Message Descriptor.
- L'handle del contenitore per contenere il messaggio convertito.

Per una descrizione completa della chiamata Bag mqGet, consultare mqGetBag.

Risposte PCF in IBM MQ

In risposta a ciascun comando, il server dei comandi genera uno o più messaggi di risposta. Un messaggio di risposta ha un formato simile a un messaggio di comando.

L'intestazione PCF ha lo stesso valore identificativo del comando a cui è una risposta (consultare MQCFH - intestazione PCF per i dettagli). L'identificativo del messaggio e l'identificativo di correlazione vengono impostati in base alle opzioni di report della richiesta.

Se il tipo di intestazione PCF del messaggio di comando è MQCFT_COMMAND, vengono generate solo le risposte standard. Tali comandi sono supportati solo su Multiplatforms. Older applications do not support PCF on z/OS ; the IBM MQ Windows Explorer is one such application (however, the IBM WebSphere MQ 6.0 or later IBM MQ Explorer does support PCF on z/OS).

Se il tipo di intestazione PCF del messaggio di comando è MQCFT_COMMAND_XR, vengono generate risposte estese o standard. Tali comandi sono supportati su z/OS e alcune piattaforme multiple. I comandi emessi su z/OS generano solo risposte estese.

Se un singolo comando specifica un nome oggetto generico, viene restituita una risposta separata nel proprio messaggio per ogni oggetto corrispondente. Per la generazione della risposta, un singolo comando con un nome generico viene considerato come più comandi singoli (ad eccezione del campo di controllo MQCFC_LAST o di MQCFC_NOT_LAST). In caso contrario, un messaggio di comando genera un messaggio di risposta.

Alcune risposte PCF potrebbero restituire una struttura anche quando non è richiesta. Questa struttura viene mostrata nella definizione della risposta (Definizioni dei formati di comando programmabili) come *sempre restituito*. Il motivo per cui, per queste risposte, è necessario denominare gli oggetti nella risposta per identificare quale oggetto vengono applicati i dati.

Descrittori di messaggi per una risposta

Un messaggio di risposta ha i seguenti campi nel descrittore del messaggio:

MsgType
Questo campo è MQMT_REPLY.

MsgId
Questo campo viene generato dal gestore code.

CorrelId
Questo campo viene creato in base alle opzioni di prospetto del messaggio di comando.

Formato
Questo campo è MQFMT_ADMIN.

Codifica
Impostare su MQENC_NATIVE.

CodedCharSetId
Impostare su MQCCSI_Q_MGR.

Persistenza
Lo stesso come nel messaggio di comando.

Priorità
Lo stesso come nel messaggio di comando.

La risposta viene generata con MQPMO_PASS_IDENTITY_CONTEXT.

Multi *Risposte standard*
Messaggi di comando con un tipo di intestazione MQCFT_COMMAND, vengono generate risposte standard. Tali comandi sono supportati solo su Multiplatforms.

Esistono tre tipi di risposta standard:

- Risposta OK
- Risposta di errore
- Risposta dati

Risposta OK

Questa risposta è costituita da un messaggio che inizia con un'intestazione del formato del comando, con un campo *CompCode* MQCC_OK o MQCC_WARNING.

Per MQCC_OK, *Reason* è MQRC_NONE.

Per MQCC_WARNING, *Reason* identifica la natura dell'avvertenza. In questo caso, l'intestazione del formato del comando potrebbe essere seguita da una o più strutture di parametri di avvertenza appropriate a questo codice di errore.

In entrambi i casi, per un comando inquire potrebbero seguire ulteriori strutture di parametri, come descritto nelle seguenti sezioni.

Risposta di errore

Se il comando ha un errore, vengono inviati uno o più messaggi di risposta di errore (più di uno potrebbe essere inviato anche per un comando che normalmente avrebbe un solo messaggio di risposta). Questi messaggi di risposta di errore hanno MQCFC_LAST o MQCFC_NOT_LAST impostati come appropriato.

Ogni messaggio inizia con un'intestazione del formato della risposta, con un valore *CompCode* MQCC_FAILED e un campo *Reason* che identifica il particolare errore. In generale, ogni messaggio descrive un errore differente. Inoltre, ogni messaggio ha zero o una (mai più di una) struttura di parametri di errore dopo l'intestazione. Questa struttura di parametri, se presente, è una struttura MQCFIN, con un campo *Parameter* che contiene uno dei seguenti:

- ID_PARAMETER_MQIACF

  Il campo *Value* nella struttura è l'identificativo del parametro che era in errore (ad esempio, MQCA_Q_NAME).

- ID_ERRORE_MQIACF

  Questo valore viene utilizzato con un valore *Reason* (nell'intestazione del formato del comando) di MQRC_UNEXPECTED_ERROR. Il campo *Value* nella struttura MQCFIN è il codice motivo non previsto ricevuto dal server dei comandi.

- MQIACF_SELECTOR

  Questo valore si verifica se una struttura di elenco (MQCFIL) inviata con il comando contiene un selettore duplicato o uno non valido. Il campo *Reason* nell'intestazione del formato del comando identifica l'errore e il campo *Value* nella struttura MQCFIN è il valore del parametro nella struttura MQCFIL del comando che era in errore.

- ERRORE_MQIACF_OFFSET

  Questo valore si verifica quando si verifica un errore di confronto dati nel comando Canale di ping. Il campo *Value* nella struttura è l'offset dell'errore di confronto del canale di ping.

- ID_MQIA_CODED_CHAR_SET_

  Questo valore si verifica quando il CCSID (coded character set identifier) nel descrittore del messaggio del comando PCF in entrata non corrisponde a quello del gestore code di destinazione. Il campo *Value* nella struttura è l'identificativo della serie di caratteri codificati del gestore code.

L'ultimo (o unico) messaggio di risposta di errore è una risposta di riepilogo, con un campo *CompCode* di MQCC_FAILED e un campo *Reason* di MQRCCF_COMMAND_FAILED. Questo messaggio non ha una struttura di parametri che segue l'intestazione.

Risposta dati

Questa risposta consiste in una risposta OK (come descritto in precedenza) a un comando inquire. La risposta OK è seguita da ulteriori strutture contenenti i dati richiesti come descritto in Definizioni dei formati di comando programmabili.

Le applicazioni non devono dipendere da queste strutture di parametri aggiuntive che vengono restituite in un ordine particolare.

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.

Controllo autorizzazione per comandi PCF in IBM MQ

Quando viene elaborato un comando PCF, il *UserIdentifier* dal descrittore del messaggio nel messaggio di comando viene utilizzato per i controlli di autorizzazione dell'oggetto IBM MQ richiesti. Il controllo dell'autorizzazione viene implementato in maniera diversa su ciascuna piattaforma, come descritto in questo argomento.

I controlli vengono eseguiti sul sistema su cui viene elaborato il comando; pertanto, questo ID utente deve esistere sul sistema di destinazione e disporre delle autorizzazioni richieste per elaborare il comando. Se il messaggio proviene da un sistema remoto, un modo per ottenere l'ID esistente sul sistema di destinazione è avere un ID utente corrispondente sia sul sistema locale che su quello remoto.

Nota: z/OS Per informazioni sul controllo delle autorizzazioni su z/OS, vedere Attività 1: Identificazione dei parametri del sistema z/OS.

IBM MQ for IBM i

IBM i

Per elaborare qualsiasi comando PCF, l'ID utente deve disporre dell'autorizzazione *dsp* per l'oggetto IBM MQ sul sistema di destinazione.

Inoltre, vengono eseguiti controlli dell'autorizzazione dell'oggetto IBM MQ per alcuni comandi PCF, come mostrato nella Tabella 2 a pagina 34.

Nella maggior parte dei casi questi controlli sono gli stessi dei controlli eseguiti dai comandi CL IBM MQ equivalenti emessi su un sistema locale. Consultare Impostazione della sicurezza su IBM i , per ulteriori informazioni sulla corrispondenza tra le autorizzazioni IBM MQ e le autorizzazioni di sistema IBM i e i requisiti di autorizzazione per i comandi CL IBM MQ . I dettagli sulla sicurezza relativi alle uscite sono forniti nella documentazione Sicurezza a livello di collegamento utilizzando un'uscita di sicurezza .

Per elaborare uno qualsiasi dei seguenti comandi l'ID utente deve essere un membro del profilo gruppo QMQMADM:

- Ping canale
- Modifica canale
- Copia canale
- Creazione canale
- Eliminazione canale
- Reimposta canale
- Risolvi canale
- Avvio canale
- Arresta canale
- Avvio iniziatore canale
- Avvio listener canale

IBM MQ for UNIX, Linux, and Windows

ALW

Per elaborare qualsiasi comando PCF, l'ID utente deve disporre dell'autorizzazione *dsp* per l'oggetto gestore code sul sistema di destinazione. Inoltre, vengono eseguiti controlli dell'autorizzazione dell'oggetto IBM MQ per alcuni comandi PCF, come mostrato nella Tabella 2 a pagina 34.

Per elaborare uno dei seguenti comandi l'ID utente deve appartenere al gruppo *mqm*.

Nota: Solo per Windows , l'ID utente può appartenere al gruppo *Amministratori* o al gruppo *mqm*.

- Modifica canale
- Copia canale
- Creazione canale
- Eliminazione canale
- Ping canale
- Reimposta canale
- Avvio canale
- Arresta canale
- Avvio iniziatore canale
- Avvio listener canale
- Risolvi canale
- Reimposta cluster
- Aggiornamento cluster
- Sospensione gestore code
- Ripristino gestore code

Autorizzazioni oggetto IBM MQ per Multiplatforms

Multi

Tabella 2. Autorizzazioni oggetto

Comando	**IBM MQ Autorizzazione di oggetti**	**Autorizzazione classe (per tipo di oggetto)**
Modifica informazioni di autenticazione	dsp e chg	n/a
Modifica canale	dsp e chg	n/a
Modifica listener canale	dsp e chg	n/a
Modifica canale di connessione client	dsp e chg	n/a
Modifica elenco nomi	dsp e chg	n/a
Modifica processo	dsp e chg	n/a
Modifica coda	dsp e chg	n/a
Modifica gestore code	chg *vedere Nota 3 e Nota 5*	n/a
Modifica servizio	dsp e chg	n/a
Cancellazione coda	clr	n/a

Tabella 2. Autorizzazioni oggetto (Continua)		
Comando	**IBM MQ Autorizzazione di oggetti**	**Autorizzazione classe (per tipo di oggetto)**
Copia informazioni di autenticazione	dsp	crt
Copia informazioni di autenticazione (Sostituisci) *vedere la Nota 1*	*da:* dsp *a* chg	crt
Copia canale	dsp	crt
Copia canale (Sostituisci) *vedere la nota 1*	*da:* dsp *a* chg	crt
Copia listener canale	dsp	crt
Copiare il listener del canale (Sostituisci) *vedere la Nota 1*	*da:* dsp *a* chg	crt
Copia canale di connessione client	dsp	crt
Copia canale di connessione client (Sostituisci) *vedere la nota 1*	*da:* dsp *a* chg	crt
Copia elenco nomi	dsp	crt
Copia elenco nomi (Sostituisci) *vedere la nota 1*	*da:* dsp *a* dsp e chg	crt
Copia processo	dsp	crt
Copia processo (Sostituisci) *vedere Nota 1*	*da:* dsp *a* chg	crt
Copia coda	dsp	crt
Copia coda (Sostituisci) *vedere la nota 1*	*da:* dsp *a* dsp e chg	crt
Creazione informazioni di autenticazione	*(informazioni di autenticazione predefinite del sistema)* dsp	crt
Crea informazioni di autenticazione (Sostituisci) *vedere la nota 1*	*(informazioni di autenticazione predefinite del sistema)* dsp *per:* chg	crt
Creazione canale	*(canale predefinito del sistema)* dsp	crt
Crea canale (Sostituisci) *vedere la Nota 1*	*(canale predefinito di sistema)* dsp *in:* chg	crt
Crea listener del canale	*(listener predefinito di sistema)* dsp	crt
Crea listener canale (Sostituisci) *vedere la nota 1*	*(listener predefinito di sistema)* dsp *in:* chg	crt
Crea canale di connessione client	*(canale predefinito del sistema)* dsp	crt
Crea canale di connessione client (Sostituisci) *vedere la nota 1*	*(canale predefinito di sistema)* dsp *in:* chg	crt
Creazione elenco nomi	*(elenco nomi predefinito di sistema)* dsp	crt

Tabella 2. Autorizzazioni oggetto (Continua)		
Comando	**IBM MQ Autorizzazione di oggetti**	**Autorizzazione classe (per tipo di oggetto)**
Crea elenco nomi (sostituisci) *vedere la Nota 1*	*(elenco nomi predefinito di sistema)* dsp *su:* dsp e chg	crt
Creazione processo	*(processo predefinito del sistema)* dsp	crt
Crea processo (Sostituisci) *vedere la Nota 1*	*(processo predefinito di sistema)* dsp *to:* chg	crt
Creazione coda	*(coda predefinita di sistema)* dsp	crt
Crea coda (Sostituisci) *vedere la Nota 1*	*(coda predefinita del sistema)* dsp *a:* dsp e chg	crt
Crea Servizio	*(coda predefinita di sistema)* dsp	crt
Crea servizio (Sostituisci) *vedere la Nota 1*	*(coda predefinita di sistema)* dsp *in:* chg	crt
Eliminazione informazioni di autenticazione	dsp e dlt	n/a
Eliminare il record di autorizzazione	*(oggetto gestore code)* chg *vedere la nota 4*	*vedere la Nota 4*
Eliminazione canale	dsp e dlt	n/a
Elimina listener canale	dsp e dlt	n/a
Elimina canale di connessione client	dsp e dlt	n/a
Eliminazione elenco nomi	dsp e dlt	n/a
Eliminazione processo	dsp e dlt	n/a
Eliminazione coda	dsp e dlt	n/a
Elimina servizio	dsp e dlt	n/a
Interrogazione informazioni di autenticazione	dsp	n/a
Interrogazione record autorizzazione	*vedere la Nota 4*	*vedere la Nota 4*
Interrogazione canale	dsp	n/a
Interroga listener canale	dsp	n/a
Inquire Channel Status (per **ChannelType** MQCHT_CLSSDR)	inq	n/a
Interroga canale di connessione client	dsp	n/a
Interrogazione elenco nomi	dsp	n/a
Interrogazione processo	dsp	n/a
Interrogazione coda	dsp	n/a
Interrogazione gestore code	*vedere la nota 3*	n/a
Interrogazione stato coda	dsp	n/a

Tabella 2. Autorizzazioni oggetto (Continua)		
Comando	**IBM MQ Autorizzazione di oggetti**	**Autorizzazione classe (per tipo di oggetto)**
Interrogazione servizio	dsp	n/a
Ping canale	ctrl	n/a
Ping gestore code	*vedere la nota 3*	n/a
Aggiornamento gestore code	(oggetto gestore code) chg	n/a
Aggiorna sicurezza (per **SecurityType** MQSECTYPE_SSL)	(oggetto gestore code) chg	n/a
Reimposta canale	ctrlx	n/a
Reimpostazione gestore code	(oggetto gestore code) chg	n/a
Reimposta statistiche coda	dsp e chg	n/a
Risolvi canale	ctrlx	n/a
Imposta record di autorizzazione	*(oggetto gestore code)* chg *vedere la nota 4*	*vedere la Nota 4*
Avvio canale	ctrl	n/a
Arresta canale	ctrl	n/a
Arresta connessione	(oggetto gestore code) chg	n/a
Avvia listener	ctrl	n/a
Arresto del listener	ctrl	n/a
Avvia servizio	ctrl	n/a
Arresta servizio	ctrl	n/a
Esc	*vedere la Nota 2*	*vedere la Nota 2*

Note:

1. Questo comando si applica se l'oggetto da sostituire esiste, altrimenti il controllo dell'autorizzazione è quello per la creazione o la copia senza sostituzione.
2. L'autorizzazione richiesta è determinata dal comando MQSC definito dal testo di escape ed è equivalente a uno dei precedenti comandi.
3. Per poter elaborare qualsiasi comando PCF, l'ID utente deve disporre dell'autorizzazione dsp per l'oggetto gestore code sul sistema di destinazione.
4. Questo comando PCF è autorizzato a meno che il server dei comandi non sia stato avviato con il parametro -a. Per impostazione predefinita, il server dei comandi viene avviato all'avvio del gestore code e senza il parametro -a. Per ulteriori informazioni, consultare Programmable command formats reference.
5. La concessione dell'autorizzazione chg di un ID utente per un gestore code consente di impostare i record di autorizzazione per tutti i gruppi e gli utenti. Non concedere questa autorizzazione agli utenti ordinari o alle applicazioni.

IBM MQ fornisce anche alcuni punti di uscita di sicurezza del canale in modo da poter fornire i propri programmi di uscita utente per il controllo di sicurezza. Per ulteriori informazioni, consultare Visualizzazione di un canale.

Multi Utilizzo di MQAI per semplificare l'utilizzo dei PCF

IBM MQ Administration Interface (MQAI) è un'interfaccia di programmazione per IBM MQ disponibile su AIX, IBM i, Linux, e Windows. Esegue le attività di gestione su un gestore code IBM MQ utilizzando i contenitori di dati per gestire le proprietà (o i parametri) degli oggetti in modo più semplice rispetto all'utilizzo di PCF (Programmable Command Format).

MQAI esegue le attività di amministrazione su un gestore code mediante l'utilizzo di *bag di dati*. I bag di dati consentono di gestire le proprietà (o i parametri) degli oggetti in modo più semplice rispetto all'uso di PCF.

I vantaggi dell'utilizzo di MQAI sono i seguenti:

Semplificare l'utilizzo dei messaggi PCF
MQAI è un modo più semplice per gestire IBM MQ. Se si utilizza MQAI, non è necessario scrivere i propri messaggi PCF. Ciò evita i problemi associati alle strutture dati complesse.

Per passare i parametri nei programmi scritti utilizzando le chiamate MQI, il messaggio PCF deve contenere il comando e i dettagli della stringa o dei dati interi. Per creare questa configurazione manualmente, è necessario aggiungere diverse istruzioni nel programma per ogni struttura e allocare spazio di memoria. Questo compito può essere lungo e laborioso.

I programmi scritti utilizzando MQAI inoltrano i parametri nella serie di dati appropriata ed è necessaria una sola istruzione per ogni struttura. L'utilizzo dei contenitori di dati MQAI elimina la necessità di gestire gli array e allocare la memoria e fornisce un certo grado di isolamento dai dettagli del PCF.

Gestire più facilmente le condizioni di errore
È difficile ottenere codici di ritorno dai comandi PCF. MQAI rende più semplice per il programma gestire le condizioni di errore.

Scambio di dati tra applicazioni
I dati dell'applicazione vengono inviati in formato PCF e compressi e decompressi da MQAI. Se i dati del messaggio sono costituiti da numeri interi e stringhe di caratteri, è possibile utilizzare MQAI per sfruttare la conversione dei dati integrata IBM MQ per i dati PCF. Ciò evita la necessità di scrivere uscite di conversione dati.

Una volta creato e popolato il contenitore dati, è possibile inviare un messaggio di comando di gestione al server dei comandi di un gestore code, utilizzando la chiamata `mqExecute` . Questa chiamata attende i messaggi di risposta. La chiamata `mqExecute` gestisce lo scambio con il server dei comandi e restituisce le risposte in una *serie di risposte*.

Esempi di utilizzo di MQAI

I seguenti programmi di esempio dimostrano l'utilizzo di MQAI per eseguire le diverse attività:

- `amqsaicq.c`: creare una coda locale.
- `amqsaiem.c`: visualizza gli eventi sullo schermo utilizzando un semplice controllo eventi.
- `amqsailq.c`: stampare un elenco di tutte le code locali e le relative profondità correnti.
- `amqsaicl.c`: stampare un elenco di tutti i canali e dei loro tipi.

Creazione dell'applicazione MQAI

Per creare la propria applicazione utilizzando MQAI, è necessario collegarsi alle stesse librerie di IBM MQ. Per informazioni su come creare le applicazioni IBM MQ , consultare Creazione di un'applicazione procedurale.

Suggerimenti e consigli per configurare IBM MQ utilizzando MQAI

MQAI utilizza i messaggi PCF per inviare i comandi di gestione al server dei comandi anziché gestire direttamente il server dei comandi stesso. Suggerimenti per la configurazione di IBM MQ utilizzando MQAI sono disponibili in “Suggerimenti e consigli per l'utilizzo di MQAI per configurare IBM MQ” a pagina 39.

Riferimenti correlati

IBM MQ Administration Interface

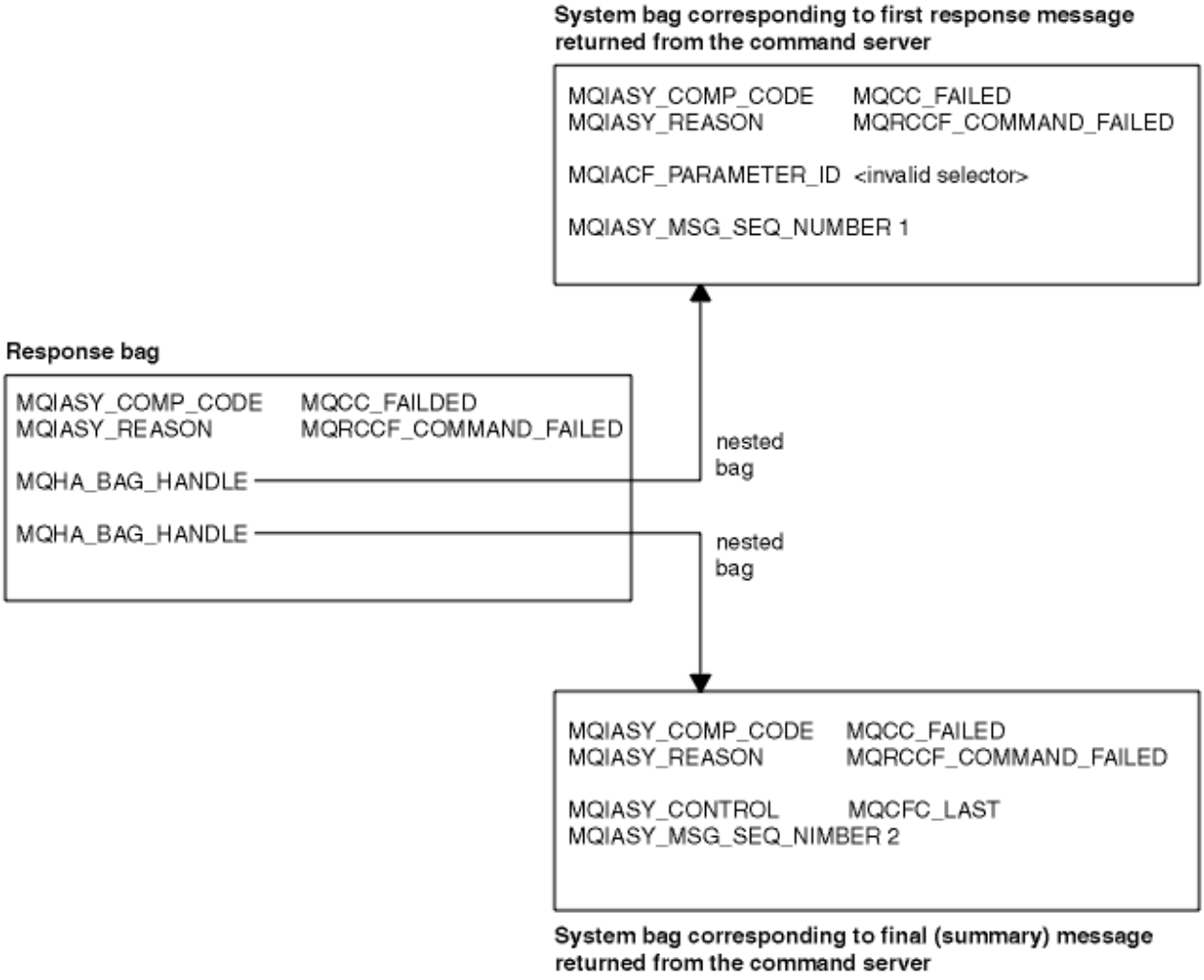
Multi Suggerimenti e consigli per l'utilizzo di MQAI per configurare IBM MQ

MQAI (IBM MQ Administration Interface) utilizza i messaggi PCF per inviare i comandi di gestione al server dei comandi piuttosto che gestire direttamente il server dei comandi stesso. Di seguito sono riportati alcuni consigli per configurare IBM MQ utilizzando MQAI.

- Le stringhe di caratteri in IBM MQ sono spazi vuoti riempiti a lunghezza fissa. Utilizzando C, le stringhe con terminazione null possono normalmente essere fornite come parametri di immissione per le interfacce di programmazione IBM MQ .
- Per cancellare il valore di un attributo stringa, impostarlo su un singolo carattere vuoto piuttosto che su una stringa vuota.
- Considerare in anticipo gli attributi che si desidera modificare e analizzare solo questi attributi.
- Alcuni attributi non possono essere modificati, ad esempio un nome coda o un tipo di canale. Accertarsi di tentare di modificare solo gli attributi che possono essere modificati. Fare riferimento all'elenco di parametri obbligatori e facoltativi per l'oggetto di modifica PCF specifico. Consultare Definitions of the Programmable Command Formats.
- Se una chiamata MQAI ha esito negativo, alcuni dettagli dell'errore vengono restituiti alla serie di risposte. Ulteriori dettagli possono essere trovati in un contenitore nidificato a cui può accedere il selettore MQHA_BAG_HANDLE. Ad esempio, se una chiamata mqExecute ha esito negativo con un codice di errore MQRCCF_COMMAND_FAILED, queste informazioni vengono restituite nella serie di risposte. Un motivo possibile per questo codice di errore è che un selettore specificato non era valido per il tipo di messaggio di comando e questo dettaglio di informazioni si trova in un contenitore nidificato a cui può accedere un gestore contenitore.

  Per ulteriori informazioni su MQExecute, consultare “Invio dei comandi di gestione al server dei comandi qm utilizzando la chiamata mqExecute” a pagina 73

  Il seguente diagramma mostra questo scenario:



Multi Argomenti MQAI avanzati

Informazioni sull'indicizzazione, sulla conversione dei dati e sull'uso del descrittore del messaggio

Indicizzazione
Gli indici vengono utilizzati durante la sostituzione o la rimozione di elementi di dati esistenti da un contenitore per preservare l'ordine di inserimento.

Conversione dati
Le stringhe contenute in un contenitore di dati MQAI possono essere in una serie di caratteri codificati e possono essere convertite utilizzando la chiamata mqSetInteger .

Utilizzo del descrittore del messaggio
MQAI genera un descrittore di messaggi impostato su un valore iniziale quando viene creato il contenitore di dati.

Multi *Indicizzazione in MQAI*

Gli indici vengono utilizzati durante la sostituzione o la rimozione di elementi dati esistenti da un contenitore. Esistono tre tipi di indicizzazione, che consentono di richiamare facilmente gli elementi di dati.

Ogni selettore e valore in un elemento dati in un contenitore ha tre numeri di indice associati:

- L'indice relativo ad altri elementi che hanno lo stesso selettore.
- L'indice relativo alla categoria del selettore (utente o sistema) a cui appartiene l'elemento.
- L'indice relativo a tutti gli elementi dati nel contenitore (utente e sistema).

Ciò consente l'indicizzazione da parte dei selettori utente, dei selettori di sistema o di entrambi, come mostrato in Figura 3 a pagina 41.

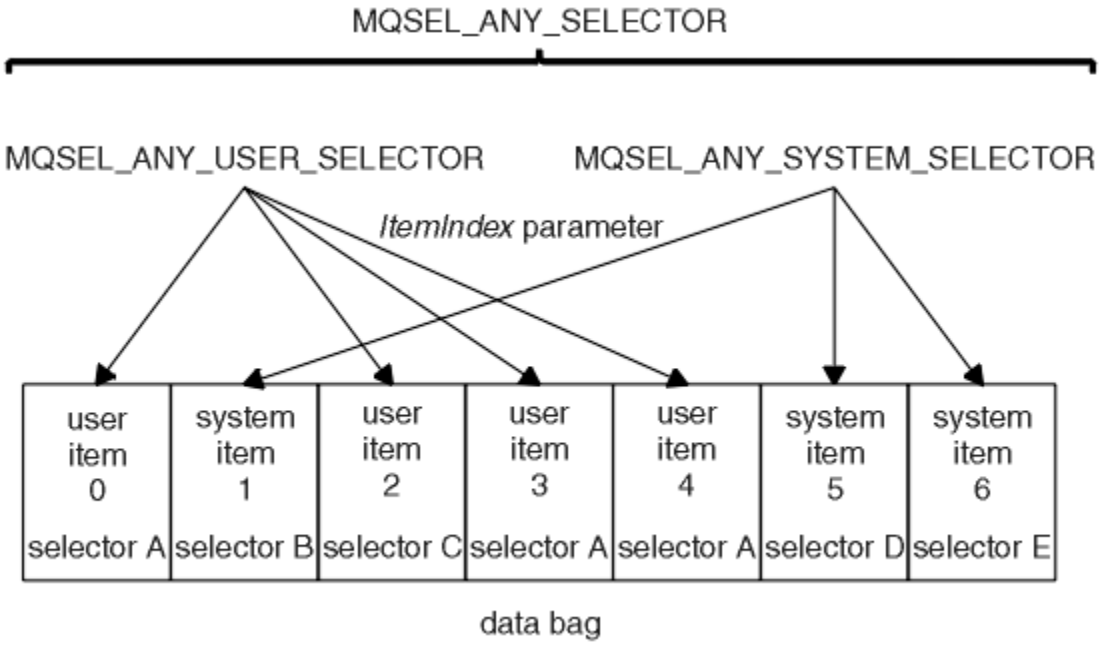


Figura 3. Indicizzazione

In Figura 3 a pagina 41, l'elemento utente 3 (selettore A) può essere indicato dalle seguenti coppie di indici:

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

L'indice è basato su zero come un array in C; se ci sono 'n' ricorrenze, l'indice varia da zero a'n-1', senza interruzioni.

Gli indici vengono utilizzati durante la sostituzione o la rimozione di elementi dati esistenti da un contenitore. Quando viene utilizzato in questo modo, l'ordine di inserimento viene conservato, ma gli indici di altri elementi dati possono essere influenzati. Per esempi, consultare “Modifica delle informazioni all'interno di un contenitore” a pagina 70 e “Eliminazione di elementi dati” a pagina 72.

I tre tipi di indicizzazione consentono un facile richiamo degli elementi dati. Ad esempio, se ci sono tre istanze di un particolare selettore in un contenitore, la chiamata mqCountItems può contare il numero di istanze di quel selettore e le chiamate mqInquire* possono specificare sia il selettore che l'indice per interrogare solo tali valori. Ciò è utile per gli attributi che possono avere un elenco di valori come alcune delle uscite sui canali.

Multi *Elaborazione di conversione dati in MQAI*

Le stringhe contenute in un contenitore di dati MQAI possono essere in una varietà di serie di caratteri codificati. Queste stringhe possono essere convertite utilizzando la chiamata numero intero mqSet.

Come i messaggi PCF, le stringhe contenute in un contenitore di dati MQAI possono trovarsi in una varietà di serie di caratteri codificati. Di solito, tutte le stringhe in un messaggio PCF si trovano nella stessa serie di caratteri codificati, ovvero la stessa serie del gestore code.

Ogni elemento stringa in un contenitore di dati contiene due valori; la stringa stessa e il CCSID. La stringa aggiunta al contenitore si ottiene dal parametro **Buffer** della chiamata mqAddString o mqSetString . Il CCSID viene ottenuto dalla voce di sistema contenente un selettore di MQIASY_CODED_CHAR_SET_ID. Questo è noto come *bag CCSID* e può essere modificato utilizzando la chiamata mqSetInteger .

Quando si interroga il valore di una stringa contenuta in un contenitore di dati, il CCSID è un parametro di emissione dalla chiamata.

Tabella 3 a pagina 42 mostra le regole applicate durante la conversione dei bag di dati in messaggi e viceversa:

Tabella 3. Elaborazione CCSID

Chiamata MQAI	**CCSID**	**Input da richiamare**	**Output da richiamare**
mqBagToBuffer	CCSID serie (1)	Ignorato	Invariato
mqBagToBuffer	CCSID stringa nel contenitore	Utilizzato	Invariato
mqBagToBuffer	CCSID stringa nel buffer	Non applicabile	Copiato da CCSID stringa nel contenitore
mqBufferToBag	CCSID serie (1)	Ignorato	Invariato
mqBufferToBag	CCSID stringa nel buffer	Utilizzato	Invariato
mqBufferToBagmqBufferToBag	CCSID stringa nel contenitore	Non applicabile	Copiato da CCSID stringa nel buffer
mqPutSacchetto	CCSID MQMD	Utilizzato	Non modificato (2)
mqPutSacchetto	CCSID serie (1)	Ignorato	Invariato
mqPutSacchetto	CCSID stringa nel contenitore	Utilizzato	Invariato
mqPutSacchetto	CCSID stringa nel messaggio inviato	Non applicabile	Copiato da CCSID stringa nel contenitore
mqGetBag	CCSID MQMD	Utilizzato per la conversione dei dati del messaggio	Impostato su CCSID dei dati restituiti (3)
mqGetBag	CCSID serie (1)	Ignorato	Invariato
mqGetBag	CCSID stringa nel messaggio	Utilizzato	Invariato
mqGetBag	CCSID stringa nel contenitore	Non applicabile	Copiato dai CCSID stringa nel messaggio
mqExecute	CCSID della serie di richiesta	Utilizzato per MQMD del messaggio di richiesta (4)	Invariato
mqExecute	CCSID della serie di risposte	Utilizzato per la conversione dati del messaggio di risposta (4)	Impostato su CCSID dei dati restituiti (3)
mqExecute	CCSID stringa nella serie di richieste	Utilizzato per il messaggio di richiesta	Invariato
mqExecute	CCSID stringa nel bag di risposta	Non applicabile	Copiato da CCSID stringa nel messaggio di risposta

Note:

1. Bag CCSID è l'elemento di sistema con selettore MQIASY_CODED_CHAR_SET_ID.

2. MQCCSI_Q_MGR è stato modificato nel CCSID del gestore code effettivo.
3. Se viene richiesta la conversione dei dati, il CCSID dei dati restituiti è uguale al valore di emissione. Se la conversione dei dati non è richiesta, il CCSID dei dati restituiti è uguale al valore del messaggio. Notare che non viene restituito alcun messaggio se la conversione dei dati è richiesta ma non riesce.
4. Se il CCSID è MQCCSI_DEFAULT, viene utilizzato il CCSID del gestore code.

Concetti correlati

"Conversione dati tra serie di caratteri codificati" a pagina 205
I dati dei messaggi nei formati definiti da IBM MQ (noti anche come formati integrati) possono essere convertiti dal gestore code da una serie di caratteri codificati a un'altra, a condizione che entrambe le serie di caratteri siano relative a una singola lingua o a un gruppo di lingue simili.

"Il file ccsid_part2.tbl" a pagina 207
Il file `ccsid_part2.tbl` viene utilizzato per fornire ulteriori informazioni CCSID. Il file `ccsid_part2.tbl` sostituisce il file `ccsid.tbl` utilizzato prima di IBM MQ 9.0.

Multi *Utilizzo del descrittore del messaggio in MQAI*

Il descrittore del messaggio generato da MQAI viene impostato su un valore iniziale quando viene creato il contenitore di dati.

Il tipo di comando PCF si ottiene dalla voce di sistema con il selettore MQIASY_TYPE. Quando si crea un contenitore dati, il valore iniziale di questo elemento viene impostato in base al tipo di contenitore creato:

Tabella 4. Tipo di comando PCF

Tipo di sacchetto	**Valore iniziale dell'elemento MQIASY_TYPE**
BAG MQCBO_ADMIN_	COMANDO MQCFT
BAG MQCBO_COMMAND_	COMANDO MQCFT
MQCBO_ *	UTENTE MQCFT

Quando MQAI genera un descrittore del messaggio, i valori utilizzati nei parametri **Format** e **MsgType** dipendono dal valore dell'elemento di sistema con il selettore MQIASY_TYPE come mostrato in Tabella 4 a pagina 43.

Tabella 5. Formato e parametri MsgType di MQMD

Tipo di comando PCF	**Formato**	**MsgType**
COMANDO MQCFT	MMQFMT_ADMIN	MQMT_REQUEST
REPORT MQCFT	MMQFMT_ADMIN	REPORT MQMT
MQCF_XX_ENCODE_CASE_ONE risposta	MMQFMT_ADMIN	MQMT_REPLY
MQCF_TRACE_ROUTE	MMQFMT_ADMIN	MQMT_DATAGRAM
EVENTO MQCFT	EVENTO MQFMT	MQMT_DATAGRAM
MQCFT_ *	MQFMT_PCF	MQMT_DATAGRAM

Tabella 5 a pagina 43 mostra che, se si crea un contenitore di gestione o un contenitore di comandi, il *Format* del descrittore del messaggio è MQFMT_ADMIN e il *MsgType* è MQMT_REQUEST. Questo è adatto per un messaggio di richiesta PCF inviato al server dei comandi quando è prevista una risposta.

Altri parametri nel descrittore del messaggio assumono i valori mostrati in Tabella 6 a pagina 44.

Tabella 6. Valori del descrittore del messaggio	
Parametro	**Valore**
StrucId	ID_STRUC_MQMD
Version	MQMD_VERSION_1
Report	MQRO_NONE
MsgType	Vedere Tabella 5 a pagina 43
Expiry	30 secondi (nota “1” a pagina 44)
Feedback	MQFB_NONE
Encoding	MQEN_NATIVE
CodedCharSetId	dipende dal CCSID della serie (nota “2” a pagina 44)
Format	Vedere Tabella 5 a pagina 43
Priority	MQPRI_PRIORITY_AS_Q_DEF
Persistence	MQPER_NOT_PERSISTENT
MsgId	MQMI_NONE
CorrelId	MQCI_NONE
BackoutCount	0
ReplyToQ	vedere nota “3” a pagina 44
ReplyToQMgr	vuoto

Note:

1. Questo valore può essere sovrascritto sulla chiamata mqExecute utilizzando il parametro **OptionsBag** . Per informazioni, consultare mqExecute.
2. Consultare “Elaborazione di conversione dati in MQAI” a pagina 41.
3. Il nome della coda di risposta specificata dall'utente o della coda dinamica temporanea generata da MQAI per i messaggi di tipo MQMT_REQUEST. In caso contrario, lasciare vuoto.

Multi Programma C di esempio per la creazione di una coda locale (amqsaicq.c)

Il programma C di esempio `amqsaicq.c` crea una coda locale utilizzando 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 Programma C di esempio per la visualizzazione di eventi utilizzando un controllo eventi (amqsaiem.c)

Il programma C di esempio amqsaiem.c illustra un controllo eventi di base utilizzando 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 Programma C di esempio per l'interrogazione degli oggetti canale (amqsaicl.c)

Il programma C di esempio `amqsaicl.c` interroga gli oggetti canale utilizzando 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 Programma C di esempio per l'interrogazione delle code e delle informazioni di stampa (amqsailq.c)

Il programma C di esempio `amqsailq.c` interroga la profondità corrente delle code locali utilizzando 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 Data bag e MQAI

Un contenitore di dati è un mezzo per gestire le proprietà o i parametri degli oggetti utilizzando MQAI (IBM MQ Administration Interface).

Borse dati

- La serie di dati contiene zero o più *elementi dati*. Questi elementi di dati vengono ordinati all'interno del contenitore quando vengono inseriti nel contenitore. Questo è chiamato *ordine di inserimento*. Ogni elemento dati contiene un *selettore* che identifica l'elemento dati e un *valore* di tale elemento dati che può essere un numero intero, un numero intero a 64 bit, un filtro di numeri interi, una stringa, un filtro di stringhe, una stringa di byte, un filtro di stringhe di byte o un handle di un altro contenitore. Gli elementi dati sono descritti in dettaglio in “Tipi di elementi dati disponibili in MQAI” a pagina 67

  Esistono due tipi di selettori: *selettori utente* e *selettori di sistema*. Questi sono descritti in Selettori MQAI. I selettori sono di solito univoci, ma è possibile avere più valori per lo stesso selettore. In questo caso, un *indice* identifica la particolare ricorrenza del selettore richiesta. Gli indici sono descritti in “Indicizzazione in MQAI” a pagina 40.

  Una gerarchia di questi concetti è mostrata nella Figura 1.

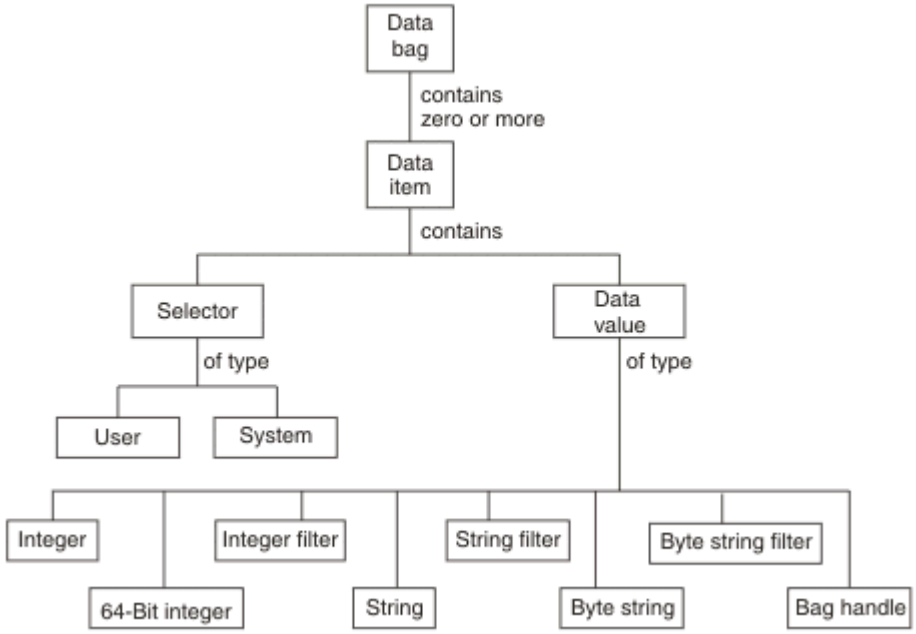


Figura 4. Gerarchia dei concetti MQAI

La gerarchia è stata illustrata in un paragrafo precedente.

Tipi di serie di dati

È possibile scegliere il tipo di contenitore dati che si desidera creare in base all'attività che si desidera eseguire:

serie utente
Un semplice contenitore utilizzato per i dati utente.

serie di gestione
Un contenitore creato per i dati utilizzati per gestire gli oggetti IBM MQ inviando messaggi di gestione a un server dei comandi. Il contenitore di amministrazione implica automaticamente alcune opzioni come descritto in “Creazione ed eliminazione di bag di dati” a pagina 66.

serie di comandi
Un contenitore creato anche per comandi per la gestione di oggetti IBM MQ . Tuttavia, a differenza del contenitore di amministrazione, il contenitore di comandi non implica automaticamente alcune opzioni, sebbene queste opzioni siano disponibili. Per ulteriori informazioni sulle opzioni, consultare “Creazione ed eliminazione di bag di dati” a pagina 66.

bag di gruppo
Un contenitore utilizzato per contenere una serie di elementi dati raggruppati. Le borse di gruppo non possono essere utilizzate per la gestione degli oggetti IBM MQ .

Inoltre, la **serie di sistemi** viene creata da MQAI quando un messaggio di risposta viene restituito dal server dei comandi e inserito in una serie di output dell'utente. Un contenitore di sistema non può essere modificato dall'utente.

I diversi modi di utilizzare i data bag sono elencati in questo argomento:

Utilizzo dei data bag

I diversi modi di utilizzare i data bag sono mostrati nel seguente elenco:

- È possibile creare ed eliminare i data bag “Creazione ed eliminazione di bag di dati” a pagina 66.
- È possibile inviare i dati tra le applicazioni utilizzando i data bag “Inserimento e ricezione di bag di dati mediante MQAI” a pagina 67.

- È possibile aggiungere elementi dati ai bag di dati “Aggiunta di elementi dati ai bag con MQAI” a pagina 68.
- È possibile aggiungere un comando di interrogazione all'interno di un contenitore dati “Aggiunta di un comando di interrogazione ad un contenitore” a pagina 69.
- È possibile richiedere informazioni all'interno dei data bag “Richiesta all'interno dei bag di dati” a pagina 69.
- È possibile contare gli elementi dati all'interno di una serie di dati “Conteggio elementi dati” a pagina 72.
- È possibile modificare le informazioni in un contenitore di dati “Modifica delle informazioni all'interno di un contenitore” a pagina 70.
- È possibile cancellare un contenitore dati “Cancellazione di un contenitore utilizzando la chiamata di borsa mqClear” a pagina 71.
- È possibile troncare una serie di dati “Troncamento di un bag utilizzando la chiamata al bag mqTruncate” a pagina 71.
- È possibile convertire i bag e i buffer “Conversione di buste e buffer” a pagina 71.

Multi ***Creazione ed eliminazione di bag di dati***

Creazione di bag di dati

Per utilizzare MQAI, è necessario prima creare un contenitore di dati utilizzando la chiamata al contenitore mqCreate. Come input per questa chiamata, fornire una o più opzioni per controllare la creazione della borsa.

Il parametro **Options** della chiamata MQCreateBag consente di scegliere se creare un contenitore utente, un contenitore comandi, un contenitore gruppi o un contenitore di gestione.

Per creare una serie di utenti, una serie di comandi o una serie di gruppi, è possibile scegliere una o più ulteriori opzioni per:

- Utilizzare il modulo di elenco quando ci sono due o più ricorrenze adiacenti dello stesso selettore in un contenitore.
- Riordinare le voci di dati man mano che sono aggiunte a un messaggio PCF per garantire che i parametri siano nell'ordine corretto. Per ulteriori informazioni sugli elementi dati, consultare “Tipi di elementi dati disponibili in MQAI” a pagina 67.
- Controllare i valori dei selettori utente per gli elementi che si aggiungono al contenitore.

Le borse di amministrazione implicano automaticamente queste opzioni.

Una serie di dati viene identificata dal relativo handle. L'handle del contenitore viene restituito dal contenitore mqCreatee deve essere fornito su tutte le altre chiamate che utilizzano il contenitore dati.

Per una descrizione completa della chiamata della borsa mqCreate, consultare mqCreateBag.

Eliminazione dei bag di dati

Qualsiasi contenitore di dati creato dall'utente deve essere eliminato anche utilizzando la chiamata mqDeleteBag. Ad esempio, se un contenitore viene creato nel codice utente, deve essere eliminato anche nel codice utente.

Le borse di sistema vengono create ed eliminate automaticamente da MQAI. Per ulteriori informazioni, consultare “Invio dei comandi di gestione al server dei comandi qm utilizzando la chiamata mqExecute” a pagina 73. Il codice utente non può eliminare un contenitore di sistema.

Per una descrizione completa della chiamata della borsa mqDelete, consultare mqDeleteBag.

Multi ***Inserimento e ricezione di bag di dati mediante MQAI***

I dati possono anche essere inviati tra le applicazioni inserendo e ottenendo i bag di dati utilizzando le chiamate di borsa mqPute mqGet. Ciò consente a IBM MQ Administration Interface (MQAI) di gestire il buffer piuttosto che l'applicazione.

La chiamata al contenitore mqPutconverte il contenuto del contenitore specificato in un messaggio PCF e invia il messaggio alla coda specificata e la chiamata al contenitore mqGetrimuove il messaggio dalla coda specificata e lo riconverte in un contenitore di dati. Pertanto, la chiamata al sacchetto mqPutè l'equivalente della chiamata mqBagToBuffer seguita da MQPUT e il sacchetto mqGetè l'equivalente della chiamata MQGET seguita da mqBufferToBag.

Per ulteriori informazioni sull'invio e la ricezione di messaggi PCF in una coda specifica, consultare “Invio e ricezione di messaggi PCF in una coda specificata” a pagina 28

Nota: Se si sceglie di utilizzare la chiamata del sacchetto mqGet, i dettagli PCF all'interno del messaggio devono essere corretti; in caso contrario, si verifica un errore appropriato e il messaggio PCF non viene restituito.

Multi ***Tipi di elementi dati disponibili in MQAI***

Gli elementi dati vengono utilizzati da MQAI (IBM MQ Administration Interface) per popolare i bag di dati quando vengono creati. Questi elementi dati possono essere elementi utente o di sistema.

Questi elementi utente contengono dati utente, ad esempio attributi di oggetti gestiti. Gli elementi di sistema devono essere utilizzati per un maggiore controllo sui messaggi generati: ad esempio, la creazione di intestazioni di messaggi. Per ulteriori informazioni sugli elementi di sistema, consultare “Elementi di sistema e MQAI” a pagina 67.

Tipi di elementi dati

Una volta creato un contenitore di dati, è possibile popolarlo con elementi interi o stringa di caratteri. È possibile richiedere informazioni su tutti e tre i tipi di elemento.

L'elemento dati può essere un numero intero o un elemento stringa di caratteri. Di seguito sono riportati i tipi di elementi dati disponibili all'interno di MQAI:

- Numero intero
- Intero a 64 bit
- Filtro numero intero
- Stringa di caratteri
- Filtro stringa
- Stringa byte
- Filtro stringa di byte
- Maniglia del sacchetto

Utilizzo di elementi dati

Questi sono i seguenti modi di utilizzare gli elementi dati:

- “Conteggio elementi dati” a pagina 72.
- “Eliminazione di elementi dati” a pagina 72.
- “Aggiunta di elementi dati ai bag con MQAI” a pagina 68.
- “Filtro e query degli elementi dati” a pagina 69.

Multi *Elementi di sistema e MQAI*

Gli elementi di sistema possono essere utilizzati da MQAI (IBM MQ Administration Interface) per:

- La creazione di intestazione PCF. Le voci di sistema possono controllare l'identificativo del comando PCF, le opzioni di controllo, il numero di sequenza del messaggio e il tipo di comando.
- Conversione dati. Le voci del sistema gestiscono l'identificativo della serie di caratteri per le voci della stringa di caratteri nel contenitore.

Come tutti gli elementi di dati, gli elementi di sistema sono costituiti da un selettore e un valore. Per informazioni su questi selettori e sulla loro funzione, consultare Selettori MQAI.

Gli elementi di sistema sono univoci. Uno o più elementi di sistema possono essere identificati da un selettore di sistema. C'è solo una ricorrenza di ogni selettore di sistema.

La maggior parte degli elementi del sistema possono essere modificati (consultare “Modifica delle informazioni all'interno di un contenitore” a pagina 70), ma le opzioni di creazione del contenitore non possono essere modificate dall'utente. Non è possibile eliminare elementi di sistema. (Consultare “Eliminazione di elementi dati” a pagina 72.)

Multi *Aggiunta di elementi dati ai bag con MQAI*

Quando un contenitore di dati viene creato utilizzando IBM MQ Administration Interface (MQAI), è possibile popolarlo con elementi dati. Questi elementi dati possono essere elementi utente o di sistema.

Per ulteriori informazioni sugli elementi dati, consultare “Tipi di elementi dati disponibili in MQAI” a pagina 67.

MQAI consente di aggiungere elementi interi, elementi interi a 64 bit, elementi di filtro interi, elementi stringa di caratteri, filtri stringa, elementi stringa di byte e elementi filtro stringa di byte ai bag, come mostrato in Figura 5 a pagina 68. Gli elementi sono identificati da un selettore. Di solito un selettore identifica solo un elemento, ma non sempre è così. Se un elemento dati con il selettore specificato è già presente nel contenitore, un'ulteriore istanza di tale selettore viene aggiunta alla fine del contenitore.

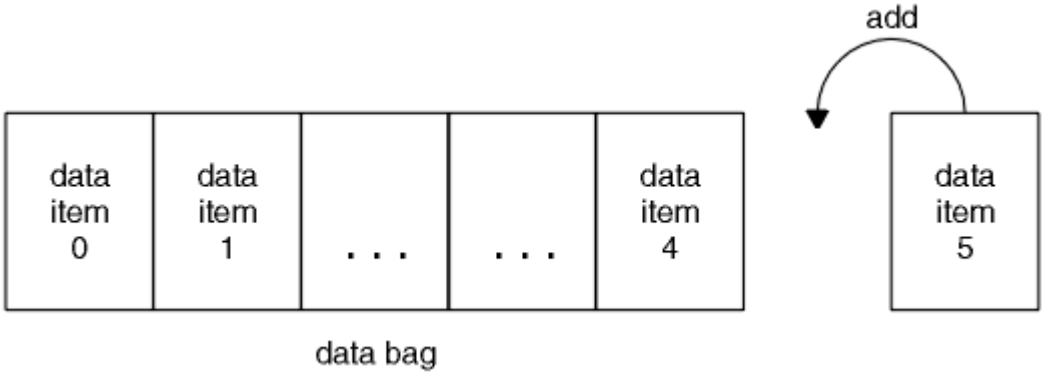


Figura 5. aggiunte degli elementi dati

Aggiungere elementi dati a un contenitore utilizzando le chiamate mqAdd*:

- Per aggiungere elementi interi, utilizzare la chiamata mqAddInteger come descritto in mqAddInteger
- Per aggiungere elementi interi a 64 bit, utilizzare la chiamata mqAddInteger64 come descritto in mqAddInteger64
- Per aggiungere elementi di filtro interi, utilizzare la chiamata mqAddIntegerFilter come descritto in mqAddIntegerFilter
- Per aggiungere elementi stringa di caratteri, utilizzare la chiamata stringa mqAddcome descritto in mqAddString
- Per aggiungere elementi filtro stringa, utilizzare la chiamata mqAddStringFilter come descritto in mqAddStringFilter
- Per aggiungere elementi stringa di byte, utilizzare la chiamata mqAddByteString come descritto in mqAddByteString
- Per aggiungere elementi filtro stringa di byte, utilizzare la chiamata filtro mqAddByteStringcome descritto in mqAddByteStringFilter

Per ulteriori informazioni sull'aggiunta di elementi dati a un contenitore, consultare “Elementi di sistema e MQAI” a pagina 67.

Multi *Aggiunta di un comando di interrogazione ad un contenitore*

La chiamata di interrogazione mqAddviene utilizzata per aggiungere un comando di interrogazione ad un contenitore. La chiamata è specificamente per scopi di amministrazione, quindi può essere utilizzata solo con sacchetti di amministrazione. Consente di specificare i selettori di attributi su cui si desidera eseguire l'interrogazione da IBM MQ.

Per una descrizione completa della chiamata di interrogazione mqAdd, consultare mqAddInquiry.

Multi *Filtro e query degli elementi dati*

Quando si utilizza MQAI per interrogare gli attributi degli oggetti IBM MQ , è possibile controllare i dati restituiti al programma in due modi.

- È possibile ***filtrare*** i dati restituiti utilizzando le chiamate mqAddInteger e mqAddString. Questo approccio consente di specificare una coppia *Selector* e *ItemValue* , ad esempio:

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

  Questo esempio specifica che il tipo di coda (*Selector*) deve essere locale (*ItemValue*) e che questa specifica deve essere corrispondente agli attributi dell'oggetto (in questo caso, una coda) che si sta analizzando.

  Altri attributi che possono essere filtrati corrispondono ai comandi PCF Inquire * che possono essere trovati in “Introduzione a IBM MQ Programmable Command Formats” a pagina 26. Ad esempio, per informazioni sugli attributi di un canale, consultare il comando Interroga canale nella documentazione di questo prodotto. I "Parametri obbligatori" e i "Parametri facoltativi" del comando Inquire Channel identificano i selettori che è possibile utilizzare per il filtro.
- È possibile ***interrogare*** particolari attributi di un oggetto utilizzando la chiamata di interrogazione mqAdd. Specifica il selettore a cui si è interessati. Se non si specifica il selettore, vengono restituiti tutti gli attributi dell'oggetto.

Di seguito viene riportato un esempio di filtro ed esecuzione di query degli attributi di una coda:

```
/* 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, ...)
```

Multi *Richiesta all'interno dei bag di dati*

È possibile richiedere informazioni su:

- Il valore di un elemento intero che utilizza la chiamata numero intero mqInquire. Vedere mqInquireInteger.
- Il valore di un elemento intero a 64 bit utilizzando la chiamata mqInquireInteger64 . Vedere mqInquireInteger64.
- Il valore di un elemento filtro intero che utilizza la chiamata mqInquireIntegerFilter . Vedere mqInquireIntegerFilter.
- Il valore di un elemento stringa di caratteri che utilizza la chiamata stringa mqInquire. Vedere mqInquiremqInquire.

- Il valore di un elemento filtro stringa utilizzando la chiamata mqInquireStringFilter . Vedere mqInquireStringFilter.
- Il valore di un elemento stringa di byte utilizzando la chiamata mqInquireByteString . Vedere mqInquireByteString.
- Il valore di un elemento filtro stringa di byte utilizzando la chiamata Filtro mqInquireByteString. Consultare mqInquireByteStringFilter.
- Il valore di un handle di borsa utilizzando la chiamata di borsa mqInquire. Vedere mqInquireBag.

È anche possibile richiedere informazioni sul tipo (integer, 64 - bit integer, integer filter, character string, string filter, byte string, byte string filter o bag handle) di un elemento specifico utilizzando la chiamata mqInquireItemInfo . Vedere mqInquireItemInfo.

Multi *Modifica delle informazioni all'interno di un contenitore*

MQAI consente di modificare le informazioni in un contenitore utilizzando le chiamate mqSet*. Sarai in grado di:

1. Modificare gli elementi dati all'interno di un contenitore. L'indice consente di sostituire una singola istanza di un parametro identificando la ricorrenza dell'elemento da modificare (consultare Figura 6 a pagina 70).

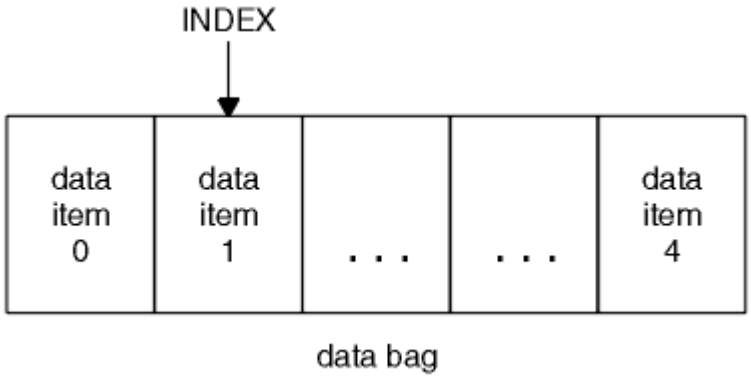


Figura 6. Modifica di un singolo elemento dati

2. Eliminare tutte le ricorrenze esistenti del selettore specificato e aggiungere una nuova ricorrenza alla fine del contenitore. (Consultare Figura 7 a pagina 70.) Un valore di indice speciale consente di sostituire ***tutte le*** istanze di un parametro.

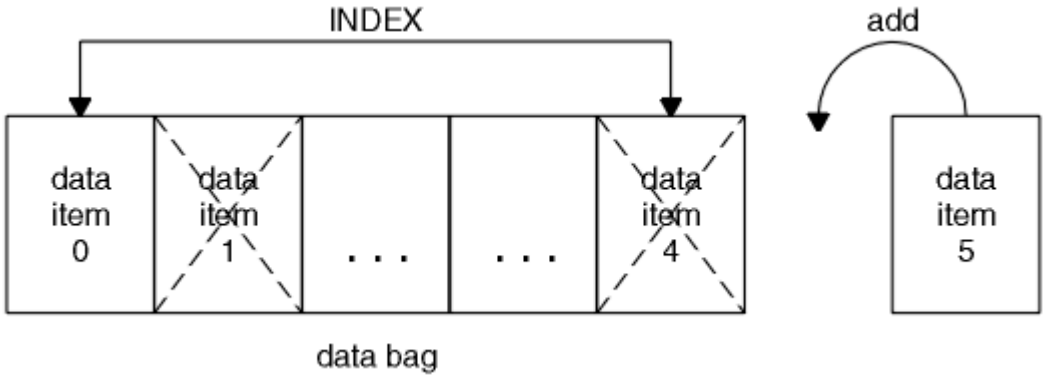


Figura 7. Modifica di tutti gli elementi dati

Nota: L'indice conserva l'ordine di inserimento all'interno del contenitore ma può influire sugli indici di altri elementi di dati.

La chiamata numero intero mqSetconsente di modificare gli elementi interi all'interno di un contenitore. La chiamata mqSetInteger64 consente di modificare elementi interi a 64 bit. La chiamata mqSetIntegerFilter consente di modificare elementi di filtro interi. La chiamata stringa mqSetconsente di modificare gli elementi stringa di caratteri. La chiamata mqSetStringFilter consente di modificare gli

elementi filtro stringa. La chiamata mqSetByteString consente di modificare gli elementi della stringa di byte. La chiamata del filtro mqSetByteStringconsente di modificare gli elementi filtro della stringa di byte. In alternativa, è possibile utilizzare queste chiamate per eliminare tutte le ricorrenze esistenti del selettore specificato e aggiungere una nuova ricorrenza alla fine del contenitore. L'elemento dati può essere un elemento utente o un elemento di sistema.

Per una descrizione completa di queste chiamate, consultare:

- mqSetNumero intero
- mqSetInteger64
- mqSetIntegerFilter
- StringamqSet
- mqSetStringFilter
- mqSetByteString
- mqSetByteStringByteString

Multi *Cancellazione di un contenitore utilizzando la chiamata di borsa mqClear*

La chiamata mqClearBag rimuove tutti gli elementi utente da una serie di utenti e reimposta gli elementi di sistema sui valori iniziali. Vengono eliminati anche i sacchetti di sistema contenuti all'interno del sacchetto.

Per una descrizione completa della chiamata della borsa mqClear, consultare mqClearBag.

Multi *Troncamento di un bag utilizzando la chiamata al bag mqTruncate*

La chiamata mqTruncateriduce il numero di elementi utente in un contenitore utente eliminando gli elementi dalla fine del contenitore, a partire dall'elemento aggiunto più di recente. Ad esempio, può essere utilizzato quando si utilizzano le stesse informazioni di intestazione per generare più di un messaggio.

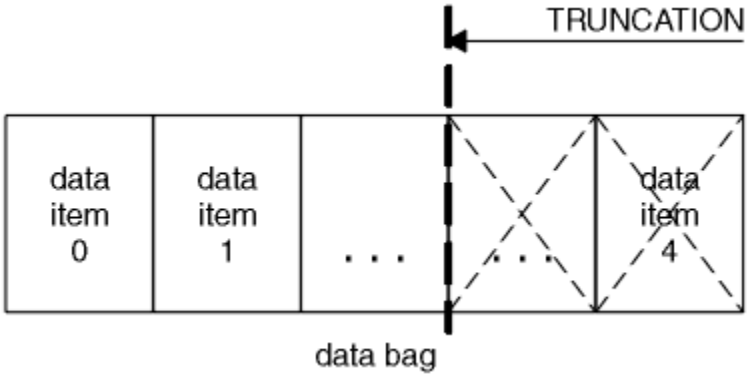


Figura 8. Troncamento di un sacchetto

Per una descrizione completa della chiamata Bag mqTruncate, consultare mqTruncateBag.

Multi *Conversione di buste e buffer*

Per inviare i dati tra le applicazioni, innanzitutto i dati del messaggio vengono inseriti in un contenitore. Quindi, i dati nel contenitore vengono convertiti in un messaggio PCF utilizzando la chiamata mqBagToBuffer . Il messaggio PCF viene inviato alla coda richiesta utilizzando la chiamata MQPUT. Ciò è mostrato in Figura Figura 9 a pagina 72. Per una descrizione completa della chiamata mqBagToBuffer , consultare mqBagToBuffer.

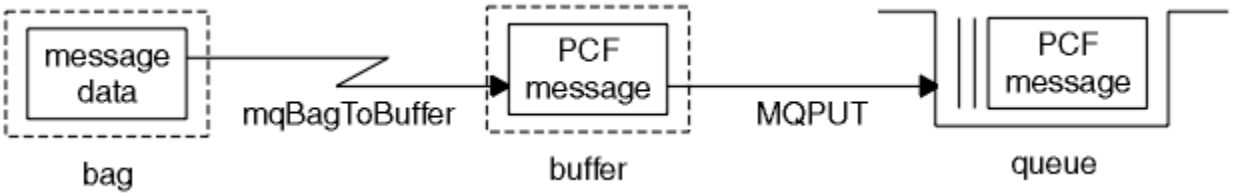


Figura 9. Conversione di sacchetti in messaggi PCF

Per ricevere i dati, il messaggio viene ricevuto in un buffer utilizzando la chiamata MQGET. I dati nel buffer vengono quindi convertiti in un contenitore utilizzando la chiamataToBag di mqBuffer, purché il buffer contenga un messaggio PCF valido. Ciò è mostrato in Figura Figura 10 a pagina 72. Per una descrizione completa della chiamata mqBufferToBag , vedere mqBufferToBag.

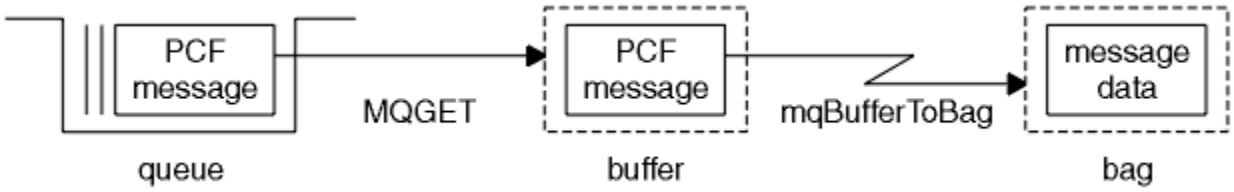


Figura 10. Conversione dei messaggi PCF in formato bag

Multi *Conteggio elementi dati*

La chiamata mqCountelementi conta il numero di elementi utente, elementi di sistema o entrambi, memorizzati in un contenitore di dati e restituisce questo numero. Ad esempio, `mqCountItems( Bag, 7, ...)`, restituisce il numero di elementi nella borsa con un selettore di 7. Può contare gli elementi per singolo selettore, per selettori utente, per selettori di sistema o per tutti i selettori.

Nota: Questa chiamata conta il numero di elementi dati, non il numero di selettori univoci nel contenitore. Un selettore può verificarsi più volte, quindi potrebbero esserci meno selettori univoci nel contenitore rispetto agli elementi di dati.

Per una descrizione completa della chiamata mqCountItem, vedere mqCountItems.

Multi *Eliminazione di elementi dati*

È possibile eliminare gli articoli dalle borse in diversi modi. Sarai in grado di:

- Rimuovere uno o più elementi utente da un contenitore. Per informazioni dettagliate, consultare “Eliminazione di elementi dati da un contenitore utilizzando la chiamata di elemento mqDelete” a pagina 72.
- Eliminare tutti gli elementi utente da un contenitore, ossia cancellare un contenitore. Per informazioni dettagliate, consultare “Cancellazione di un contenitore utilizzando la chiamata di borsa mqClear” a pagina 71.
- Eliminare gli elementi utente dalla fine di un contenitore, ovvero troncare un contenitore. Per informazioni dettagliate, consultare “Troncamento di un bag utilizzando la chiamata al bag mqTruncate” a pagina 71.

Multi *Eliminazione di elementi dati da un contenitore utilizzando la chiamata di elemento mqDelete*

La chiamata di elemento mqDeleterimuove uno o più elementi utente da un contenitore. L'indice viene utilizzato per eliminare:

1. Una singola ricorrenza del selettore specificato. (Consultare Figura 11 a pagina 73.)

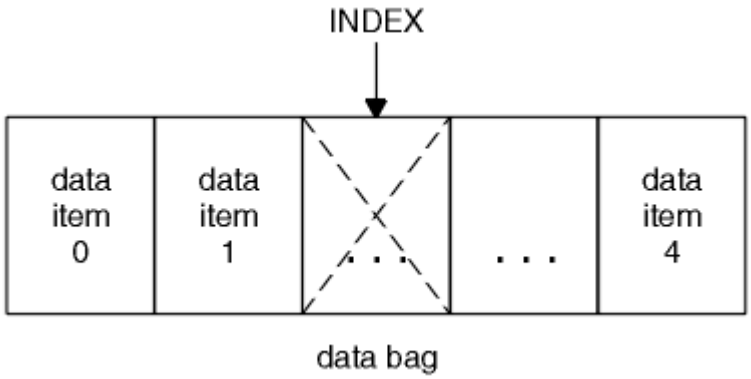


Figura 11. Eliminazione di un singolo elemento dati

o

2. Tutte le ricorrenze del selettore specificato. (Consultare Figura 12 a pagina 73.)

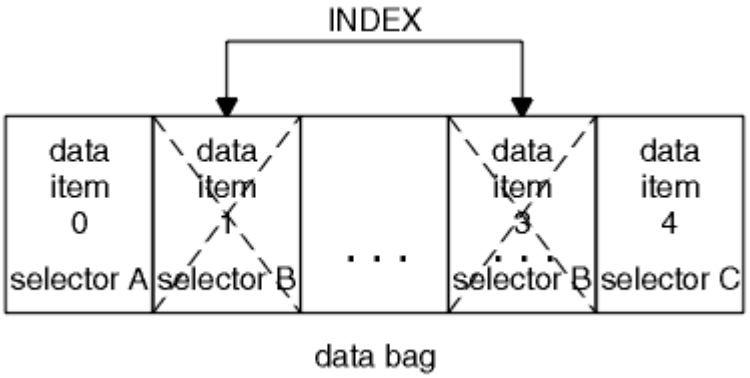


Figura 12. Eliminazione di tutti gli elementi dati

Nota: L'indice conserva l'ordine di inserimento all'interno del contenitore ma può influire sugli indici di altri elementi di dati. Ad esempio, la chiamata dell'elemento mqDeletenon conserva i valori di indice degli elementi dati che seguono l'elemento eliminato perché gli indici sono riorganizzati per colmare il divario che rimane dall'elemento eliminato.

Per una descrizione completa della chiamata mqDeleteItem, consultare mqDeleteItem.

Multi Invio dei comandi di gestione al server dei comandi qm utilizzando la chiamata mqExecute

Quando una serie di dati è stata creata e popolata, è possibile inviare un messaggio di comando di gestione al server dei comandi di un gestore code utilizzando la chiamata mqExecute . Questa operazione gestisce lo scambio con il server dei comandi e restituisce le risposte in un contenitore.

Una volta creato e popolato il contenitore dati, è possibile inviare un messaggio di comando di gestione al server dei comandi di un gestore code. Il modo più semplice per eseguire questa operazione è utilizzando la chiamata mqExecute . La chiamata mqExecute invia un messaggio di comando di gestione come messaggio non persistente e attende eventuali risposte. Le risposte vengono restituite in un contenitore di risposte. Queste potrebbero contenere informazioni relative agli attributi relativi a diversi oggetti IBM MQ o una serie di messaggi di risposta di errore PCF, ad esempio. Pertanto, il contenitore della risposta potrebbe contenere solo un codice di ritorno o potrebbe contenere *contenitori nidificati*.

I messaggi di risposta vengono inseriti nei bag di sistema creati dal sistema. Ad esempio, per richieste relative ai nomi degli oggetti, viene creata una serie di sistemi per contenere tali nomi oggetto e la serie viene inserita nella serie utente. Le maniglie di queste borse vengono quindi inserite nel sacchetto di risposta e al sacchetto nidificato può accedere il selettore MQHA_BAG_HANDLE. Il contenitore di sistema rimane nella memoria, se non viene eliminato, fino a quando il contenitore di risposta non viene eliminato.

Il concetto di *nidificazione* viene mostrato in Figura 13 a pagina 74.

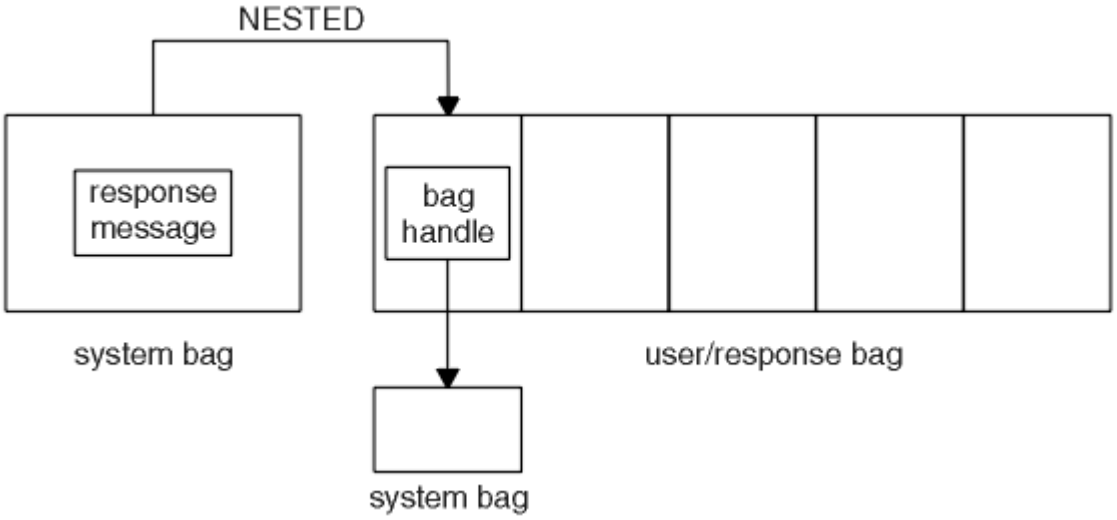


Figura 13. Nidificazione

Come input per la chiamata mqExecute , è necessario fornire:

- Un handle di connessione MQI.
- Il comando da eseguire. Deve essere uno dei valori MQCMD_ *.

  Nota: Se questo valore non viene riconosciuto da MQAI, viene comunque accettato. Tuttavia, se la chiamata mqAddInquiry è stata utilizzata per inserire i valori nel contenitore, questo parametro deve essere un comando INQUIRE riconosciuto da MQAI. Ovvero, il parametro deve essere nel formato MQCMD_INQUIRE_ *.
- Facoltativamente, un handle del contenitore contenente le opzioni che controllano l'elaborazione della chiamata. Questo è anche il punto in cui è possibile specificare il tempo massimo in millisecondi che MQAI deve attendere per ogni messaggio di risposta.
- Un handle del contenitore di gestione che contiene i dettagli del comando di gestione da immettere.
- Un handle del contenitore risposte che riceve i messaggi di risposta.

I seguenti handle sono facoltativi:

- Un handle di oggetto della coda in cui deve essere inserito il comando di gestione.

  Se non viene specificato alcun handle di oggetto, il comando di gestione viene posizionato su SYSTEM.ADMIN.COMMAND.QUEUE appartenente al Gestore code attualmente connesso. Questa è l'opzione predefinita.
- Una gestione oggetto della coda in cui devono essere inseriti i messaggi di risposta.

  È possibile scegliere di inserire i messaggi di risposta in una coda dinamica creata automaticamente da MQAI. La coda creata esiste solo per la durata della chiamata e viene eliminata dall'MQAI all'uscita dalla chiamata mqExecute .

Per esempi di utilizzo della chiamata mqExecute , consultare Codice di esempio

Amministrazione mediante REST API

È possibile utilizzare administrative REST API per amministrare gli oggetti IBM MQ , come i gestori code e le code, gli agent e i trasferimenti Managed File Transfer . Le informazioni vengono inviate e ricevute da administrative REST API in formato JSON. Queste API RESTful possono aiutarti a integrare la gestione IBM MQ negli strumenti DevOps e di automazione.

Prima di iniziare

Nota: V 9.4.0 administrative REST API non è disponibile in un'installazione IBM MQ Web Server autonoma. Per ulteriori informazioni relative alle opzioni di installazione per il componente IBM MQ che esegue administrative REST API, consultare IBM MQ Console e REST API.

Per informazioni di riferimento sulle risorse REST disponibili, consultare Il riferimento administrative REST API.

Procedura

Introduzione a administrative REST API

Inizia rapidamente con administrative REST API e prova alcune richieste di esempio utilizzando cURL per creare, aggiornare, visualizzare ed eliminare una coda.

Prima di iniziare

Per iniziare a utilizzare administrative REST API, gli esempi in questa attività hanno i requisiti seguenti:

- Gli esempi utilizzano cURL per effettuare richieste REST per visualizzare informazioni sui gestori code sul sistema e per creare una coda, aggiornare, visualizzare ed eliminare una coda. Pertanto, per completare questa attività è necessario che cURL sia installato sul sistema.
- Per completare questa attività, è necessario essere un utente con particolari privilegi in modo da poter utilizzare il comando **dspmqweb**:
  - z/OS Su z/OS, è necessario disporre dell'autorità per eseguire il comando **dspmqweb** e l'accesso in scrittura al file `mqwebuser.xml`.
  - Multi Su tutti gli altri sistemi operativi, è necessario essere un utente privilegiato.

  IBM i Su IBM i, i comandi devono essere in esecuzione in QSHELL.

Procedura

1. Verificare di aver configurato il server mqweb per l'utilizzo da parte di administrative REST API, administrative REST API per MFT, messaging REST APIo IBM MQ Console.

   Per ulteriori informazioni sulla configurazione del server mqweb con un registro di base, consultare Configurazione di base per il server mqweb.
2. z/OS

   Su z/OS, impostare la variabile di ambiente WLP_USER_DIR in modo da poter utilizzare il comando **dspmqweb** . Impostare la variabile in modo che punti alla configurazione del server mqweb immettendo il seguente comando:

   ```
   export WLP_USER_DIR=WLP_user_directory
   ```

   dove *WLP_user_directory* è il nome della directory passata a `crtmqweb`. Ad esempio:

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

Per ulteriori informazioni, consultare Creazione del server mqweb.

3. Determinare l'URL REST API immettendo il seguente comando:

```
dspmqweb status
```

Gli esempi riportati di seguito presuppongono che l'URL REST API sia l'URL predefinito `https://localhost:9443/ibmmq/rest/v1/`. Se l'URL è diverso da quello predefinito, sostituirlo nella seguente procedura.

4. Prova una richiesta GET nella risorsa `qmgr` utilizzando l'autenticazione di base con l'utente `mqadmin` :

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

5. Creare, visualizzare, modificare ed eliminare una coda utilizzando la risorsa `mqsc` :

Questo esempio utilizza un gestore code `QM1`. Creare un gestore code con lo stesso nome o sostituire un gestore code esistente sul proprio sistema.

a) Effettuare una richiesta POST sulla risorsa `mqsc` per creare la coda locale:

Nel corpo della richiesta, il nome della nuova coda è impostato su `Q1`. Viene utilizzata l'autenticazione di base e viene impostata un'intestazione HTTP `ibm-mq-rest-csrf-token` con un valore arbitrario nella richiesta REST cURL . Questa intestazione aggiuntiva è richiesta per le richieste POST, PATCH e 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) Effettuare una richiesta POST sulla risorsa `mqsc` per visualizzare la coda locale creata al passo “5.a” a pagina 76:

```
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) Effettuare una richiesta POST sulla risorsa di `mqsc` per aggiornare la descrizione della coda:

```
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) Effettuare una richiesta POST sulla risorsa `mqsc` per visualizzare la nuova descrizione della coda. Specificare l'attributo **`responseParameters`** nel corpo della richiesta in modo che la risposta includa il campo della descrizione:

```
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) Effettuare una richiesta POST sulla risorsa di `mqsc` per eliminare la coda:

```
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) Effettuare una richiesta POST sulla risorsa `mqsc` per provare che la coda è stata eliminata:

```
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\"}"
```

Operazioni successive

- Gli esempi utilizzano l'autenticazione di base per proteggere la richiesta. È possibile utilizzare invece l'autenticazione basata su token o sul client. Per ulteriori informazioni, consultare Utilizzo dell'autenticazione del certificato client con la REST API e la IBM MQ Console e Utilizzo dell'autenticazione basata su token con la REST API.
- Ulteriori informazioni sull'utilizzo di administrative REST API e sulla creazione di URL con parametri di query: “Utilizzo di administrative REST API” a pagina 78.
- Sfogliare le informazioni di riferimento per le risorse administrative REST API disponibili e tutti i parametri di query facoltativi disponibili: administrative REST API reference.
- Informazioni su come utilizzare administrative REST API per gestire oggetti IBM MQ su sistemi remoti: “Gestione remota mediante REST API” a pagina 80.
- Informazioni sull'utilizzo di administrative REST API con MFT:“Introduzione a REST API per MFT” a pagina 77.
- Rilevare messaging REST API, un'interfaccia RESTful per la messaggistica IBM MQ : Messaggistica utilizzando REST API.
- Rilevare IBM MQ Console, una GUI basata su browser: “Amministrazione mediante IBM MQ Console” a pagina 91.

Introduzione a REST API per MFT

Inizia rapidamente con administrative REST API per Managed File Transfer e prova alcune richieste di esempio per visualizzare lo stato dell'agente MFT e un elenco dei trasferimenti.

Prima di iniziare

- Gli esempi utilizzano cURL per inviare le richieste REST per visualizzare un elenco di trasferimenti e visualizzare lo stato dell'agent MFT . Pertanto, per completare questa attività è necessario che cURL sia installato sul sistema.
- Per completare questa attività, è necessario essere un utente con particolari privilegi in modo da poter utilizzare il comando **dspmqweb**:
  - z/OS Su z/OS, è necessario disporre dell'autorità per eseguire il comando **dspmqweb** e l'accesso in scrittura al file `mqwebuser.xml`.
  - Multi Su tutti gli altri sistemi operativi, è necessario essere un utente privilegiato.

Procedura

1. Verificare che il server mqweb sia configurato per administrative REST API per MFT:
   - Verificare di aver configurato il server mqweb per l'utilizzo da parte di administrative REST API, administrative REST API per MFT, messaging REST APIo IBM MQ Console.Per ulteriori informazioni sulla configurazione del server mqweb con un registro di base, consultare Configurazione di base per il server mqweb.
   - Se il server mqweb è stato configurato, assicurarsi che il passo 8 di Configurazione di base per il server mqweb sia stato completato per abilitare administrative REST API per MFT.
2. z/OS
   Su z/OS, impostare la variabile di ambiente WLP_USER_DIR in modo da poter utilizzare il comando **dspmqweb** . Impostare la variabile in modo che punti alla configurazione del server mqweb immettendo il seguente comando:

   ```
   export WLP_USER_DIR=WLP_user_directory
   ```

dove *WLP_user_directory* è il nome della directory passata a `crtmqweb`. Ad esempio:

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

Per ulteriori informazioni, consultare Creazione del server mqweb.

3. Determinare l'URL REST API immettendo il seguente comando:

```
dspmqweb status
```

Gli esempi riportati di seguito presuppongono che l'URL REST API sia l'URL predefinito `https://localhost:9443/ibmmq/rest/v1/`. Se l'URL è diverso da quello predefinito, sostituirlo nella seguente procedura.

4. Effettuare una richiesta GET sulla risorsa `agent` per restituire i dettagli di base su tutti gli agenti, inclusi il nome, il tipo e lo stato:

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

5. Creare alcuni trasferimenti da visualizzare utilizzando il comando **fteCreateTransfer** .

Il server mqweb memorizza nella cache le informazioni sui trasferimenti e restituisce tali informazioni quando viene effettuata una richiesta. Questa cache viene reimpostata quando il server mqweb viene riavviato. È possibile verificare se il server è stato riavviato visualizzando i `console.log` e i `messages.log` fileo su z/OS, esaminando l'output dell'attività avviata.

6. Effettuare una richiesta GET sulla risorsa `transfer` per restituire i dettagli di un massimo di quattro trasferimenti effettuati dall'avvio del server mqweb:

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

Operazioni successive

- Gli esempi utilizzano l'autenticazione di base per proteggere la richiesta. È possibile utilizzare invece l'autenticazione basata su token o sul client. Per ulteriori informazioni, consultare Utilizzo dell'autenticazione basata sul token con REST API e Utilizzo dell'autenticazione del certificato client con REST API e IBM MQ Console.
- Ulteriori informazioni sull'utilizzo di administrative REST API e sulla creazione di URL con parametri di query: “Utilizzo di administrative REST API” a pagina 78.
- Sfogliare le informazioni di riferimento per le risorse administrative REST API per MFT disponibili e tutti i parametri di query facoltativi disponibili: administrative REST API reference.
- Rilevare messaging REST API, un'interfaccia RESTful per la messaggistica IBM MQ : Messaggistica utilizzando REST API.
- Rilevare IBM MQ Console, una GUI basata su browser: “Amministrazione mediante IBM MQ Console” a pagina 91.

Utilizzo di administrative REST API

Quando si utilizza administrative REST API, si richiamano i metodi HTTP sugli URL che rappresentano i vari oggetti IBM MQ , come i gestori code o le code. Il metodo HTTP, ad esempio POST, rappresenta il tipo di azione da eseguire sull'oggetto rappresentato dall'URL. Ulteriori informazioni sull'azione potrebbero essere fornite in JSON come parte del payload del metodo HTTP o codificate nei parametri della query. Le informazioni sul risultato dell'esecuzione dell'azione potrebbero essere restituite come corpo della risposta HTTP.

Prima di iniziare

Considerare quanto segue prima di utilizzare administrative REST API:

- È necessario eseguire l'autenticazione con il server mqweb per utilizzare administrative REST API. È possibile eseguire l'autenticazione utilizzando l'autenticazione di base HTTP, l'autenticazione del certificato client o l'autenticazione basata su token. Per ulteriori informazioni su come utilizzare questi metodi di autenticazione, consultare SicurezzaIBM MQ Console e REST API.
- REST API è sensibile al maiuscolo / minuscolo. Ad esempio, un HTTP GET sul seguente URL non visualizza le informazioni se il gestore code è denominato `qmgr1`.

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

- Non tutti i caratteri che possono essere utilizzati nei nomi oggetto IBM MQ possono essere codificati direttamente in un URL. Per codificare correttamente questi caratteri, è necessario utilizzare la codifica URL appropriata:
  - Una barra, /, deve essere codificata come %2F.
  - Un segno di percentuale,%, deve essere codificato come %25.
- A causa del funzionamento di alcuni browser, non denominare gli oggetti utilizzando solo caratteri punto o barra.

Informazioni su questa attività

Quando si usa REST API per eseguire un'azione su un oggetto, è necessario prima costruire un URL per rappresentare tale oggetto. Ogni URL inizia con un prefisso, che descrive a quale nome host e porta inviare la richiesta. Il resto dell'URL descrive un oggetto particolare o una serie di oggetti, noti come una risorsa.

L'azione che deve essere eseguita sulla risorsa definisce se l'URL necessita o meno di parametri di query. Definisce anche il metodo HTTP utilizzato e se le informazioni aggiuntive vengono inviate all'URL, o restituite da esso, in formato JSON. Le informazioni aggiuntive potrebbero far parte della richiesta HTTP o essere restituite come parte della risposta HTTP.

Dopo aver creato l'URL e creato un payload JSON facoltativo per l'invio nella richiesta HTTP, puoi inviare la richiesta HTTP a IBM MQ. È possibile inviare la richiesta utilizzando l'implementazione HTTP integrata nel linguaggio di programmazione scelto. Puoi anche inviare le richieste utilizzando strumenti della riga di comando come cURL, un browser web o un componente aggiuntivo del browser web.

Importante: È necessario, come minimo, eseguire le operazioni “1.a” a pagina 79 e “1.b” a pagina 79.

Procedura

1. Creare l'URL:
   a) Determinare l'URL del prefisso immettendo il seguente comando:

      ```
      dspmqweb status
      ```

      L'URL che vuoi utilizzare include la frase `/ibmmq/rest/` .
   b) Aggiungere la risorsa al percorso URL.

      Sono disponibili le seguenti risorse IBM MQ :

      - /admin/installazione
      - /admin/qmgr
      - /admin/coda
      - /admin/sottoscrizione
      - /admin/canale
      - /action/qmgr/{qmgrname}/mqsc

      Sono disponibili le seguenti risorse Managed File Transfer :

      - /admin/agent

- /admin/trasferimento
- /admin/monitor

Ad esempio, per interagire con i gestori code, aggiungere `/qmgr` all'URL del prefisso per creare il seguente URL:

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

c) Opzionale: Aggiungere ulteriori segmenti di percorso facoltativi all'URL.

Nelle informazioni di riferimento per ciascun tipo di oggetto, i segmenti facoltativi possono essere identificati nell'URL dalle parentesi graffe che lo circondano { }.

Ad esempio, aggiungere il nome gestore code `QM1` all'URL per creare il seguente URL:

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

d) Opzionale: Aggiungere un parametro di query facoltativo all'URL.

Aggiungere un punto interrogativo,?, nome della variabile, segno uguale = e un valore o un elenco di valori per l'URL.

Ad esempio, per richiedere tutti gli attributi del gestore code QM1, creare il seguente URL:

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

e) Aggiungere ulteriori parametri di query facoltativi all'URL.

Aggiungere una e commerciale, &, all'URL e ripetere il passo d.

2. Richiamare il metodo HTTP pertinente sull'URL. Specificare qualsiasi payload JSON facoltativo e fornire le credenziali di sicurezza appropriate per l'autenticazione. Ad esempio:
   - Utilizzare l'implementazione HTTP/REST del linguaggio di programmazione scelto.
   - Utilizza uno strumento come un componente aggiuntivo del browser del client REST o cURL.

Gestione remota mediante REST API

È possibile utilizzare REST API per gestire i gestori code remoti e gli oggetti IBM MQ associati a questi gestori code. Questa gestione remota include gestori code che si trovano nello stesso sistema, ma che non si trovano nella stessa installazione IBM MQ del server mqweb. Pertanto, è possibile utilizzare REST API per gestire l'intera rete IBM MQ con una sola installazione che esegue il server mqweb. Per gestire i gestori code remoti, è necessario configurare il gateway administrative REST API in modo tale che almeno un gestore code nella stessa installazione del server mqweb agisca come gestore code del gateway. Quindi, è possibile specificare il gestore code remoto nell'URL della risorsa REST API per eseguire l'azione amministrativa specificata.

Prima di iniziare

È possibile impedire la gestione remota disabilitando il gateway administrative REST API . Per ulteriori informazioni, vedi Configurazione del gateway administrative REST API.

Per utilizzare il gateway administrative REST API , devono essere soddisfatte le seguenti condizioni:

- Il server mqweb deve essere configurato e avviato. Per ulteriori informazioni sulla configurazione e l'avvio del server mqweb, consultare “Introduzione a administrative REST API” a pagina 75.
- Il gestore code che si desidera configurare come gestore code del gateway deve trovarsi nella stessa installazione del server mqweb.
- Il gestore code remoto che si desidera gestire deve essere IBM MQ 8.0 o successivo.
- È necessario assicurarsi che gli attributi specificati nella richiesta siano validi per il sistema a cui si sta inviando la richiesta. Ad esempio, se il gestore code del gateway si trova su Windows e il gestore code remoto si trova su z/OS, non è possibile richiedere che l'attributo `dataCollection.statistics` venga restituito per una richiesta HTTP GET sulla risorsa queue .

- È necessario assicurarsi che gli attributi specificati nella richiesta siano validi per il livello di IBM MQ a cui si sta inviando la richiesta. Ad esempio, se il gestore code remoto sta eseguendo IBM MQ 8.0, non è possibile richiedere che l'attributo `extended.enableMediaImageOperations` venga restituito per una richiesta HTTP GET sulla risorsa queue .
- È necessario utilizzare una delle seguenti risorse REST supportate:
  - `/queue`
  - `/subscription`
  - `/channel`
  - `/mqsc`
  - `/qmgr`

    La risorsa `/qmgr` restituisce solo un sottoinsieme di attributi quando si interroga un gestore code remoto: `name`, `status.started`, `status.channelInitiatorState`, `status.ldapConnectionState`, `status.connectionCounte` `status.publishSubscribeState`.

Informazioni su questa attività

Per utilizzare il gateway administrative REST API per gestire i gestori code remoti, è necessario preparare i gestori code per la gestione remota. Ovvero, è necessario configurare le code di trasmissione, i listener e i canali mittente e destinatario tra il gestore code del gateway e il gestore code remoto. È quindi possibile inviare una richiesta REST al gestore code remoto specificando il gestore code nell'URL della risorsa. Il gestore code del gateway viene specificato utilizzando il comando **setmqweb** per impostare l'attributo `mqRestGatewayQmgr` sul nome del gestore code del gateway o inviando il nome del gestore code del gateway in un'intestazione inviata con la richiesta. La richiesta viene inviata tramite il gestore code del gateway al gestore code remoto. La risposta viene restituita con un'intestazione che indica il gestore code utilizzato come gestore code del gateway.

Procedura

1. Configurare le comunicazioni tra il gestore code del gateway e i gestori code remoti che si desidera gestire. Questi passi di configurazione sono gli stessi necessari per configurare la gestione remota da parte di runmqsc e PCF.

   Per ulteriori informazioni su questi passi, consultare “Configurazione dei gestori code per la gestione remota” a pagina 199.
2. Configurare la sicurezza sui gestori code remoti:

   a) Verificare che gli ID utente pertinenti esistano sul sistema su cui è in esecuzione il gestore code remoto. L'ID utente che deve esistere sul sistema remoto dipende dal ruolo dell'utente REST API :
   - Se l'utente REST API si trova nel gruppo RO MQWebAdmin o MQWebAdmin, l'ID utente che ha avviato il server mqweb deve esistere sul sistema remoto. Su IBM MQ Appliance, l'utente che avvia il server mqweb è `mqsystem`.
   - Se l'utente REST API fa parte del gruppo MQWebUser , l'ID utente REST API deve esistere sul sistema remoto.

   b) Verificare che agli ID utente pertinenti siano concessi i livelli di autorizzazione necessari per accedere alle risorse REST API appropriate sul gestore code remoto:
   - Autorizzazione per inserire messaggi in `SYSTEM.ADMIN.COMMAND.QUEUE`.
   - Autorizzazione per inserire messaggi in `SYSTEM.REST.REPLY.QUEUE`.
   - Autorizzazione ad accedere alle code di trasmissione definite per la gestione remota.
   - Autorizzazione a visualizzare gli attributi del gestore code.
   - Autorizzazione per eseguire le richieste REST. Per ulteriori informazioni, consultare la sezione Requisiti di sicurezza degli argomenti di riferimento delle risorse REST API.

3. Configurare quale gestore code locale viene utilizzato come gateway. È possibile configurare un gestore code del gateway predefinito, specificare il gestore code del gateway in un'intestazione HTTP o utilizzare una combinazione di entrambi gli approcci:
   - Configurare un gestore code gateway predefinito utilizzando il comando **setmqweb** :

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

     dove *qmgrName* è il nome del gestore code del gateway.

     Questo gestore code gateway viene utilizzato quando si verificano entrambe le seguenti condizioni:
     - Un gestore code non è stato specificato nell'intestazione `ibm-mq-rest-gateway-qmgr` di una richiesta REST.
     - Il gestore code specificato nell'URL della risorsa REST API non è un gestore code locale.
   - Configurare il gestore code gateway su ogni richiesta REST impostando l'intestazione HTTP `ibm-mq-rest-gateway-qmgr` sul nome del gestore code gateway.
4. Includere il nome del gestore code remoto che si desidera gestire nell'URL della risorsa.

   Ad esempio, per ottenere un elenco di code dal gestore code remoto `remoteQM`, utilizzare il seguente URL:

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

Risultati

Viene restituita un'intestazione `ibm-mq-rest-gateway-qmgr` con la risposta REST. Questa intestazione specifica quale gestore code è stato utilizzato come gestore code del gateway.

Se si hanno difficoltà con l'utilizzo di administrative REST API per gestire i gestori code remoti:

- Verificare che il gestore code remoto sia in esecuzione.
- Controllare che il server dei comandi sia in esecuzione sul sistema remoto.
- Verificare che l'intervallo di disconnessione del canale non sia scaduto. Ad esempio, se un canale è stato avviato ma è stato chiuso dopo un certo periodo di tempo. Questo è particolarmente importante se si avviano i canali manualmente.

Esempio

Nel seguente esempio, sono disponibili tre installazioni IBM MQ su due macchine. Su `Machine 1`, sono presenti un `Installation 1` e un `Installation 2`. Su `Machine 2`, è presente un `Installation 3`. Un server mqweb è configurato per `Installation 1`. Esiste un singolo gestore code in ogni installazione e questi gestori code sono configurati per l'amministrazione remota. Ovvero, i seguenti listener, canali e code sono configurati e avviati:

- Sul gestore code QM1, in `Installation 1`, su `Machine 1`:
  - Canale mittente QM1.to.QM2
  - Canale ricevente QM2.to.QM1
  - Canale mittente QM1.to.QM3
  - Canale ricevente QM3.to.QM1
  - Coda di trasmissione QM2
  - Coda di trasmissione QM3
  - Un listener configurato sulla porta 1414
- Sul gestore code QM2, in `Installation 2`, su `Machine 1`:
  - Canale mittente QM2.to.QM1
  - Canale ricevente QM1.to.QM2

  - Coda di trasmissione QM1
  - Un listener configurato sulla porta 1415
- Sul gestore code QM3, in `Installation 3`, su `Machine 2`:
  - Canale mittente QM3.to.QM1
  - Canale ricevente QM1.to.QM3
  - Coda di trasmissione QM1
  - Il listener predefinito

Una coda, `Qon2` , è definita su QM2e una coda `Qon3` è definita su QM3.

L'utente `mquser` è definito su entrambe le macchine, gli è stato concesso il ruolo MQWebAdmin in REST APIe gli è stata concessa l'autorizzazione ad accedere alle code appropriate su ciascun gestore code.

Il comando setmqweb viene utilizzato per configurare il gestore code QM1 come gestore code gateway predefinito.

Il seguente diagramma mostra questa configurazione:

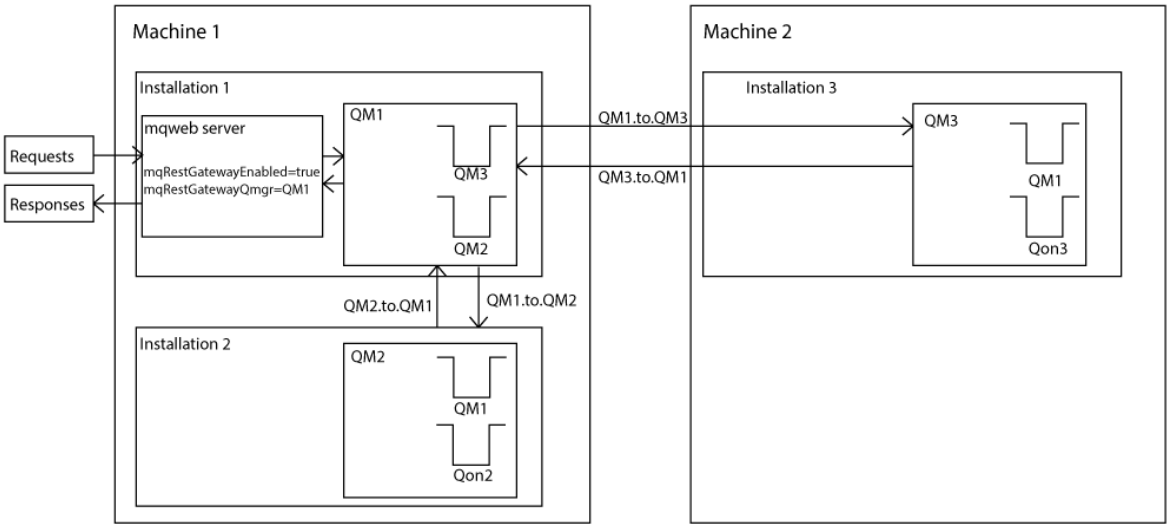


Figura 14. Diagramma di configurazione di esempio per la gestione remota utilizzando REST API.

La seguente richiesta REST viene inviata al server mqweb:

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

Viene ricevuta la seguente risposta:

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

```
    }]
}
```

La seguente richiesta REST viene inviata al server mqweb:

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

Viene ricevuta la seguente risposta:

```
{
   "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"
    }]
}
```

REST API data/ora

Quando le informazioni di data e ora vengono restituite da administrative REST API, vengono restituite in UTC (Coordinated Universal Time) e in un formato impostato.

La data e l'ora vengono restituite nel seguente formato data/ora:

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

Ad esempio, `2012-04-23T18:25:43.000Z`, dove Z indica che il fuso orario è UTC (Coordinated Universal Time).

L'accuratezza di questa data / ora non è garantita. Ad esempio, se il server mqweb non viene avviato nello stesso fuso orario del gestore code specificato nell'URL della risorsa, la data / ora potrebbe non essere accurata. Inoltre, se sono necessarie regolazioni dell'ora legale, la data / ora potrebbe non essere accurata.

REST API gestione degli errori

REST API riporta gli errori restituendo un codice di risposta HTTP appropriato, ad esempio 404 (Non trovato) e una risposta JSON. Qualsiasi codice di risposta HTTP non compreso nell'intervallo 200 - 299 viene considerato un errore.

Il formato della risposta di errore

La risposta è in formato JSON nella codifica UTF-8 . Contiene oggetti JSON nidificati:

- Un oggetto JSON esterno che contiene un singolo array JSON denominato `error`.
- Ogni elemento nell'array è un oggetto JSON che rappresenta le informazioni su un errore. Ogni oggetto JSON contiene le seguenti proprietà:

  il tipo
  Stringa.

  Il tipo di errore.

  messageId
  Stringa.

Un identificativo univoco per il messaggio nel formato MQWBnnnnX. Questo identificativo ha i seguenti elementi:

MQWB
Un prefisso che indica che il messaggio è stato originato nell'API REST IBM MQ .

nnnn
Un numero univoco che identifica il messaggio.

X
Una singola lettera che denota la severità del messaggio:

- I se un messaggio è puramente informativo.
- W se un messaggio indica un problema.
- E se un messaggio indica che si è verificato un errore.
- S se un messaggio indica che si è verificato un errore grave.

messaggio
Stringa.

Una descrizione dell'errore.

spiegazione
Stringa.

Una spiegazione dell'errore.

azione
Stringa.

Una descrizione delle operazioni che è possibile eseguire per risolvere l'errore.

qmgrName
z/OS Questo campo è disponibile solo per z/OS dove il gestore code è un membro del gruppo di condivisione code. È necessario che sia stato specificato il parametro di query facoltativo **commandScope** o l'attributo **queueSharingGroupDisposition** .

Stringa.

Il nome del gestore code che ha rilevato l'errore.

Questo campo non è applicabile per messaging REST API.

completionCode
Questo campo è disponibile solo quando **type** è pcf, javao rest.

Numero.

Il codice di completamento MQ associato all'errore.

reasonCode
Questo campo è disponibile solo quando **type** è pcf, javao rest.

Numero.

Il codice motivo MQ associato all'errore.

eccezioni
Questo campo è disponibile solo quando **type** è java.

Array.

Un array di eccezioni concatenamento Java o JMS. Ogni elemento dell'array di eccezioni contiene un array di stringhe **stackTrace** .

L'array di stringhe **stackTrace** contiene i dettagli di ciascuna eccezione suddivisa in linee.

Errori con i gruppi di condivisione code

z/OS

In un gruppo di condivisione code, è possibile specificare un parametro di query facoltativo di **commandScope** per determinati comandi. Questo parametro consente la propagazione del comando ad altri gestori code nel gruppo di condivisione code. Uno di questi comandi può avere esito negativo indipendentemente, determinando la riuscita di alcuni comandi e l'esito negativo di alcuni comandi per il gruppo di condivisione code.

Nei casi in cui un comando non riesce parzialmente, viene restituito un codice di errore HTTP di 500. Per ogni gestore code che ha generato un errore, le informazioni su tale errore vengono restituite come elemento nell'array JSON `error` . Per ogni gestore code che ha eseguito correttamente il comando, il nome del gestore code viene restituito come un elemento in un array JSON `success` .

Esempi

- Il seguente esempio mostra la risposta di errore a un tentativo di ottenere informazioni su un gestore code che non esiste:

```
"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 Il seguente esempio mostra la risposta di errore ad un tentativo di eliminazione di una coda in un gruppo di condivisione code che non esiste per alcuni gestori code:

```
"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"}]
```

Errori con richieste MFT

Se i servizi MFT REST API non sono abilitati e si richiama MFT REST API, si riceve la seguente eccezione:

```
{"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"
}]}
```

Se i servizi MFT REST API sono abilitati e il gestore code di coordinamento non è impostato nel file `mqwebuser.xml` , si riceve la seguente eccezione:

```
{"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"
}]}
```

Rilevamento REST API

La documentazioni per REST API è disponibile in IBM Documentation e in formato Swagger. Swagger è un approccio comunemente utilizzato per documentare le API REST. La documentazione Swagger per REST API può essere visualizzata abilitando la funzione API Discovery (apiDiscovery) sul server mqweb.

Prima di iniziare

Stabilized

Importante: La funzione apiDiscovery è stata stabilizzata. È ancora possibile utilizzare questa funzione. Attualmente, IBM MQ non supporta l'utilizzo della funzione `mpOpenAPI` .

È necessario abilitare la sicurezza per il server mqweb per la visualizzazione della documentazione Swagger utilizzando API Discovery. Per ulteriori informazioni sulla procedura richiesta per abilitare la sicurezza, consultare IBM MQ Console e REST API sicurezza.

Procedura

1. Individuare il file `mqwebuser.xml` in una delle seguenti directory:
   - ALW *MQ_DATA_PATH*/web/installations/*installationName*/servers/mqweb
   - z/OS *WLP_user_directory*/servers/mqweb

     Dove *WLP_user_directory* è la directory specificata quando lo script **crtmqweb** è stato eseguito per creare la definizione del server mqweb.
2. Aggiungere l'XML appropriato al file `mqwebuser.xml` :
   - Se le tag `<featureManager>` sono presenti nel proprio file `mqwebuser.xml` , aggiungere il seguente XML all'interno delle tag `<featureManager>` :

     ```
     <feature>apiDiscovery-1.0</feature>
     ```
   - Se le tag `<featureManager>` non esistono nel proprio file `mqwebuser.xml` , aggiungere il seguente XML nelle tag `<server>` :

     ```
     <featureManager>
          <feature>apiDiscovery-1.0</feature>
     </featureManager>
     ```
3. Visualizzare la documentazione Swagger utilizzando uno dei seguenti metodi:
   - Visualizzare una pagina Web che è possibile sfogliare e provare REST API immettendo il seguente URL in un browser:

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

Oltre all'autenticazione di ogni richiesta, è necessario includere un'intestazione `ibm-mq-rest-csrf-token` per ogni richiesta POST, PATCH o DELETE. Il contenuto di questa intestazione può essere qualsiasi stringa, incluso uno spazio.

Questa intestazione della richiesta viene utilizzata per confermare che le credenziali utilizzate per autenticare la richiesta vengono utilizzate dal proprietario delle credenziali. Ovvero, il token viene utilizzato per evitare attacchi cross - site request forgery.

- Richiama un singolo documento Swagger 2 che descrive l'intero REST API immettendo un HTTP GET al seguente URL:

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

  Questo documento può essere utilizzato per le applicazioni in cui si desidera navigare in modo programmatico nelle API disponibili.

host
Specifica il nome host o indirizzo IP su cui è disponibile REST API .

Il valore predefinito è `localhost`.

porta
Specifica il numero di porta HTTPS utilizzato da administrative REST API .

Il valore predefinito è 9443.

Se il nome host o il numero di porta viene modificato rispetto al valore predefinito, è possibile determinare i valori corretti dall'URL REST API . Utilizzare il comando **dspmqweb status** per visualizzare l'URL.

Informazioni correlate
dspmqweb status (visualizza lo stato del server mqweb)

REST API supporto lingua nazionale

REST API supporta, con determinate qualifiche, la possibilità di specificare le lingue nazionali come parte di una richiesta HTTP.

Sfondo

Le intestazioni HTTP consentono di specificare un particolare comportamento sulle richieste e di fornire ulteriori informazioni nelle risposte.

Incluso nelle intestazioni HTTP è la possibilità di richiedere che le informazioni vengano restituite in una lingua nazionale. Il REST API rispetta questa intestazione dove possibile.

Specifica di una lingua nazionale

Nell'intestazione HTTP ACCEPT - LANGUAGE, è possibile fornire una o più tag di lingua. È possibile, facoltativamente, associare una classificazione ai tag, consentendo la specifica di un elenco ordinato per preferenza. Questa pagina contiene un'utile discussione sul principio.

REST API rispetta questa intestazione, selezionando una lingua dall'intestazione ACCEPT - LANGUAGE e restituendo i messaggi in tale lingua. Quando l'intestazione ACCEPT - LANGUAGE non contiene alcuna lingua supportata da REST API , i messaggi vengono restituiti in una lingua predefinita. Questa lingua predefinita corrisponde alla locale predefinita del server Web REST API .

La sezione “Quali dati vengono tradotti?” a pagina 89 spiega quali dati vengono tradotti.

Indicazione della lingua applicabile nelle risposte

L'intestazione HTTP CONTENT - LANGUAGE nelle risposte da REST API indica la lingua in cui vengono restituiti i messaggi.

Quali dati vengono tradotti?

I messaggi di errore e informativi sono tradotti, mentre l'altro testo non lo è.

- I dati restituiti da un gestore code non vengono tradotti, ad esempio nel caso di esecuzione di un comando MQSC tramite REST API, le risposte del gestore code sono nella locale del gestore code.
- La documentazione generata (Swagger) per REST API, come esposta tramite la funzione `apiDiscovery` , è in inglese.

Quali lingue sono supportate?

Oltre all'inglese, i messaggi di errore e informativi di REST API sono tradotti nelle seguenti lingue.

Cinese(Semplificato)
Contrassegnato dalla tag della lingua `zh_CN`

Cinese (tradizionale)
Contrassegnato dalla tag della lingua `zh_TW`

Ceco
Contrassegnato dalla tag della lingua `cs`

Francese
Contrassegnato dalla tag della lingua `fr`

Ungherese
Contrassegnato dalla tag della lingua `hu`

Italiano
Contrassegnato dalla tag della lingua `it`

Giapponese
Contrassegnato dalla tag della lingua `ja`

Coreano
Contrassegnato dalla tag della lingua `ko`

Polacco
Contrassegnato dalla tag della lingua `pl`

(Brasiliano) Portoghese
Contrassegnato dalla tag della lingua `pt_BR`

Russo
Contrassegnato dalla tag della lingua `ru`

Spagnolo
Contrassegnato dalla tag della lingua `es`

Esempi

Negli esempi, il server Web ha una locale predefinita in inglese.

Specifica di una singola lingua supportata
Nelle intestazioni della richiesta, `ACCEPT-LANGUAGE` è impostata su `fr`. Questa impostazione specifica che il francese è la lingua preferita per il testo traducibile.

Nelle intestazioni della risposta, `CONTENT-LANGUAGE` è impostata su `fr`. Questa impostazione indica che i messaggi di errore e informativi nella risposta sono in francese.

Specifica di un elenco di lingue
Nelle intestazioni della richiesta, `ACCEPT-LANGUAGE` è impostata su `am, fr`. Questa impostazione specifica che l'amarico e il francese sono lingue accettabili per il testo traducibile e che l'amarico è la lingua preferita per il testo traducibile.

Nelle intestazioni della risposta, `CONTENT-LANGUAGE` è impostata su `fr`. Questa impostazione indica che i messaggi di errore e informativi nella risposta sono in francese, poiché REST API non supporta l'amarico.

Specifica di una singola lingua non supportata

Nelle intestazioni della richiesta, `ACCEPT-LANGUAGE` è impostata su `am`. Questa impostazione specifica che l'amarico è la lingua preferita per il testo traducibile.

Nelle intestazioni della risposta, `CONTENT-LANGUAGE` è impostata su `en`. Questa impostazione indica che i messaggi di errore e informativi nella risposta sono in inglese, poiché REST API non supporta l'amarico.

REST API versioni

Il numero di versione REST API fa parte dell'URL di base per le richieste REST. Ad esempio, `https://localhost:9443/ibmmq/rest/`**`v2`**`/admin/installation`. Il numero di versione viene utilizzato per isolare i client dalle modifiche al REST API che potrebbero essere introdotte nelle release future.

IBM MQ 9.2.0 introduce la versione 2 di REST API. Questo aumento di versione si applica a administrative REST API, messaging REST APIe MFT REST API. Questo aumento di versione modifica l'URL della risorsa utilizzato per REST API. Il prefisso URL per gli URL della risorsa alla Versione 2 è il seguente URL:

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

Stabilized Alcune modifiche introdotte in REST API potrebbero modificare la funzione REST API esistente in modo che i client che utilizzano REST API potrebbero dover essere aggiornati. Per evitare che tali modifiche forzino l'aggiornamento dei client, il numero di versione REST API viene aumentato e la funzione esistente viene stabilizzata al numero precedente. La nuova funzione che potrebbe modificare la funzione esistente viene aggiunta a REST API al nuovo numero di versione. Di conseguenza, i client possono continuare ad utilizzare REST API alla versione precedente senza essere aggiornati.

Le modifiche REST API che potrebbero comportare la richiesta di un aggiornamento del client includono le seguenti modifiche:

- Rimozione del supporto per un attributo esistente nel JSON inviato o restituito da REST API.
- Rimozione di un URL, un verbo HTTP o un'intestazione. Ad esempio, se viene rinominato un URL o un'intestazione o se viene utilizzato un verbo diverso.
- Aggiunta di un nuovo attributo JSON obbligatorio ai dati inviati a un URL esistente.
- Aggiunta di una nuova intestazione HTTP obbligatoria ai dati inviati a un URL esistente.
- Aggiunta di un nuovo parametro di query obbligatorio a un URL esistente.

Quando questo tipo di modifica viene introdotto nella funzione REST API che esisteva in una release Long Term Support (LTS), il numero di versione di REST API viene aumentato per la prima di queste modifiche. Eventuali modifiche successive apportate in una release di Continuous Delivery (CD) che potrebbero richiedere modifiche ai client che utilizzano REST API utilizzano il nuovo numero di versione.

Questo numero di versione rimane lo stesso nelle release successive di CD fino alla release successiva di LTS . Pertanto, il numero di versione aumenta al massimo una volta tra le release LTS .

Stabilized Quando il numero di versione viene aumentato, la funzione REST API esistente viene stabilizzata al numero di versione precedente. Vale a dire, la funzione REST API esistente disponibile nella release LTS rimane disponibile al numero di versione precedente, ma non vengono apportate ulteriori modifiche a tale versione. Qualsiasi nuova funzione aggiunta a REST API viene aggiunta alla nuova versione di REST API . Tuttavia, eventuali aggiunte apportate a REST API nelle release CD prima dell'aumento della versione non sono garantite per essere incluse nella versione precedente di REST API.

Deprecated I client esistenti possono continuare ad utilizzare REST API al numero di versione precedente senza richiedere alcuna modifica. Le versioni precedenti di REST API potrebbero essere obsolete ed eventualmente rimosse.

Alcune modifiche non richiedono modifiche ai client che utilizzano REST API. Queste modifiche non comportano un aumento del numero di versione. Pertanto, assicurarsi che tutti i client che utilizzano REST API non debbano essere aggiornati quando vengono introdotti questi tipi di modifiche. Queste modifiche al file REST API potrebbero includere le seguenti modifiche:

- Aggiunta di un nuovo attributo JSON ai dati esistenti restituiti da REST API.
- Aggiunta di un nuovo URL.
- Aggiunta di un nuovo verbo HTTP a un URL esistente.
- Aggiunta di un nuovo codice di stato a un URL esistente.
- Aggiunta di nuovi attributi JSON facoltativi ai dati inviati a un URL esistente.
- Aggiunta di nuovi parametri di query su un URL esistente.
- Aggiunta di nuove intestazioni ai dati inviati a un URL esistente.
- Ritorno di nuove intestazioni da REST API.

Modifiche alla nuova funzione API REST Continuous Delivery

Per la nuova funzione REST API aggiunta in una release CD , tutte le modifiche apportate a questa nuova funzione che potrebbero richiedere modifiche ai client REST API non aumentano il numero di versione. In altre parole, la nuova funzione può essere modificata prima della release successiva di LTS senza aumentare il numero di versione. Quando la funzione è inclusa in una release LTS , eventuali modifiche successive che potrebbero richiedere modifiche ai client REST API aumentano il numero di versione.

Esempio

1. In LTS release X, REST API è alla versione 1.
2. In CD release X.0.1, viene aggiunto il supporto per un nuovo URL. Questa modifica non richiede modifiche ai client che utilizzano REST API. Pertanto, REST API rimane alla versione 1.
3. In CD X.0.2, viene aggiunto il supporto per un nuovo URL. Questa modifica non richiede modifiche ai client che utilizzano l'API REST. Pertanto, REST API rimane alla versione 1.
4. Alla release LTS Y, REST API è alla versione 1.
5. In CD release Y.0.1, viene ridenominato un URL esistente. Questa modifica potrebbe richiedere modifiche ai client che utilizzano REST API. Pertanto, una nuova versione di REST API viene creata come versione 2. L'URL ridenominato è incluso nella versione 2 di REST API, insieme a tutta la funzione esistente. Tutte le nuove funzioni aggiunte a REST API vengono aggiunte alla versione 2. La versione 1 rimane stabilizzata al livello di LTS release Y.
6. In CD release Y.0.2, un altro URL esistente viene ridenominato. Poiché la versione è già aumentata in CD release Y, REST API rimane alla versione 2. La versione 1 rimane stabilizzata al livello di LTS release Y.
7. Nella release LTS Z, REST API rimane alla versione 2. La versione 1 rimane stabilizzata al livello di LTS release Y.

Amministrazione mediante IBM MQ Console

È possibile eseguire le attività di gestione di base utilizzando IBM MQ Console.

Nota: Non disabilitare il server dei comandi su nessuno dei gestori code quando si utilizza IBM MQ Console. Se il server dei comandi è disabilitato per un gestore code:

- Il IBM MQ Console non risponde, con lunghi ritardi nell'elaborazione dei comandi
- I comandi emessi per il gestore code vanno in timeout.

Attività correlate

Traccia di IBM MQ Console

Introduzione a IBM MQ Console

Configurare il server mqweb; stabilire l'URI per IBM MQ Console; collegarsi alla console; accedere alla console.

Prima di iniziare

Per completare questa attività, è necessario essere un utente con particolari privilegi in modo da poter utilizzare il comando **dspmqweb**:

- z/OS Su z/OS, è necessario disporre dell'autorità per eseguire il comando **dspmqweb** e l'accesso in scrittura al file `mqwebuser.xml`.
- Multi Su tutti gli altri sistemi operativi, è necessario essere un utente privilegiato.

IBM i Su IBM i, i comandi devono essere in esecuzione in QSHELL.

Informazioni su questa attività

Tenere presente le seguenti restrizioni:

- z/OS
  - I gestori code su z/OS non possono essere creati, eliminati, avviati o arrestati.
  - Gli iniziatori di canali su z/OS non possono essere avviati o arrestati e lo stato dell'iniziatore di canali non viene visualizzato.
  - I listener non possono essere visualizzati o gestiti.
  - I comandi start, ping, resolve e reset channel possono essere emessi solo con CHLDISP (DEFAULT).
  - Gli oggetti definiti con QSGDISP (GROUP) non possono essere visualizzati o gestiti.
  - Impossibile gestire la sicurezza del gestore code.
  - L'utilizzo delle risorse di sistema non può essere monitorato.
- Multi
  - Non è possibile utilizzare IBM MQ Console per utilizzare i canali AMQP.
  - Non è possibile utilizzare IBM MQ Console per utilizzare i canali MQTT .

Procedura

1. Se il server mqweb non è ancora configurato per l'utilizzo da parte di IBM MQ Console, configurare il server mqweb.

   Per ulteriori informazioni sulla configurazione del server mqweb con un registro di base, consultare Configurazione di base per il server mqweb.
2. z/OS

   Su z/OS, impostare la variabile di ambiente WLP_USER_DIR in modo da poter utilizzare il comando **dspmqweb** . Impostare la variabile in modo che punti alla configurazione del server mqweb immettendo il seguente comando:

   ```
   export WLP_USER_DIR=WLP_user_directory
   ```

   dove *WLP_user_directory* è il nome della directory passata a `crtmqweb`. Ad esempio:

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

   Per ulteriori informazioni, consultare Creazione del server mqweb.
3. Determinare l'URI per IBM MQ Console immettendo il seguente comando:

   ```
   dspmqweb status
   ```

   Il comando genera un output simile al seguente:

   ```
   MQWB1124I: Server 'mqweb' is running.
   ```

```
URLS:
https://localhost:9443/ibmmq/rest/v1/
https://localhost:9443/ibmmq/console/
```

L'URI per IBM MQ Console termina con il suffisso `console/`.

4. Connettersi a IBM MQ Console immettendo l'URL dal passo precedente in un browser.

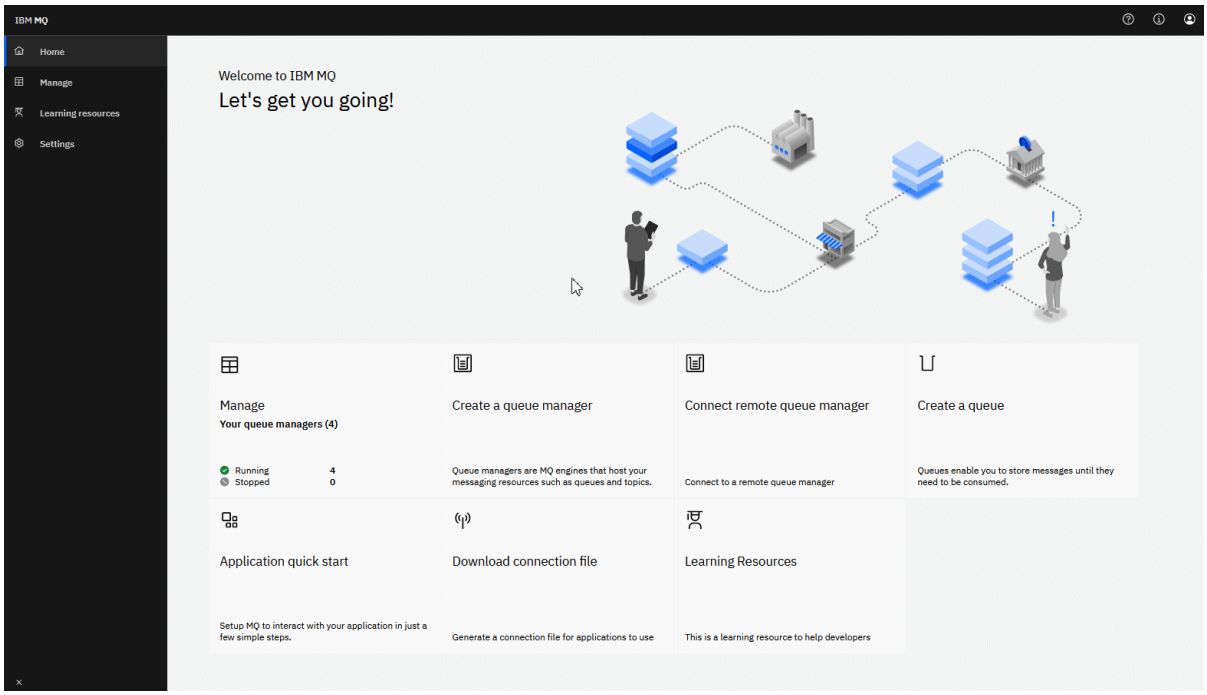
   Un'eccezione di sicurezza potrebbe essere prodotta dal browser perché il certificato predefinito fornito con il server mqweb non è un certificato attendibile. Scegliere di procedere con IBM MQ Console.
5. Accedi a IBM MQ Console. Utilizzare il nome utente `mqadmin` la password `mqadmin`.

Operazioni successive

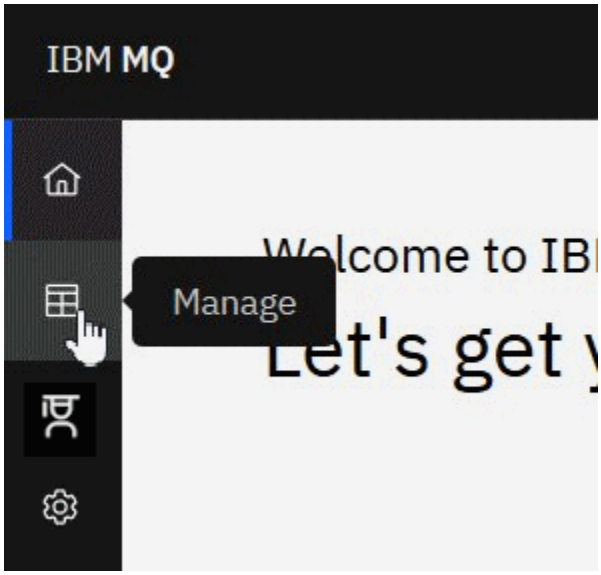
Per impostazione predefinita, IBM MQ Console utilizza l'autenticazione basata su token per autenticare gli utenti. È anche possibile utilizzare l'autenticazione del certificato client. Per ulteriori informazioni, consultare Utilizzo dell'autenticazione del certificato client con REST API e IBM MQ Console.

V 9.4.0 Breve panoramica di IBM MQ Console

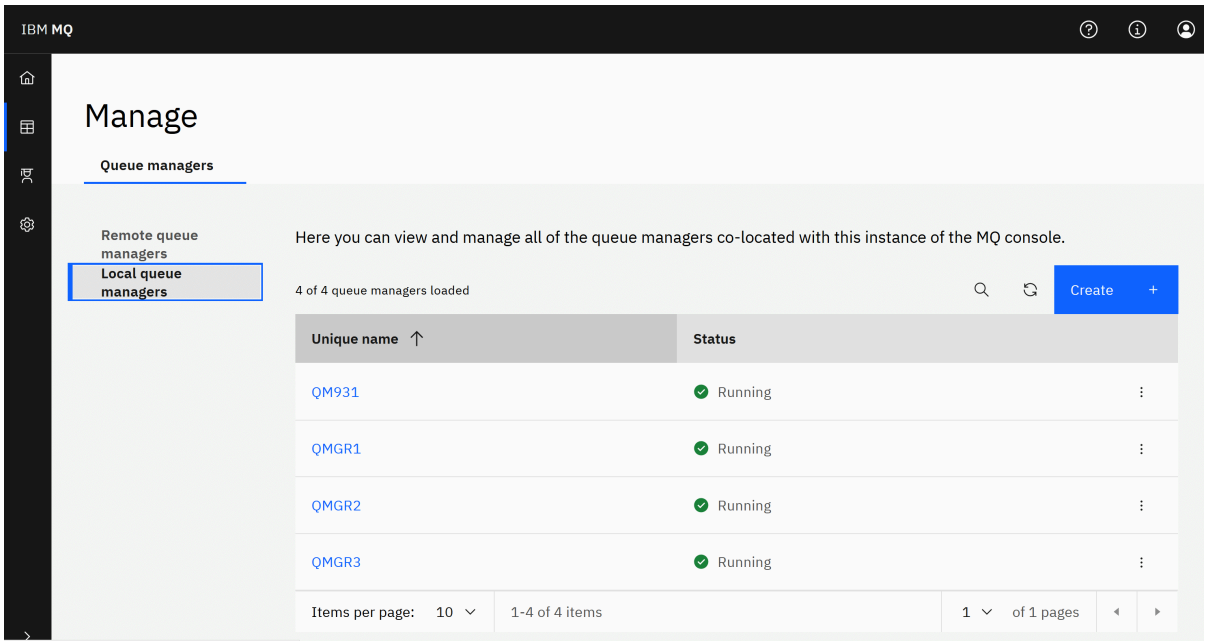
Quando si accede per la prima volta a IBM MQ Console , si viene portati alla pagina di destinazione. Da qui è possibile scegliere di gestire i gestori code esistenti, creare un gestore code o una coda, passare ad alcuni argomenti di formazione o aprire le informazioni sul prodotto IBM MQ in IBM Documentation. È inoltre possibile avviare l'avvio rapido dell'applicazione, che guida l'utente attraverso il processo di configurazione rapida e semplice della messaggistica tra gestori code e applicazioni nuovi o esistenti.



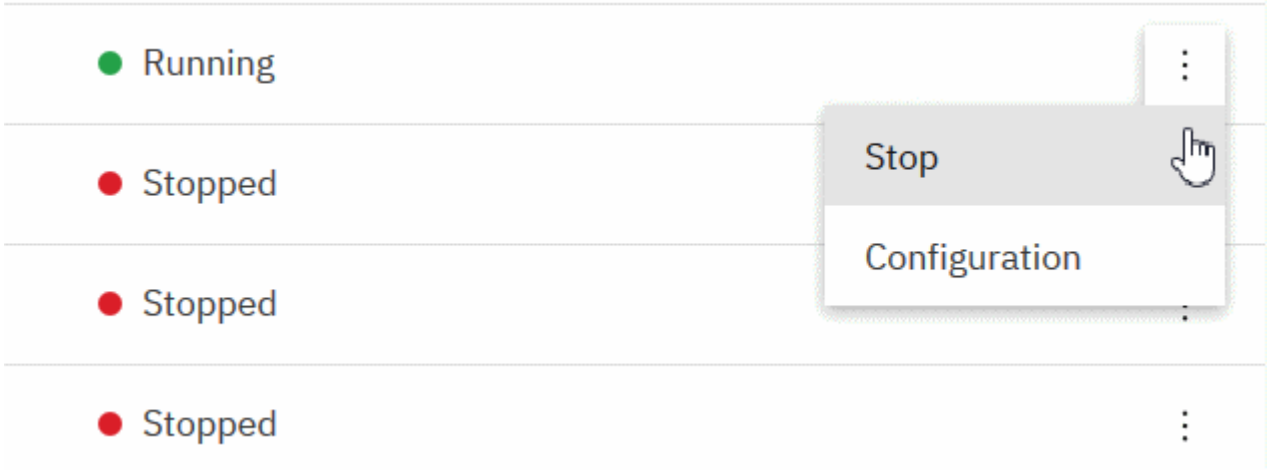
In alternativa, è possibile semplicemente fare clic sull'icona Gestisci per avviare la gestione degli oggetti IBM MQ .



La vista Gestisci mostra inizialmente i gestori code e il relativo stato corrente. È anche possibile creare nuovi gestori code e connettersi ai gestori code remoti.

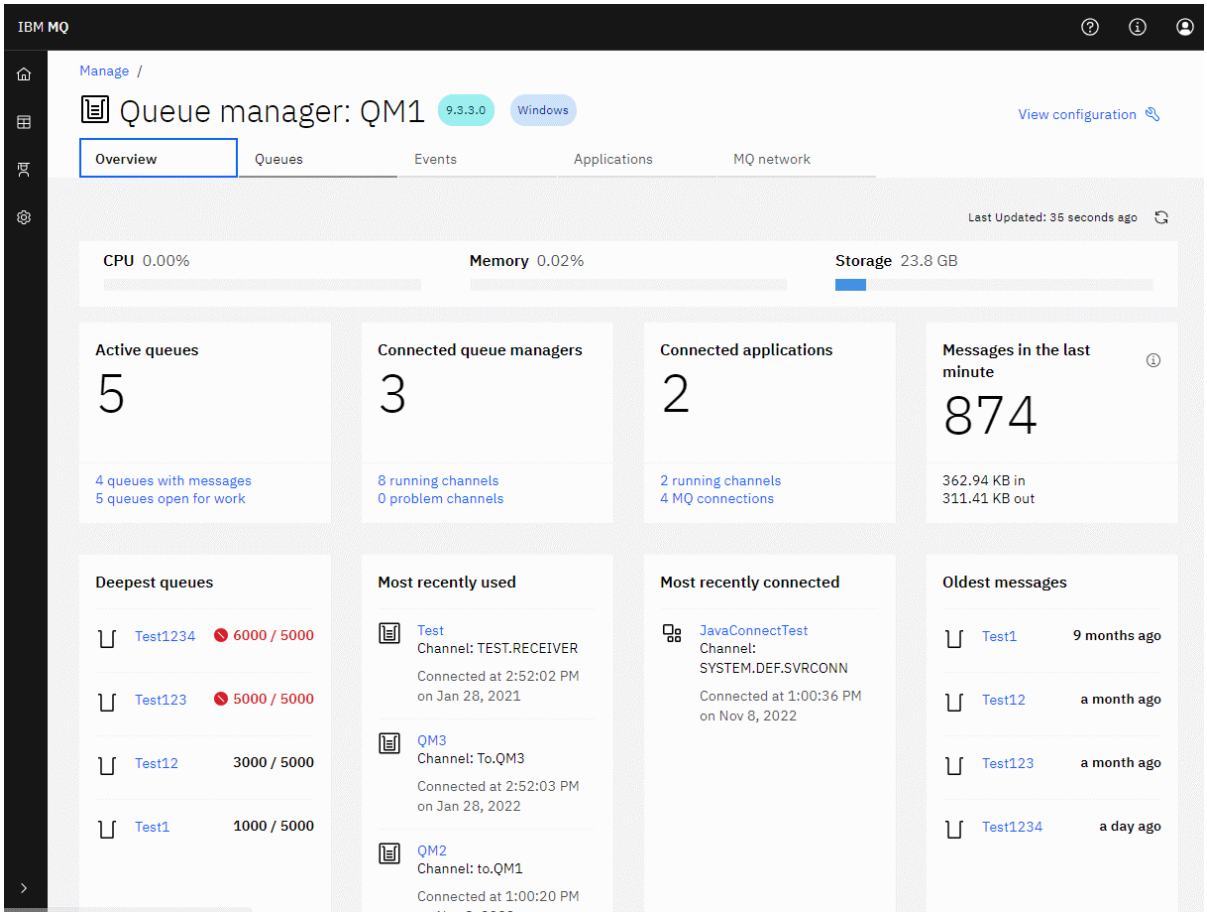


Ogni gestore code dispone di un menu che consente di arrestare o configurare un gestore code in esecuzione, oppure di avviare o eliminare un gestore code arrestato.



I record di autorizzazione, gli oggetti Informazioni di autenticazione e i record di autenticazione di canale per il gestore code si trovano nella scheda **Sicurezza** della pagina **Configurazione** del gestore code, in cui è possibile crearne e aggiungerne di nuovi.

Fare clic sul nome di un gestore code in esecuzione per aprirne il dashboard.



Dal dashboard del gestore code è possibile completare le seguenti azioni:

V 9.4.0 Sulla scheda **Panoramica** , visualizzare le informazioni riportate di seguito:

CPU

Stima in percentuale dell'utilizzo di CPU da parte del gestore code. (Non applicabile su z/OS.)

Memoria
Stima percentuale di utilizzo della memoria da parte del gestore code. (Non applicabile su z/OS o Windows.)

Memoria
Stima percentuale dello spazio libero del disco su cui risiede il gestore code. (Non applicabile su z/OS.)

Code attive
Conteggio delle code che hanno messaggi o che sono aperte per l'input o l'output.

Gestori code connessi
Conteggio dei gestori code attualmente connessi come derivati dai canali attivi.

Applicazioni connesse
Conteggio delle applicazioni attualmente collegate.

Messaggi nell'ultimo minuto
Visualizza un riepilogo degli argomenti di sistema PUT/GET che mostrano la velocità di trasmissione dei messaggi ogni 10 secondi. (Non applicabile su z/OS.)

Sottoscrizioni
Visualizza un numero di sottoscrizioni. Visibile solo su z/OS e su altre piattaforme in cui il monitoraggio degli argomenti di sistema è inibito (vedere proprietà setmqweb).

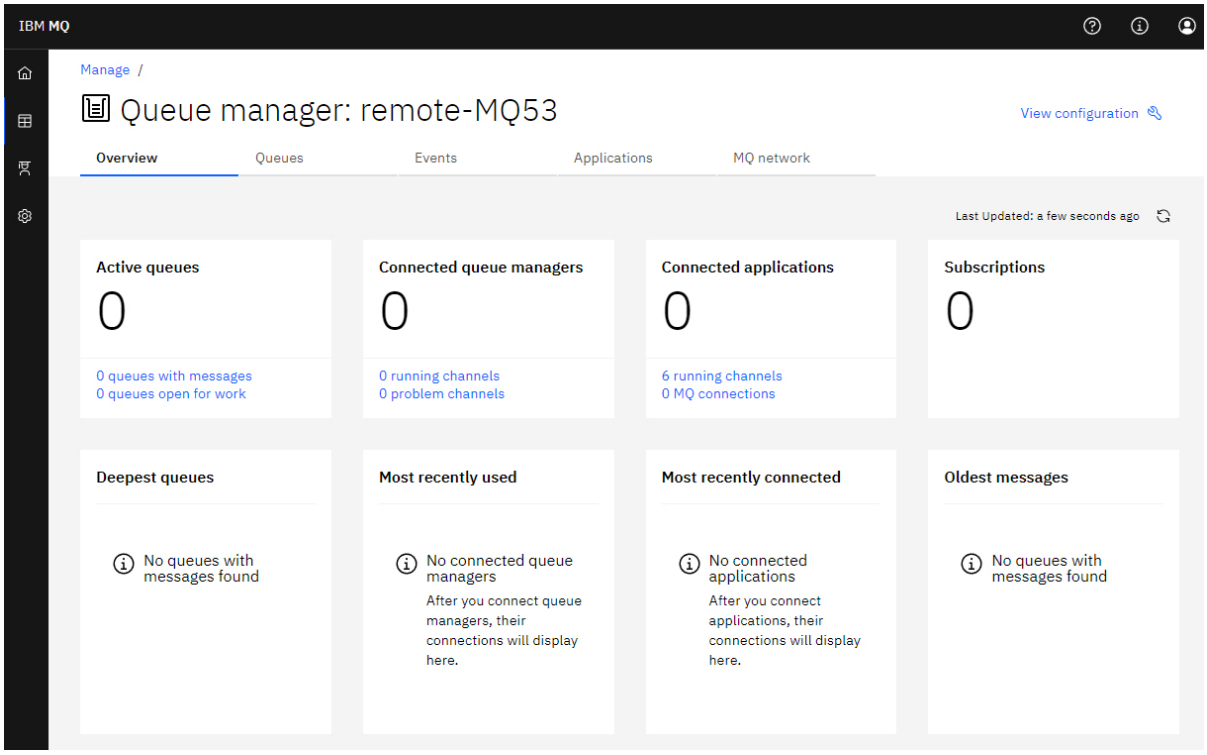
Code della massima lunghezza
Elenca le code in ordine di profondità. Mostra la profondità della coda corrente e la profondità massima della coda.

Utilizzato più di recente
Elenca i gestori code attualmente connessi, ordinati per data dell'ultimo messaggio.

Connesso più di recente
Elenca le applicazioni attualmente connesse come derivate da canali di connessione server attivi, ordinati per data e ora di avvio del canale.

Messaggi meno recenti
Elenca le code ordinate per data e ora del messaggio meno recente.

Le informazioni visualizzate nella scheda **Panoramica** derivano dal monitoraggio degli argomenti di sistema (consultare Metriche pubblicate negli argomenti di sistema). z/OS non supporta il monitoraggio degli argomenti di sistema e il monitoraggio per scopi di visualizzazione della console può essere disabilitato su altre piattaforme (consultare proprietà setmqweb). In questi casi, la scheda **Panoramica** visualizza informazioni più limitate e il suo aspetto è simile al seguente esempio:



Nella scheda **Code** :

- Crea nuove code
- Fare clic sul nome di una coda per visualizzare i messaggi esistenti e crearne di nuovi e per configurare la coda.

Nella scheda **Eventi** :

Argomenti

- Crea nuovi argomenti
- Configura argomenti esistenti
- Fare clic su un nome argomento per visualizzare le sottoscrizioni corrispondenti

Sottoscrizioni

- Crea nuove sottoscrizioni gestite o non gestite
- Configura sottoscrizioni esistenti

V 9.4.0 Nella scheda **Applicazioni** :

Panoramica

Contiene riquadri che forniscono panoramiche delle seguenti statistiche:

Applicazioni connesse

Visualizza un conteggio del numero di applicazioni connesse. Fornisce collegamenti alle schede seguenti:

- **Istanze dell'applicazione**
- **Collegamenti**

Istanze del canale in esecuzione

Visualizza un conteggio del numero di istanze del canale SVRCONN e da tale link a quelle definite o arrestate nella scheda **Canale app** .

Connessioni
Visualizza un conteggio del numero di connessioni. Fornisce collegamenti alle seguenti informazioni nella scheda **Connessioni** :

- Connessioni locali (quelle senza un nome canale)
- Connessioni remote (quelle con un nome canale)

Applicazioni più comuni
Visualizza un elenco di applicazioni frequenti, ordinate in base al numero di connessioni utilizzate.

Canali più comuni
Visualizza un elenco di canali frequenti, ordinati per il numero di istanze attive.

Transazioni meno recenti
Visualizza un elenco di transazioni meno recenti per nome applicazione. Queste transazioni hanno connessioni con unità di lavoro aperte e sono ordinate per data e ora di avvio UOW.

Versioni remote connesse
Visualizza un elenco di versioni IBM MQ comuni connesse, ovvero, le istanze del canale che hanno una REMOTE_VERSION specificata.

Sicurezza del canale dell'applicazione
Visualizza un elenco di protocolli di sicurezza del canale connessi comuni, ovvero, le istanze del canale che dispongono di un SECURITY_PROTOCOL specificato.

Velocità di trasferimento canale
Visualizza un elenco di canali comuni ordinati per velocità di trasferimento di messaggi e byte. Utilizza la data e ora di avvio del canale per calcolare la durata e utilizza MSGS e MQIACH_BYTES_SENT/ MQIACH_BYTES_RCVD per calcolare la frequenza.

Applicazioni
Visualizzare le informazioni sulle applicazioni connesse al gestore code.

Canali
Visualizzare l'attività sui canali connessi alle applicazioni.

Canali app

- Avvio, arresto, ping e configurazione dei canali
- Creare nuovi canali
- Reimposta canali

Istanze di canale dell'app

- Visualizza lo stato delle istanze del canale dell'applicazione
- Risolvi messaggi dubbi sui canali

V 9.4.0 Nella scheda **MQ Network** :

Panoramica
Contiene riquadri che forniscono panoramiche delle seguenti statistiche:

Esecuzione delle istanze del canale del gestore code
Visualizza un numero di istanze del canale non SVRCONN. Visualizza i collegamenti ai tipi di istanze del canale riportati di seguito sulla scheda **Gestori code connessi** :

- Canali definiti
- Canali arrestati

Gestori code connessi
Visualizza un conteggio dei gestori code connessi da MQCA_REMOTE_Q_MGR_NAME. Fornisce inoltre un conteggio dei gestori code restituiti da MQCMD_INQUIRE_CLUSTER_Q_MGR.

Appartenenza cluster
Se è presente un solo cluster di gestori code, visualizza il nome del cluster e se il gestore code è un repository completo o parziale. Visualizza quanti gestori code sono visibili nel cluster. Se è presente più di un cluster, visualizza il numero di cluster più un conteggio dei gestori code del repository completo e parziale in ciascun cluster.

Canali del gestore code in errore
Visualizza un elenco di canali in stato di nuovo tentativo (non arrestati/in esecuzione). Calcola il numero di tentativi rimanenti se si trova nello stato Nuovo tentativo. L'elenco contiene i canali con i seguenti tipi di stato:

- MQCHS_PAUSED
- MQCHS_RETRYING

Ritardi dei messaggi più lunghi
Visualizza un elenco di canali che hanno un indicatore di tempo XMIT (lungo periodo).

Code di trasmissione non presidiate
Visualizza un elenco di code di trasmissione con profondità di coda diversa da zero e senza handle associati.

Versioni di connessione remota
Visualizza un elenco di versioni IBM MQ comuni connesse, ovvero, le istanze del canale che hanno una REMOTE_VERSION specificata.

Sicurezza del canale del gestore code
Visualizza un elenco di protocolli di sicurezza del canale connessi comuni, ovvero, le istanze del canale che dispongono di un SECURITY_PROTOCOL specificato.

Integrità del cluster

Visualizza un numero di statistiche indipendenti relative allo stato del cluster. Lo stato include:

- Il numero di oggetti cluster (code, argomenti, gestori code).
- Il numero di gestori code sospesi (MQIACF_SUSPEND impostato su YES).
- La profondità del SISTEMA SYSTEM.CLUSTER.COMMAND.QUEUE .
- Il numero di voci del gestore code del cluster che iniziano con SYSTEM.TEMP.

Se tutti questi valori sono zero, questo riquadro non viene visualizzato e viene visualizzato il riquadro **Listener** .

Listener
Visualizza un elenco di listener e se si trovano in uno stato di esecuzione. Visualizzato solo se il riquadro **Integrità cluster** non viene visualizzato.

Gestori di code connessi
Visualizzare i dettagli dei gestori code attualmente connessi al gestore code.

Canali gestore code

- Avvio, arresto, ping e configurazione dei canali
- Creare nuovi canali
- Reimposta canali

Istanze di canale del gestore code

- Visualizza lo stato delle istanze del canale gestore code
- Risolvi messaggi dubbi sui canali

V 9.4.0 IBM MQ Console: operazioni con i gestori code locali

Creare, configurare e controllare i gestori code locali dal livello superiore della vista Gestisci

.

Informazioni su questa attività

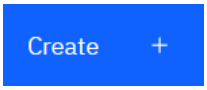
Multi La vista Gestisci elenca i gestori code locali aggiunti all'installazione di IBM MQ da cui è in esecuzione IBM MQ Console . I gestori code associati a diverse installazioni di IBM MQ nello stesso sistema non sono elencati.

z/OS Su z/OS, la vista Gestisci elenca i gestori code che sono alla stessa versione di IBM MQ Consolee che sono definiti sul sistema su cui è in esecuzione IBM MQ Console . I gestori code con una versione diversa da IBM MQ Console non sono elencati.

È possibile selezionare singoli gestori code dall'elenco da utilizzare.

Nota: Il IBM MQ Console può connettersi a un gestore code RDQM locale quando è attivo (vale a dire, ha il ruolo primario), ma non offre alcuna funzione specifica di RDQM.

Procedura

- Per creare un nuovo gestore code locale:
  a) Fare clic sul pulsante Crea 

nella vista di elenco del gestore code.
  b) Immettere un nome per il nuovo gestore code. Il nome può contenere un massimo di 48 caratteri. I caratteri validi sono lettere e numeri e i caratteri ".", "/", "_"e "%" .
  c) Opzionale: Immettere una porta TCP/IP disponibile per il gestore code su cui ascoltare. Il numero di porta non deve essere superiore a 65535.
  d) Fai clic su **Crea**. Il nuovo gestore code viene creato e avviato.
- Per avviare un gestore code locale:
  a) Individuare il gestore code che si desidera avviare nell'elenco.
  b) Selezionare **Avvia** dal menu ⋮ .
- Per arrestare un gestore code locale:
  a) Selezionare il gestore code che si desidera arrestare dall'elenco nel widget del gestore code locale.
  b) Selezionare **Arresta** dal menu ⋮ .
- Per cancellare un gestore code locale:
  a) Se il gestore code è in esecuzione, arrestarlo.
  b) Selezionare **Visualizza configurazione** dal menu ⋮ e selezionare **Elimina gestore code**.
  c) Confermare che si desidera eliminare il gestore code immettendone il nome nella finestra di conferma. Il gestore code e tutti gli oggetti associati vengono eliminati.
- Per visualizzare e modificare le proprietà di un gestore code locale:
  a) Verificare che il gestore code sia in esecuzione e individuarlo nell'elenco di gestori code.
  b) Selezionare **Visualizza configurazione** dal menu ⋮ .

c) Assicurarsi che la scheda **Proprietà** sia selezionata. Visualizzare le proprietà e modificarle come richiesto. Se la casella di testo della proprietà è disabilitata, la proprietà è di sola lettura o può essere modificata solo dalla riga comandi. Per informazioni su una proprietà, è possibile visualizzare le informazioni relative alla proprietà in Proprietà del gestore code.

- Per gestire le impostazioni di protezione per il gestore code locale:
  a) Verificare che il gestore code sia in esecuzione e selezionarlo dall'elenco dei gestori code.
  b) Selezionare **Visualizza configurazione** dal menu .
  c) Assicurarsi che la scheda **Sicurezza** sia selezionata.
  d) È possibile gestire oggetti di autenticazione, record di autorizzazione o oggetti di autenticazione di canale. Per ulteriori informazioni, consultare i seguenti argomenti:
     – “IBM MQ Console: utilizzo degli oggetti delle informazioni di autenticazione” a pagina 101
     – “IBM MQ Console: utilizzo dei record di autorizzazione del gestore code” a pagina 102
     – “IBM MQ Console: operazioni con i record di autenticazione di canale” a pagina 103

V 9.4.0 *IBM MQ Console: utilizzo degli oggetti delle informazioni di autenticazione*

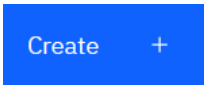
È possibile utilizzare la console per aggiungere ed eliminare gli oggetti delle informazioni di autenticazione su un gestore code. È inoltre possibile visualizzare e impostare le proprietà e gestire i record di autorizzazione per gli oggetti.

Informazioni su questa attività

La vista delle informazioni di autenticazione elenca le informazioni di autenticazione esistenti per un determinato gestore code. È possibile selezionare singole informazioni di autenticazione dall'elenco da gestire.

Le informazioni di autenticazione del gestore code fanno parte del supporto IBM MQ per TLS (Transport Layer Security). Questi oggetti contengono le definizioni richieste per eseguire il controllo della revoca dei certificati utilizzando OCSP o CRL (Certificate Revocation Lists) sui server LDAP e le definizioni richieste per abilitare il controllo dell'ID utente e della password.

Procedura

- Per visualizzare le informazioni di autenticazione per un gestore code:
  a) Verificare che il gestore code sia in esecuzione e selezionarlo dall'elenco dei gestori code.
  b) Selezionare **Visualizza configurazione** dal menu .
  c) Assicurarsi che la scheda **Sicurezza** sia selezionata.
  d) Selezionare **Informazioni di autenticazione** dal pannello di navigazione.
- Per aggiungere un oggetto delle informazioni di autenticazione:
  a) Fare clic sul pulsante Crea 

nella vista elenco delle informazioni di autenticazione.
  b) Specificare il nome dell'oggetto relativo alle informazioni di autenticazione. I caratteri validi sono lettere e numeri e i caratteri ".", "/", "_"e "%" .
  c) Specificare il tipo di oggetto delle informazioni di autenticazione.
  d) Specificare ulteriori informazioni appropriate per il tipo di oggetto:
     – Per **CRL LDAP**, specificare il **nome server LDAP**. Questo nome è il nome host, l'indirizzo decimale con punti IPv4 o la notazione esadecimale IPv6 dell'host su cui è in esecuzione il server LDAP, con un numero di porta facoltativo. Facoltativamente, è possibile specificare un nome utente e una password per l'utente che accede al server LDAP.

- Per **OCSP**, specificare **URL responder OCSP**. Questo URL è l'URL del responder utilizzato per controllare la revoca del certificato. Questo valore deve essere un URL HTTP contenente il nome host e il numero di porta del responder OCSP. Se il responder OCSP utilizza la porta 80, che è il valore predefinito per HTTP, è possibile omettere il numero di porta. Gli URL HTTP sono definiti in RFC 1738.
- Per **IDPW OS**, non ci sono ulteriori requisiti, anche se è possibile specificare ulteriori opzioni per questo tipo di autenticazione.
- Per **IDPW LDAP**, specificare il **Nome server LDAP** e il nome **Utente breve** . Il nome del server LDAP è il nome host, l'indirizzo decimale puntato IPv4 o la notazione esadecimale IPv6 dell'host su cui è in esecuzione il server LDAP, con un numero di porta facoltativo. Il nome utente breve è il campo nel record utente LDAP utilizzato come nome breve per la connessione. Facoltativamente, è possibile specificare ulteriori opzioni per questo tipo di autenticazione.

e) Fare clic su **Aggiungi**.

- Per eliminare un oggetto delle informazioni di autenticazione:

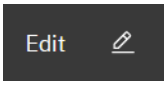
  a) Selezionare l'icona della spanner per l'oggetto delle informazioni di autenticazione che si desidera eliminare dall'elenco.

  b) Nella vista delle proprietà dell'oggetto, fare clic su **Elimina oggetto informazioni di autenticazione**.

  c) Confermare che si desidera eliminare l'oggetto delle informazioni di autenticazione facendo clic su **Elimina**. L'oggetto viene eliminato.

- Per visualizzare e modificare le proprietà di un oggetto delle informazioni di autenticazione:

  a) Selezionare l'icona della chiave per l'oggetto delle informazioni di autenticazione che si desidera visualizzare dall'elenco.

  b) Per modificare le proprietà visualizzate, fare clic sul pulsante Modifica 

  c) Modificare le proprietà come desiderato. Se la casella di testo della proprietà è disabilitata, la proprietà è di sola lettura o può essere modificata solo dalla riga comandi.

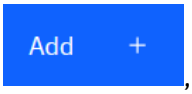
  d) Fare clic su **Salva** per salvare le modifiche.

- Per visualizzare e modificare i record di autorizzazione per un oggetto delle informazioni di autenticazione:

  a) Selezionare l'icona della chiave per l'oggetto delle informazioni di autenticazione per cui si desidera visualizzare il record di autorità dall'elenco.

  b) Selezionare la scheda **Sicurezza** .

  c) Per modificare o eliminare un record di autorizzazione esistente, selezionare **Modifica** o **Elimina** dal menu .

  d) Per aggiungere un nuovo record di autorizzazione, fare clic sul pulsante **Aggiungi** 

, fornire i dettagli del nuovo record di autorizzazione e fare clic su **Crea**.

V 9.4.0 *IBM MQ Console: utilizzo dei record di autorizzazione del gestore code*

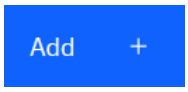
È possibile controllare l'accesso di utenti e gruppi ai gestori code specificando un record di autorizzazione per tale utente o gruppo.

Informazioni su questa attività

È possibile ottimizzare l'accesso di un utente di messaggistica o di un gruppo di utenti di messaggistica a un determinato gestore code utilizzando i record di autorizzazione. Esistono due tipi di record di autorizzazioni: i record **accesso gestore code** che controllano le autorizzazioni generali e l' **autorizzazione a creare** i record che controllano quali utenti e gruppi possono creare gli oggetti per il gestore code.

Procedura

- Per visualizzare i record di autorizzazioni per un gestore code:
  a) Verificare che il gestore code sia in esecuzione e selezionarlo dall'elenco dei gestori code.
  b) Selezionare **Visualizza configurazione** dal menu ⋮.
  c) Assicurarsi che la scheda **Sicurezza** sia selezionata.
  d) Selezionare **Record di autorizzazioni** dal pannello di navigazione. La vista mostra i record di autorizzazioni in due riquadri, consentendo di gestire i record di autorizzazioni generali e di creare record di autorizzazioni.
- Per aggiungere un record di autorizzazione generale:
  a) Fare clic sul pulsante Aggiungi 

nella vista elenco **Accesso gestore code** .
  b) Scegliere se si sta aggiungendo un record di autorizzazione per un utente o un gruppo.
  c) Specificare il nome dell'utente o del gruppo per cui si sta aggiungendo un record di autorizzazione (il record di autorizzazione lo prende come nome).
  d) Selezionare le autorizzazioni che si desidera concedere.
  e) Fai clic su **Crea**.
- Per aggiungere un record di autorizzazione di creazione:
  a) Fare clic sul pulsante Aggiungi 

nella vista elenco **Autorizzazione alla creazione** .
  b) Scegliere se si sta aggiungendo un record di autorizzazione per un utente o un gruppo.
  c) Specificare il nome dell'utente o del gruppo per cui si sta aggiungendo un record di autorizzazione (il record di autorizzazione lo prende come nome).
  d) Selezionare i tipi di oggetto che si sta concedendo l'autorizzazione alla creazione.
  e) Fai clic su **Crea**.
- Per eliminare un record di autorizzazioni:
  a) Selezionare il record di autorizzazione che si desidera eliminare e selezionare **Elimina**.
  b) Confermare che si desidera eliminare l'oggetto delle informazioni di autenticazione facendo clic su **Elimina**. L'oggetto viene eliminato.
- Per visualizzare e modificare le proprietà di un record di autorizzazione:
  a) Fare clic sul record di autorizzazioni che si desidera visualizzare.
  b) Modificare le impostazioni come richiesto e fare clic su **Salva** per salvare le modifiche.

V 9.4.0 *IBM MQ Console: operazioni con i record di autenticazione di canale*

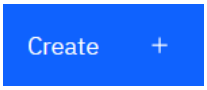
È possibile utilizzare IBM MQ Console per aggiungere ed eliminare i record di autenticazione di canale su un gestore code. È anche possibile visualizzare e impostare proprietà per i record di autenticazione di canale.

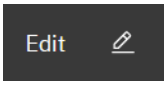
Informazioni su questa attività

Per esercitare un controllo più preciso sull'accesso concesso ai sistemi di connessione a livello di canale, è possibile utilizzare i record di autenticazione di canale.

Per rafforzare la sicurezza, è possibile utilizzare i record di autenticazione del canale di blocco per bloccare l'accesso ai canali. È anche possibile utilizzare i record di autenticazione di canale della mappa di indirizzi per consentire l'accesso agli utenti specificati. Per ulteriori informazioni sui record di autenticazione di canale, consultare Record di autenticazione di canale.

Procedura

- Per visualizzare le informazioni di autenticazione di canale per un gestore code:
  a) Verificare che il gestore code sia in esecuzione e selezionarlo dall'elenco dei gestori code.
  b) Selezionare **Visualizza configurazione** dal menu .
  c) Assicurarsi che la scheda **Sicurezza** sia selezionata.
  d) Selezionare **Autenticazione di canali** dal pannello di navigazione.
- Per aggiungere un record di autenticazione di canale:
  a) Fare clic sul pulsante Crea 

nella vista elenco delle informazioni di autenticazione del canale.
  b) Scegliere il tipo di regola che si desidera utilizzare. Selezionare un **Consenti**, **Blocca**o **Avvisa**.
  c) Scegliere il tipo di identità per cui si sta configurando una regola di autenticazione di canale. Sono disponibili diversi tipi di identità, a seconda del tipo di regola selezionato.
  d) Fornire le informazioni richieste per l'identità che si sta specificando. Per impostazione predefinita, vengono visualizzate le proprietà minime consigliate per cui fornire i valori. È possibile visualizzare tutte le proprietà disponibili selezionando **Creazione personalizzata**.
  e) Fare clic su **Crea** per creare il record di autenticazione di canale.
  Per ulteriori informazioni sulle impostazioni disponibili per i record di autenticazione di canale, consultare Record di autenticazione di canale e SET CHLAUTH
- Per eliminare un record di autenticazione di canale:
  a) Fare clic sull'icona chiave  accanto al record di autenticazione di canale che si desidera eliminare.
  b) Nella vista Modifica autenticazione di canale, fare clic su **Elimina oggetto autenticazione di canale**.
  c) Confermare che si desidera eliminare il record di autenticazione di canale facendo clic su **Elimina**. Il record di autenticazione di canale viene eliminato.
- Per visualizzare e modificare le proprietà di un record di autenticazione di canale:
  a) Fare clic sull'icona della chiave  accanto al record di autenticazione di canale che si desidera modificare o visualizzare. Vengono visualizzate le proprietà.
  b) Fare clic sul pulsante Modifica 

.
  c) Modificare le proprietà come desiderato. Se la casella di testo della proprietà è disabilitata, la proprietà è di sola lettura o può essere modificata solo dalla riga comandi.
  d) Fare clic su **Salva** per salvare le modifiche.

V 9.4.0 Multi

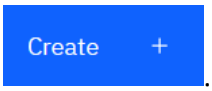
IBM MQ Console: utilizzo dei listener

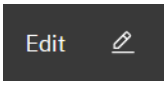
È possibile utilizzare IBM MQ Console per aggiungere ed eliminare i listener, avviare e arrestare i listener, visualizzare e impostare proprietà del listener e gestire i record di autorizzazioni per un listener.

Informazioni su questa attività

La vista dei listener visualizza i listener che esistono per un determinato gestore code. È possibile selezionare singoli listener da utilizzare.

Procedura

- Per visualizzare i listener di un gestore code:
  a) Verificare che il gestore code sia in esecuzione e selezionarlo dall'elenco dei gestori code.
  b) Selezionare **Visualizza configurazione** dal menu ⋮.
  c) Selezionare la scheda **Listener** .
- Per creare un listener:
  a) Fare clic sul pulsante Crea 

.
  b) Fornire le informazioni richieste per il listener che si sta creando.
  c) Fai clic su **Crea**. Il nuovo listener viene creato.
- Per avviare un listener:
  a) Individuare il listener che si desidera avviare nell'elenco.
  b) Selezionare **Avvia** dal menu ⋮.
- Per arrestare un listener:
  a) Individuare il listener che si desidera avviare nell'elenco.
  b) Selezionare **Arresta** dal menu ⋮.
- Per visualizzare e modificare le proprietà di un listener:
  a) Individuare il listener nell'elenco.
  b) Selezionare **Visualizza configurazione** dal menu ⋮.
  c) Assicurarsi che la scheda **Proprietà** sia selezionata. Per modificare le proprietà, fare clic sul pulsante Modifica 

  d) Modificare le proprietà come desiderato. Se la casella di testo della proprietà è disabilitata, la proprietà è di sola lettura o può essere modificata solo dalla riga comandi. Per ulteriori informazioni sulle proprietà, consultare Proprietà del listener nella documentazione di MQ Explorer.
  e) Fare clic su **Salva** per salvare le modifiche.
- Per visualizzare e modificare i record di autorizzazione per un listener:
  a) Individuare il listener nell'elenco.
  b) Selezionare **Visualizza configurazione** dal menu ⋮.
  c) Selezionare la scheda **Sicurezza**.

d) Gestire i record di autorizzazioni come descritto per i record di autorizzazioni del gestore code. Consultare “IBM MQ Console: utilizzo dei record di autorizzazione del gestore code” a pagina 102.

- Per eliminare un listener:
  a) Individuare il listener nell'elenco.
  b) Selezionare **Visualizza configurazione** dal menu .
  c) Fare clic su **Elimina listener**.

V 9.4.0 IBM MQ Console: aggiunta di un gestore code remoto

È possibile utilizzare IBM MQ Console per gestire un gestore code in esecuzione su un sistema remoto.

Prima di iniziare

- È necessario preparare il gestore code sul sistema remoto in modo che possa essere gestito in remoto, fare riferimento al passo “1” a pagina 108, “2” a pagina 108, “3” a pagina 108e “4” a pagina 109 di “Aggiunta di un gestore code remoto a IBM MQ Console utilizzando la riga comandi” a pagina 107.
- È inoltre necessario abilitare le connessioni remote da IBM MQ Console. Per ulteriori informazioni, consultare Configurazione del comportamento della connessione del gestore code remoto.

Informazioni su questa attività

Utilizzare una CCDT (client connection definition table) in formato JSON per specificare i dettagli della connessione remota. Puoi creare una CCDT JSON utilizzando un editor di testo (vedi passo “5” a pagina 109 di “Aggiunta di un gestore code remoto a IBM MQ Console utilizzando la riga comandi” a pagina 107) oppure puoi crearne una utilizzando IBM MQ Console.

In alternativa, è possibile creare la CCDT da IBM MQ Console specificando i dettagli di connessione direttamente quando si aggiunge il gestore code remoto.

È inoltre possibile connettere un gestore code remoto a IBM MQ Console utilizzando la riga comandi per tutte le attività richieste (in aggiunta alla preparazione del gestore code remoto e alla creazione di un CCDT). Consultare “Aggiunta di un gestore code remoto a IBM MQ Console utilizzando la riga comandi” a pagina 107.

Attenzione: se si ricevono i seguenti messaggi:

```
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'.'
```

si sta tentando di passare un `java.lang.Object` al token di account, quando è previsto un tipo di oggetto java `byte[]` .

Procedura

- Per aggiungere un gestore code remoto specificando un CCDT esistente:
  a) Dalla home page, fare clic su **Connetti gestore code remoto**.
  b) Specificare il nome del gestore code remoto.
  c) Facoltativamente, specificare un nome univoco per il gestore code. Se non si specifica un nome univoco, il nome effettivo viene utilizzato con il prefisso "remote-" aggiunto.
  d) Assicurati che sia selezionato **Connect using a JSON CCDT** .
  e) Fare clic su **Sfoglia** e selezionare il file contenente la CCDT JSON che si desidera utilizzare.
  f) Fare clic su **Avanti** per passare alla pagina utente e facoltativamente specificare un nome utente e una password per connettersi al gestore code remoto. Se non si specificano queste informazioni, le informazioni di autenticazione vengono prese dal file di configurazione della connessione remota.

g) Fare clic su **Avanti** per passare alla pagina Certificato. Se CCDT specifica le informazioni "transmissionSecurity", vengono utilizzate queste informazioni. Puoi facoltativamente incollare un certificato (come chiave pubblica codificata base64) e questo viene aggiunto al truststore globale.

Il certificato viene memorizzato temporaneamente in *WLP_USER_DIR*/generated.crts/*uniqueName-qmgrName*.crt prima di essere aggiunto al truststore. Quando la connessione viene aggiunta correttamente, il certificato viene eliminato da questa ubicazione.

h) Fare clic su **Avanti** per visualizzare la pagina di riepilogo. È possibile utilizzare il pulsante **Indietro** per rivisitare le pagine precedenti e apportare correzioni. Se si è soddisfatti delle informazioni, fare clic su **Connetti** per connettersi al gestore code remoto.

- Per aggiungere un gestore code remoto e specificare le informazioni di connessione manualmente:
  a) Dalla home page, fare clic su **Connetti gestore code remoto**.
  b) Specificare il nome del gestore code remoto.
  c) Facoltativamente, specificare un nome univoco per il gestore code. Se non si specifica un nome univoco, il nome effettivo viene utilizzato con il prefisso "remote-" aggiunto.
  d) Selezionare **Immissione manuale**.
  e) Immettere il nome del canale di connessione client che verrà utilizzato dalla connessione.
  f) Specificare il nome dell'host su cui è in esecuzione il gestore code remoto. Se vengono rilevate installazioni remote di MQ , vengono visualizzati i nomi host ed è possibile selezionare l'host del gestore code remoto a cui si desidera connettersi. In alcune configurazioni di rete, non è possibile rilevare istanze MQ remote. In questo caso, aggiungere il nome host e la porta manualmente.
  g) Fare clic su **Avanti** per passare alla pagina utente e facoltativamente specificare un nome utente e una password per connettersi al gestore code remoto. Se non si specificano queste informazioni, le informazioni di autenticazione vengono prese dal file di configurazione della connessione remota.
  h) Fare clic su **Avanti** per passare alla pagina Certificato. È possibile selezionare un CipherSpec SSL dall'elenco a discesa. Puoi facoltativamente incollare un certificato (come chiave pubblica codificata base64) e questo viene aggiunto al truststore globale.

  Il certificato viene memorizzato temporaneamente in *WLP_USER_DIR*/generated.crts/*uniqueName-qmgrName*.crt prima di essere aggiunto al truststore. Quando la connessione viene aggiunta correttamente, il certificato viene eliminato da questa ubicazione.

  i) Fare clic su **Avanti** per visualizzare la pagina di riepilogo. È possibile utilizzare il pulsante **Indietro** per rivisitare le pagine precedenti e apportare correzioni. Se si è soddisfatti delle informazioni, fare clic su **Connetti** per connettersi al gestore code remoto.

  Le informazioni di connessione specificate vengono scritte nel file CCDT nella directory web. Il percorso è *WLP_USER_DIR*/generated.ccdt/ccdt-*uniqueName*.

Risultati

Il gestore code remoto viene visualizzato nell'elenco dei gestori code remoti in IBM MQ Console. Se la connessione ha esito positivo, è possibile gestire gli oggetti del gestore code remoto nello stesso modo in cui si gestiscono gli oggetti di un gestore code locale.

V 9.4.0 *Aggiunta di un gestore code remoto a IBM MQ Console utilizzando la riga comandi*

È possibile aggiungere un gestore code remoto a IBM MQ Console utilizzando il comando **setmqweb remote** sulla riga comandi. Un gestore code remoto può essere un gestore code in esecuzione in un'installazione differente sullo stesso sistema di IBM MQ Consoleoppure un gestore code in esecuzione su un sistema differente.

Prima di iniziare

Nota: La procedura in questa attività richiede l'esecuzione dei comandi MQSC:

- ALW Su AIX, Linux, and Windows, si immettono comandi MQSC da un prompt dei comandi **runmqsc** . Consultare Esecuzione interattiva dei comandi MQSC in **runmqsc** e Esecuzione dei comandi MQSC dai file di testo in **runmqsc**. Per questa attività, se si utilizza AIX, Linux, and Windows, aprire un prompt dei comandi runmqsc che utilizza QM1:

  ```
  runmqsc QM1
  ```

- IBM i Su IBM i, si crea un elenco di comandi in un file Script, quindi si esegue il file utilizzando il comando **STRMQMMQSC** . Consultare Gestione utilizzando i comandi MQSC su IBM i.
- z/OS Su z/OS, i comandi MQSC possono essere emessi da un numero di origini, a seconda del comando. Consultare Le origini da cui è possibile emettere comandi MQSC e PCF su IBM MQ for z/OS.

Assicurarsi che il server mqweb sia configurato per consentire le connessioni del gestore code remoto a IBM MQ Console. Per ulteriori informazioni, fare riferimento alla sezione Configurazione del comportamento delle connessioni del gestore code remoto.

Procedura

1. Sul gestore code remoto, creare un canale di connessione server per consentire la gestione remota del gestore code utilizzando il comando MQSC **DEFINE CHANNEL** .
   Ad esempio, per creare un canale di connessione server QM1.SVRCONN per il gestore code QM1, immettere il comando MQSC seguente:

   ```
   DEFINE CHANNEL(QM1.SVRCONN) CHLTYPE(SVRCONN) TRPTYPE(TCP)
   ```

   Per ulteriori informazioni su **DEFINE CHANNEL** e sulle opzioni disponibili, consultare DEFINE CHANNEL.
2. Verificare che un utente appropriato sia autorizzato a gestire il gestore code e gli oggetti MQ associati al gestore code.
   - ALW Su AIX, Linux, and Windows utilizzare il comando di controllo **setmqaut** su una riga comandi standard.
   - z/OS Su z/OS, definire i profili RACF per fornire all'utente autorizzato l'accesso al gestore code.

   Ad esempio su AIX, Linux, and Windows, per autorizzare l'utente exampleUser ad accedere al gestore code QM1, immettere il seguente comando di controllo:

   ```
   setmqaut -m QM1 -t qmgr -p exampleUser +connect +inq +setall +dsp
   ```

   Questo utente autorizzato potrebbe essere uno dei seguenti:
   - Un ID utente che è lo stesso dell'ID utente che avvia il server mqweb che esegue IBM MQ Console sul sistema da cui si desidera amministrare in remoto questo gestore code.
   - Un ID utente che corrisponde a un ID utente e a una password inclusi nel comando **setmqweb remote** nel passaggio “7” a pagina 109. Includendo l'ID utente e password nel comando **setmqweb remote** , questo ID utente e password vengono utilizzati per l'autenticazione quando IBM MQ Console si connette al gestore code.
   - Un ID utente determinato da regole di sicurezza del canale. Ad esempio, è possibile impostare una regola di autenticazione di canale sul canale di connessione server per consentire le connessioni dall'indirizzo IP da cui si utilizza IBM MQ Console per la gestione remota e associare tutte queste connessioni a uno specifico ID utente autorizzato a utilizzare il gestore code. Per ulteriori informazioni, consultare Creazione di nuove regole CHLAUTH per canali.
3. ALW
   Se non c'è alcun listener in esecuzione sul gestore code remoto, creare un listener per accettare le connessioni di rete in entrata utilizzando il comando MQSC **DEFINE LISTENER** .

Ad esempio, per creare un listener REMOTE.LISTENER sulla porta 1414 per il gestore code remoto QM1, immettere il seguente comando MQSC:

```
runmqsc QM1
DEFINE LISTENER(REMOTE.LISTENER) TRPTYPE(TCP) PORT(1414)
end
```

4. Verificare che il listener sia in esecuzione utilizzando il comando MQSC **START LISTENER** .

ALW Ad esempio, in AIX, Linux, and Windows per avviare il listener REMOTE.LISTENER per il gestore code QM1, immettere il seguente comando MQSC:

```
runmqsc QM1
START LISTENER(REMOTE.LISTENER)
end
```

z/OS Ad esempio, su z/OS, per avviare il listener, immettere il seguente comando MQSC:

```
/cpf START LISTENER TRPTYPE(TCP) PORT(1414)
```

Notare che lo spazio di indirizzo dell'iniziatore di canali deve essere avviato prima di poter avviare un listener su z/OS.

5. Creare un file CCDT JSON che contiene le informazioni di connessione del gestore code remoto:
   - Generare un file CCDT utilizzando il IBM MQ Console associato alla stessa installazione del gestore code a cui si desidera connettersi in remoto.

     Nel pannello **Home** , fare clic sul riquadro **Scarica file di connessione** .
   - Crea un file CCDT in formato JSON che definisce la connessione. Per ulteriori informazioni sulla creazione di una CCDT in formato JSON, consultare Configurazione di una CCDT in formato JSON.

     Il file CCDT deve includere le informazioni name, `clientConnectione` `type` . Facoltativamente, è possibile includere ulteriori informazioni come le informazioni `transmissionSecurity` . Per ulteriori informazioni su tutte le definizioni di attributo del canale CCDT, consultare Elenco completo delle definizioni di attributo del canale CCDT.

     Il seguente esempio mostra un file CCDT JSON di base per una connessione del gestore code remoto. Imposta il nome del canale sullo stesso nome del canale di collegamento server di esempio creato nel passaggio “1” a pagina 108e la porta di connessione sullo stesso valore della porta utilizzata dal listener. L'host di connessione è impostato con il nome host del sistema su cui è in esecuzione il gestore code remoto di esempio, QM1:

```
{
  "channel": [{
      "name": "QM1.SVRCONN",
      "clientConnection": {
        "connection": [{
            "host": "example.com",
            "port": 1414
          }],
        "queueManager": "QM1"
      },
      "type": "clientConnection"
    }]
}
```

6. Copiare il file CCDT JSON nel sistema in cui è in esecuzione IBM MQ Console .
7. Dall'installazione che esegue IBM MQ Console, utilizzare il comando **setmqweb remote** per aggiungere le informazioni sul gestore code remoto alla configurazione IBM MQ Console .

   Come minimo, per aggiungere un gestore code remoto al IBM MQ Console è necessario fornire il nome del gestore code, un nome univoco per il gestore code (per differenziare tra altri gestori code remoti che potrebbero avere lo stesso nome gestore code) e l'URL CCDT per il gestore code. Il nome univoco è il nome di visualizzazione in IBM MQ Console, quindi specificare un nome che renda chiaro che si tratta di un gestore code remoto, ad esempio `"remote-QM2"`. Sono disponibili diverse opzioni aggiuntive che è possibile specificare, ad esempio il nome utente e la password da utilizzare per la connessione

del gestore code remoto o i dettagli del truststore e del keystore. Per un elenco completo di parametri che possono essere specificati con il comando **setmqweb remote** , consultare setmqweb remote.

Ad esempio, per aggiungere il gestore code remoto di esempio QM1, utilizzando il file CCDT di esempio, immettere il seguente comando:

```
setmqweb remote add -uniqueName "MACHINEAQM1" -qmgrName "QM1" -ccdtURL "c:\myccdts\ccdt.json"
```

Risultati

Il gestore code remoto viene visualizzato nell'elenco dei gestori code remoti in IBM MQ Console al successivo aggiornamento dell'elenco delle connessioni remote. Se la connessione ha esito positivo, è possibile gestire gli oggetti del gestore code remoto nello stesso modo in cui si gestiscono gli oggetti di un gestore code locale.

Esempio

Il seguente esempio imposta la connessione del gestore code remoto per il gestore code QM1. IBM MQ Console è autorizzato a gestire il gestore code in base alle autorizzazioni fornite all'utente `exampleUser`. Le credenziali di questo utente sono fornite a IBM MQ Console quando il comando **setmqweb remote** è utilizzato per configurare le informazioni di connessione del gestore code remoto.

1. Sul sistema in cui si trova il gestore code remoto QM1 , vengono creati un canale di connessione server e un listener. Il listener viene avviato e viene concessa l'autorizzazione all'utente `exampleUser` per gestire il gestore code. Ad esempio, su AIX, Linux, and Windows, eseguire i seguenti comandi:

```
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. Sul sistema su cui è in esecuzione IBM MQ Console , viene creato un file `QM1_ccdt.json` con le seguenti informazioni di connessione:

```
{
  "channel": [{
      "name": "QM1.SVRCONN",
      "clientConnection": {
        "connection": [{
            "host": "example.com",
            "port": 1414
          }],
        "queueManager": "QM1"
      },
      "type": "clientConnection"
    }]
}
```

3. Sul sistema su cui è in esecuzione IBM MQ Console , le informazioni sulla connessione del gestore code remoto per il gestore code QM1 vengono aggiunte al server mqweb. Le credenziali per `exampleUser` sono incluse nelle informazioni di connessione:

```
setmqweb remote add -uniqueName "remote-QM1" -qmgrName "QM1" -ccdtURL
"c:\myccdts\QM1_ccdt.json" -username "exampleUser" -password "password"
```

4. La IBM MQ Console mostra il gestore code remoto QM1.

V 9.4.0 IBM MQ Console: operazioni con gli oggetti

A ciascun gestore code IBM MQ sono associati diversi tipi di oggetti.

Informazioni su questa attività

È possibile utilizzare la console per gestire i seguenti tipi di oggetto IBM MQ :

- Code
- Oggetti eventi:
  - Argomenti
  - Sottoscrizioni
- Oggetti applicazioni:
  - Connessioni
  - Canali app
  - Istanze di canale dell'app
- Oggetti di rete MQ :
  - Gestori code connessi
  - Canali gestore code
  - Istanze di canale del gestore code

Procedura

Per gestire un oggetto IBM MQ :

1. Nella vista elenco dei gestori code, selezionare il gestore code proprietario degli oggetti che si desidera gestire.
2. Fare clic sulla scheda di rete Code, Eventi, Applicazioni o MQ per selezionare il tipo di oggetto che si desidera utilizzare.
3. Consultare uno dei seguenti argomenti per istruzioni dettagliate sulla gestione degli oggetti.

V 9.4.0 *IBM MQ Console: gestione delle code*

È possibile visualizzare le code esistenti per un determinato gestore code nella scheda **Code** . È possibile aggiungere ed eliminare code, aggiungere e cancellare messaggi da una coda, sfogliare messaggi, visualizzare e impostare le proprietà di una coda e gestire i record di autorizzazioni di una coda.

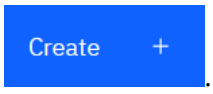
Informazioni su questa attività

La vista delle code elenca le code che esistono per un gestore code specifico. Si accede all'elenco di code facendo clic su un gestore code e selezionando la scheda **Code** . È possibile selezionare singole code dall'elenco da gestire.

z/OS Non è possibile visualizzare o modificare i record di autorizzazione per le code su z/OS.

Procedura

- Per aggiungere una coda:

  a) Nella scheda **Code** , fare clic sul pulsante Crea 

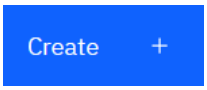
.

  b) Selezionare il tipo di coda che si desidera creare:
  - Coda locale - memorizza i messaggi nel gestore code a cui appartiene.
  - Coda alias - un puntatore a un'altra coda sullo stesso gestore code.
  - Coda remota - un puntatore a un'altra coda su un gestore code differente.
  - Coda modello - un modello per una coda utilizzato quando viene creato un gestore code dinamico.

c) Fornire le informazioni richieste per il tipo di coda che si sta creando. Per impostazione predefinita, vengono visualizzate le proprietà minime consigliate per cui fornire i valori. È possibile visualizzare tutte le proprietà disponibili selezionando **Creazione personalizzata**.

d) Fai clic su **Crea**. Viene creata la nuova coda.

- Per inserire i messaggi in una coda:
  a) Fare clic sulla coda a cui si desidera aggiungere i messaggi nell'elenco nella vista elenco code. Non è possibile selezionare una coda modello.

  b) Fare clic sul pulsante Crea 

  c) Immettere il messaggio che si desidera inserire nella coda.

  d) Fai clic su **Crea**.
- Per cancellare i messaggi da una coda:
  a) Fare clic sulla coda locale da cui si desidera cancellare i messaggi nell'elenco delle code.

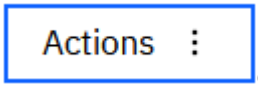
  b) Fare clic sull'icona Cancella coda 

  c) Confermare che si desidera cancellare la coda facendo clic su **Cancella coda**.
- 

  Per eliminare un singolo messaggio da una coda:
  a) Individuare il messaggio che si desidera eliminare.

  b) Fare clic sull'icona di eliminazione accanto al messaggio  .

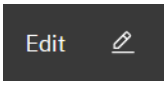
  c) Confermare che si desidera cancellare il messaggio facendo clic su **Elimina**.
- Per sfogliare i messaggi su una coda, fare clic sulla coda nella vista elenco delle code. Viene visualizzato un elenco dei messaggi su tale coda.
- Per eliminare una coda:
  a) Fare clic sulla coda locale che si desidera eliminare nell'elenco delle code.

  b) Fare clic sul pulsante Azioni 

e selezionare **Elimina coda**.

  c) Confermare che si desidera eliminare la coda facendo clic su **Elimina**. La coda è stata eliminata.
- Per visualizzare e modificare le proprietà di una coda:

  a) Selezionare **Visualizza configurazione** dal menu accanto alla coda che si desidera modificare.

  b) Fare clic sul pulsante Modifica 

  c) Modificare le proprietà come desiderato. Se la casella di testo della proprietà è disabilitata, la proprietà è di sola lettura o può essere modificata solo dalla riga comandi. Per informazioni sulle proprietà, consultare Proprietà delle code nella documentazione IBM MQ Explorer

  d) Fare clic su **Salva** per salvare le modifiche.
- Per visualizzare e modificare i record di autorizzazione per una coda:

  a) Selezionare **Visualizza configurazione** dal menu accanto alla coda che si desidera modificare.

  b) Selezionare la scheda **Sicurezza**.

  c) Gestire i record di autorizzazioni come descritto per i record di autorizzazioni del gestore code. Consultare “IBM MQ Console: utilizzo dei record di autorizzazione del gestore code” a pagina 102.

- V 9.4.0
  Per visualizzare gli oggetti IBM MQ associati ad una coda:

  a) Selezionare **Visualizza oggetti associati** dal menu accanto alla coda che si desidera visualizzare.

  b) Visualizzare gli oggetti nel pannello visualizzato. Fare clic sui collegamenti per visualizzare ulteriori dettagli su ciascuno degli oggetti elencati.

  È possibile utilizzare il pannello per visualizzare le applicazioni che inseriscono i messaggi nelle code e le relazioni tra code differenti. Ciò può aiutare a individuare e risolvere i problemi.

V 9.4.0 *IBM MQ Console: operazioni con gli argomenti*

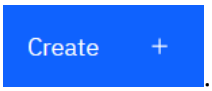
È possibile utilizzare IBM MQ Console per aggiungere ed eliminare argomenti e visualizzare e impostare le proprietà di un argomento.

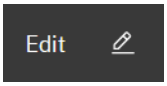
Informazioni su questa attività

La vista degli argomenti elenca gli argomenti esistenti per un gestore code specifico. Si accede agli argomenti dalla scheda **Eventi** del gestore code. È possibile selezionare singoli argomenti dall'elenco da utilizzare.

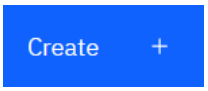
z/OS Non è possibile visualizzare o modificare i record di autorizzazioni per un argomento su z/OS.

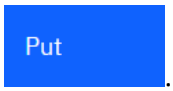
Procedura

- Per aggiungere un argomento:
  a) Dalla vista Gestore code, aprire la scheda **Eventi** e fare clic su **Argomenti**.
  b) Fare clic sul pulsante Crea 

  .
  c) Fornire le informazioni richieste per l'argomento che si sta creando. Per impostazione predefinita, vengono visualizzate le proprietà minime consigliate per cui fornire i valori. È possibile visualizzare tutte le proprietà disponibili selezionando **Creazione personalizzata**.
  d) Fai clic su **Crea**. Viene creato il nuovo argomento.
- Per eliminare un argomento:
  a) Fare clic sull'icona della chiave  accanto all'argomento che si desidera eliminare.
  b) Nella vista Modifica coda, fare clic su **Elimina argomento**.
  c) Confermare che si desidera eliminare l'argomento facendo clic su **Elimina**. L'argomento viene eliminato.
- Per visualizzare e modificare le proprietà di un argomento:
  a) Fare clic sull'icona della chiave  accanto all'argomento che si desidera modificare.
  b) Fare clic sul pulsante Modifica 

  c) Modificare le proprietà come desiderato. Se la casella di testo della proprietà è disabilitata, la proprietà è di sola lettura o può essere modificata solo dalla riga comandi. Per informazioni sulle proprietà, consultare Proprietà degli argomenti nella documentazione di MQ Explorer.
  d) Fare clic su **Salva** per salvare le modifiche.

- Per pubblicare un messaggio su un argomento, è necessario avere almeno una sottoscrizione corrispondente.
  a) Fare clic sull'argomento che si desidera pubblicare nell'elenco di argomenti.
  b) Fare clic sul nome della sottoscrizione corrispondente.
  c) Fare clic sul pulsante Crea 

  d) Immettere il messaggio che si desidera pubblicare.
  e) Fare clic sul pulsante Inserisci 

. Il messaggio viene scritto in tutte le sottoscrizioni corrispondenti.
- Per sottoscrivere un argomento, consultare "IBM MQ Console: operazioni con le sottoscrizioni" a pagina 114:
- Per visualizzare e modificare i record di autorizzazione per un argomento:
  a) Fare clic sull'icona della chiave  accanto all'argomento per cui si desidera modificare i record di autorizzazione.
  b) Selezionare la scheda **Sicurezza**.
  c) Gestire i record di autorizzazione come descritto per i record di autorizzazione del gestore code; consultare "IBM MQ Console: utilizzo dei record di autorizzazione del gestore code" a pagina 102.

V 9.4.0 *IBM MQ Console: operazioni con le sottoscrizioni*

È possibile utilizzare IBM MQ Console per aggiungere ed eliminare le sottoscrizioni e visualizzare e impostare le proprietà di una sottoscrizione.

Informazioni su questa attività

La vista Sottoscrizioni elenca le sottoscrizioni che esistono per un determinato gestore code. Si accede alle sottoscrizioni dalla scheda **Eventi** del gestore code. È possibile selezionare singoli argomenti dall'elenco da utilizzare. È possibile selezionare singole sottoscrizioni dall'elenco da utilizzare.

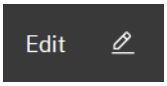
Per ulteriori informazioni sulle sottoscrizioni, consultare Sottoscrittori e sottoscrizioni e DEFINE SUB.

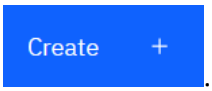
z/OS Non è possibile visualizzare o modificare i record di autorizzazione per una sottoscrizione su z/OS.

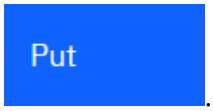
Procedura

- Per aggiungere una sottoscrizione:
  a) Dalla vista Gestore code, aprire la scheda **Eventi** e fare clic su **Sottoscrizioni**.
  b) Scegliere se si desidera creare una sottoscrizione gestita o non gestita.
  c) Fornire le informazioni richieste per la sottoscrizione che si sta creando. Per impostazione predefinita, vengono visualizzate le proprietà minime consigliate per cui fornire i valori. È possibile visualizzare tutte le proprietà disponibili selezionando **Creazione personalizzata**.
  d) Fai clic su **Crea**. Viene creata la nuova sottoscrizione.
- Per eliminare una sottoscrizione:
  a) Fare clic sull'icona della chiave  accanto alla sottoscrizione che si desidera eliminare.
  b) Nella vista Modifica coda, fare clic su **Elimina sottoscrizione**.

c) Confermare che si desidera eliminare la sottoscrizione facendo clic su **Elimina**. La sottoscrizione viene eliminata.

- Per visualizzare e modificare le proprietà di una sottoscrizione:
  a) Fare clic sull'icona della chiave  accanto alla sottoscrizione che si desidera modificare.
  b) Fare clic sul pulsante Modifica 

  c) Modificare le proprietà come desiderato. Se la casella di testo della proprietà è disabilitata, la proprietà è di sola lettura o può essere modificata solo dalla riga comandi.
  d) Fare clic su **Salva** per salvare le modifiche.
- Per pubblicare un messaggio sull'argomento a cui è sottoscritta la sottoscrizione:
  a) Fare clic sulla sottoscrizione di cui si desidera pubblicare l'argomento nell'elenco di sottoscrizioni.
  b) Fare clic sul pulsante Crea 

  .
  c) Immettere il messaggio che si desidera pubblicare.
  d) Fare clic sul pulsante Inserisci 

  . Il messaggio viene scritto in tutte le sottoscrizioni che corrispondono all'argomento pubblicato.

V 9.4.0 *IBM MQ Console: utilizzo dei canali del gestore code*

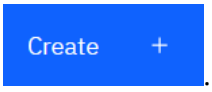
È possibile utilizzare le operazioni IBM MQ Console con i canali del gestore code: è possibile aggiungere ed eliminare canali del gestore code, avviare e arrestare canali, reimpostare e risolvere canali e eseguire il ping dei canali. È inoltre possibile visualizzare e impostare le proprietà di un canale del gestore code e gestire i record di autorizzazioni per il canale.

Informazioni su questa attività

Un canale gestore code è un link di comunicazione logica per la trasmissione dei messaggi tra i gestori code in una rete. La vista del canale del gestore code include un pannello che mostra una vista rapida del numero di canali in esecuzione, del numero di nuovi tentativi e del numero di arrestati.

z/OS Non è possibile visualizzare o modificare i record di autorizzazione per un canale su z/OS.

Procedura

- Per aggiungere un canale del gestore code:
  a) Dalla vista Gestore code, aprire la scheda **Rete diMQ** , fare clic sul pulsante **Canali gestore code** e fare clic sul pulsante Crea 

  .
  b) Selezionare il tipo di canale del gestore code che si desidera creare e fare clic sul pulsante Avanti 

  .
  c) Fornire le informazioni richieste per il canale che si sta creando. Per impostazione predefinita, vengono visualizzate le proprietà minime consigliate per cui fornire i valori. È possibile visualizzare tutte le proprietà disponibili selezionando **Creazione personalizzata**.
  d) Fai clic su **Crea**. Il nuovo canale viene creato con lo stato **inactive** .
- Per avviare un canale del gestore code:

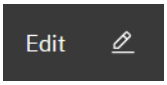
a) Individuare il canale che si desidera avviare nell'elenco.

b) Selezionare **Avvia** dal menu ⋮.

- Per arrestare un canale del gestore code:
  a) Individuare il canale che si desidera arrestare nell'elenco.

  b) Selezionare **Arresta** dal menu ⋮.
- Per visualizzare le proprietà di un canale del gestore code:
  a) Individuare il canale nell'elenco.

  b) Selezionare **Visualizza configurazioni** dal menu ⋮.

  c) Verificare che la pagina **Proprietà** sia selezionata. Per modificare le proprietà, fare clic sul pulsante Modifica 

  d) Modificare le proprietà come desiderato. Se la casella di testo della proprietà è disabilitata, la proprietà è di sola lettura o può essere modificata solo dalla riga comandi. Per ulteriori informazioni sulle proprietà, consultare Proprietà del canale nella documentazione di MQ Explorer.

  e) Fare clic su **Salva** per salvare le modifiche.
- Per reimpostare un canale del gestore code:
  a) Individuare il canale nell'elenco.

  b) Selezionare **Avanzate** dal menu ⋮.

  c) Nella sezione **Reimposta** , specificare un numero di sequenza del messaggio.
  È necessario reimpostare un canale se non si avvia perché le due estremità non sono d'accordo sul numero di sequenza del messaggio successivo da inviare. Il numero di sequenza del messaggio specifica tale numero.

  d) Fare clic su **Reimposta canale**.
- Per risolvere un canale mittente o server:
  a) Individuare il canale nell'elenco.

  b) Selezionare **Avanzate** dal menu ⋮.

  c) Nella sezione **Risolvi** , scegliere se eseguire il commit o il backout del batch di messaggi corrente facendo clic su **Ripristina messaggi nella coda di trasmissione** o su **Elimina messaggi**.
- Per eseguire il ping di un canale del gestore code:
  a) Individuare il canale nell'elenco.

  b) Selezionare **Ping** dal menu ⋮.
- Per visualizzare e modificare i record di autorizzazione per un canale del gestore code:
  a) Individuare il canale nell'elenco.

  b) Selezionare **Visualizza configurazione** dal menu ⋮.

  c) Selezionare la scheda **Sicurezza**.

  d) Gestire i record di autorizzazione come descritto per i record di autorizzazione del gestore code; consultare “IBM MQ Console: utilizzo dei record di autorizzazione del gestore code” a pagina 102.
- Per eliminare un canale del gestore code:

a) Individuare il canale nell'elenco.

b) Selezionare **Configura** dal menu .

c) Fare clic su **Elimina canale**.

V 9.4.0 *IBM MQ Console: utilizzo dei canali dell'applicazione*

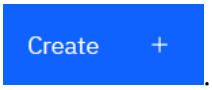
È possibile utilizzare IBM MQ Console per gestire i canali dell'applicazione: è possibile aggiungere ed eliminare canali, avviare e arrestare canali, ripristinare e risolvere canali e eseguire il ping dei canali. È inoltre possibile visualizzare e impostare le proprietà di un canale dell'applicazione e gestire i record di autorizzazioni per il canale.

Informazioni su questa attività

Un canale applicazione è un link di comunicazione logica, utilizzato dalle applicazioni per connettersi a un gestore code in una rete. La vista del canale dell'applicazione include un pannello che mostra una vista rapida del numero di canali in esecuzione, del numero di nuovi tentativi e del numero di arrestati.

z/OS Non è possibile visualizzare o modificare i record di autorizzazione per un canale su z/OS.

Procedura

- Per aggiungere un canale dell'applicazione:
  a) Dalla vista del gestore code, aprire la scheda **Applicazioni** e fare clic su **Canali app** e fare clic sul pulsante Crea 

.

  b) Fare clic sul pulsante Avanti 

.

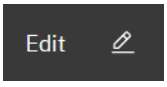
  c) Fornire le informazioni richieste per il canale che si sta creando. Per impostazione predefinita, vengono visualizzate le proprietà minime consigliate per cui fornire i valori. È possibile visualizzare tutte le proprietà disponibili selezionando **Creazione personalizzata**.

  d) Fai clic su **Crea**. Il nuovo canale viene creato con lo stato **inactive** .
- Per avviare un canale dell'applicazione:
  a) Individuare il canale che si desidera avviare nell'elenco.

  b) Selezionare **Avvia** dal menu .
- Per arrestare un canale dell'applicazione:
  a) Individuare il canale che si desidera arrestare nell'elenco.

  b) Selezionare **Arresta** dal menu .
- Per visualizzare le proprietà di un canale dell'applicazione:
  a) Individuare il canale nell'elenco.

  b) Selezionare **Visualizza configurazione** dal menu .

  c) Verificare che la pagina **Proprietà** sia selezionata. Per modificare le proprietà, fare clic sul pulsante Modifica 

d) Modificare le proprietà come desiderato. Se la casella di testo della proprietà è disabilitata, la proprietà è di sola lettura o può essere modificata solo dalla riga comandi. Per ulteriori informazioni sulle proprietà, consultare Proprietà del canale nella documentazione di MQ Explorer.

e) Fare clic su **Salva** per salvare le modifiche.

- Per reimpostare un canale dell'applicazione:
  a) Individuare il canale nell'elenco.
  b) Selezionare **Avanzate** dal menu ⋮.
  c) Nella sezione **Reimposta** , specificare un numero di sequenza del messaggio.
  È necessario reimpostare un canale se non si avvia perché le due estremità non sono d'accordo sul numero di sequenza del messaggio successivo da inviare. Il numero di sequenza del messaggio specifica tale numero.
  d) Fare clic su **Reimposta canale**.
- Per risolvere un canale mittente o server:
  a) Individuare il canale nell'elenco.
  b) Selezionare **Avanzate** dal menu ⋮.
  c) Nella sezione **Risolvi** , scegliere se eseguire il commit o il backout del batch di messaggi corrente facendo clic su **Ripristina messaggi nella coda di trasmissione** o su **Elimina messaggi**.
- Per emettere un comando ping per un canale:
  a) Individuare il canale nell'elenco.
  b) Selezionare **Ping** dal menu ⋮.
- Per visualizzare e modificare i record di autorizzazione per un canale dell'applicazione:
  a) Individuare il canale nell'elenco.
  b) Selezionare **Configura** dal menu ⋮.
  c) Selezionare la scheda **Sicurezza**.
  d) Gestire i record di autorizzazione come descritto per i record di autorizzazione del gestore code; consultare “IBM MQ Console: utilizzo dei record di autorizzazione del gestore code” a pagina 102 .
- Per eliminare un canale dell'applicazione:
  a) Individuare il canale nell'elenco.
  b) Selezionare **Configura** dal menu ⋮.
  c) Fare clic su **Elimina canale**.

V 9.4.0 *IBM MQ Console: operazioni con le applicazioni*

È possibile utilizzare IBM MQ Console per visualizzare informazioni sulle applicazioni connesse a un gestore code.

Informazioni su questa attività

Un'applicazione è connessa a un gestore code attraverso una rete utilizzando un canale server - conn. La vista Applicazioni include un pannello che mostra una vista rapida del numero di applicazioni connesse a un gestore code.

Procedura

- Per visualizzare le informazioni sull'applicazione:
  a) Dalla vista del gestore code, aprire la scheda **Applicazioni** .
  b) Fare clic su **Applicazioni connesse** per aprire la vista applicazioni.
  c) Se sono presenti più istanze di un'applicazione, fare clic sulla freccia verso il basso per visualizzare i dettagli di ciascuna istanza.
  d) Fare clic sugli oggetti nella vista per ottenere ulteriori dettagli.

V 9.4.0 z/OS *IBM MQ Console: Working with storage classes*

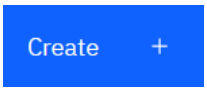
You can use the IBM MQ Console to add, view, delete and update storage classes on z/OS queue managers.

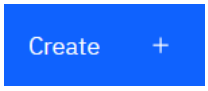
About this task

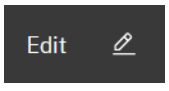
The storage classes view lists the storage classes that exist for a specific queue manager. You access **Storage classes** from the sidebar on the queue manager **Queues** tab.

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

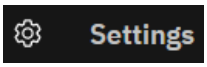
- To view the properties of a page set

  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 impostazioni

È possibile specificare alcune impostazioni generali per IBM MQ Console.

Fare clic sull'icona delle impostazioni 

per passare alla vista delle impostazioni IBM MQ Console .

Utilizzare le impostazioni per controllare le funzioni seguenti:

- Aggiornamento automatico dei gestori code ogni 10 secondi. Questa funzione può essere attivata o disattivata.
- Indica se gli oggetti di sistema vengono visualizzati. È possibile specificare questo valore per tutti i tipi di oggetto oppure selezionare i tipi di oggetto singolarmente.
- Se le informazioni di traccia vengono raccolte o meno.

Linux Windows Amministrazione mediante IBM MQ Explorer

IBM MQ Explorer consente di eseguire la gestione locale o remota della rete solo da un computer su cui è in esecuzione Windowso Linux x86-64 .

IBM MQ for Windows e IBM MQ per Linux x86-64 forniscono un'interfaccia di gestione denominata IBM MQ Explorer per eseguire attività di gestione come alternativa all'utilizzo dei comandi di controllo o MQSC. Confronto delle serie di comandi mostra le operazioni che è possibile effettuare utilizzando IBM MQ Explorer.

Il IBM MQ Explorer consente di eseguire la gestione locale o remota della rete da un computer su cui è in esecuzione Windowso Linux x86-64, puntando il IBM MQ Explorer sui gestori code e sui cluster a cui si è interessati. Può connettersi in remoto ai gestori code in esecuzione su qualsiasi piattaforma supportata, tra cui z/OS, consentendo di visualizzare, esplorare e modificare l'intera struttura portante di messaggistica dalla console.

Per configurare i gestori code IBM MQ remoti in maniera che IBM MQ Explorer possa gestirli, consultare “Software prerequisito e definizioni per IBM MQ Explorer” a pagina 123.

Consente di eseguire attività, di solito associate all'impostazione e all'ottimizzazione dell'ambiente di lavoro per IBM MQ, localmente o in remoto all'interno di un dominio di sistema Windows o Linux x86-64 .

Su Linux, l'avvio di IBM MQ Explorer potrebbe non riuscire se si dispone di più di un'installazione di Eclipse . Se ciò si verifica, avviare IBM MQ Explorer utilizzando un ID utente diverso da quello utilizzato per l'altra installazione di Eclipse .

Su Linux, per avviare correttamente IBM MQ Explorer , è necessario essere in grado di scrivere un file nella propria directory home e la directory home deve esistere.

IBM MQ Explorer può essere installato dal download IBM MQ Explorer autonomo disponibile da Fix Central. Per ulteriori informazioni, consultare Installazione e disinstallazione IBM MQ Explorer come applicazione autonoma su Linux e Windows.

Linux Windows Operazioni che è possibile eseguire con IBM MQ Explorer

È possibile utilizzare IBM MQ Explorer per eseguire attività di gestione utilizzando una serie di finestre di dialogo Proprietà e Viste contenuto. È anche possibile estendere IBM MQ Explorer scrivendo uno o più plug-in Eclipse .

IBM MQ Explorer attività

Con IBM MQ Explorer, è possibile effettuare le seguenti attività:

- Creare ed eliminare un gestore code (solo sulla macchina locale).
- Avviare e arrestare un gestore code (solo sulla macchina locale).
- Definire, visualizzare e modificare le definizioni di IBM MQ oggetti come code e canali.
- Sfoglia i messaggi su una coda.
- Avviare e arrestare un canale.
- Visualizzare le informazioni di stato su un canale, listener, coda o oggetti di servizio.
- Visualizzare i gestori code in un cluster.
- Verificare quali applicazioni, utenti o canali hanno una particolare coda aperta.
- Creare un nuovo cluster di gestori code utilizzando la procedura guidata Crea nuovo cluster.
- Aggiungere un gestore code a un cluster utilizzando la procedura guidata Aggiungi gestore code al cluster.
- Gestire l'oggetto delle informazioni di autenticazione, utilizzato con la sicurezza del canale TLS (Transport Layer Security).
- Creare ed eliminare gli iniziatori di canali, i controlli dei trigger e i listener.
- Avviare o arrestare i server dei comandi, gli iniziatori di canali, i controlli dei triggere i listener.
- Impostare specifici servizi su avvio automatico all'avvio di un gestore code.
- Modificare le proprietà dei gestori code.
- Modificare il gestore code predefinito locale.
- Creare JMS oggetti da IBM MQ oggettie IBM MQ oggetti da JMS oggetti.
- Creare un JMS factory di connessione per uno qualsiasi dei tipi attualmente supportati.
- Modificare i parametri per qualsiasi servizio, come il numero di porta TCP per un listener o il nome della coda dell'iniziatore di canali.

- Avviare o arrestare la traccia del servizio.

Finestre di dialogo Proprietà e Viste contenuto

Le attività di gestione vengono eseguite utilizzando una serie di finestre di dialogo Proprietà e Viste contenuto.

Vista Contenuto
Una vista Contenuto è un pannello che può visualizzare quanto segue:

- Attributi e opzioni di gestione relativi a IBM MQ .
- Attributi e opzioni di gestione relativi a uno o più oggetti correlati.
- Attributi e opzioni di amministrazione per un cluster.

Finestre delle proprietà
Una finestra di dialogo delle proprietà è un pannello che visualizza gli attributi relativi ad un oggetto in una serie di campi, alcuni dei quali è possibile modificare.

Si naviga attraverso IBM MQ Explorer utilizzando la vista Navigator . Il Navigator consente di selezionare la vista Contenuto richiesta.

Estensione di IBM MQ Explorer

IBM MQ Explorer presenta le informazioni in uno stile congruente con quello del framework Eclipse e con le altre applicazioni plug-in supportate da Eclipse .

Attraverso l'estensione di IBM MQ Explorer, gli amministratori di sistema hanno la possibilità di personalizzare IBM MQ Explorer per migliorare il modo in cui amministrano IBM MQ.

Per ulteriori informazioni, consultare Estensione di MQ Explorer.

Linux Windows Decidere se utilizzare il IBM MQ Explorer

Quando si decide se utilizzare IBM MQ Explorer durante l'installazione, considerare le informazioni elencate in questo argomento.

È necessario essere consapevoli dei seguenti punti:

Nomi oggetto
Se si utilizzano nomi minuscoli per i gestori code e altri oggetti con IBM MQ Explorer, quando si utilizzano gli oggetti utilizzando i comandi MQSC, è necessario racchiudere i nomi oggetto tra virgolette singole oppure IBM MQ non li riconosce.

Gestori code di grandi dimensioni
Il IBM MQ Explorer funziona meglio con i gestori code di piccole dimensioni. Se si dispone di un numero elevato di oggetti su un singolo gestore code, potrebbero verificarsi dei ritardi mentre IBM MQ Explorer estrae le informazioni richieste da presentare in una vista.

Cluster
I cluster IBM MQ possono potenzialmente contenere centinaia o migliaia di gestori code. Il IBM MQ Explorer presenta i gestori code in un cluster utilizzando una struttura ad albero. La dimensione fisica di un cluster non influenza la velocità di IBM MQ Explorer in modo significativo perché IBM MQ Explorer non si connette ai gestori code nel cluster finché non vengono selezionati.

Impostazione di IBM MQ Explorer

Questa sezione illustra i passi che è necessario eseguire per configurare IBM MQ Explorer.

- “Software prerequisito e definizioni per IBM MQ Explorer” a pagina 123
- “Sicurezza per IBM MQ Explorer” a pagina 123
- “Visualizzazione e nascondimento di gestori code e cluster in IBM MQ Explorer” a pagina 127
- “Appartenenza al cluster e IBM MQ Explorer” a pagina 127

- “Conversione dati per IBM MQ Explorer” a pagina 128

Software prerequisito e definizioni per IBM MQ Explorer

Assicurarsi di soddisfare i seguenti requisiti prima di provare a utilizzare IBM MQ Explorer.

IBM MQ Explorer può connettersi ai gestori code remoti utilizzando solo il protocollo di comunicazione TCP/IP.

Verificare che:

1. Un server dei comandi è in esecuzione su ogni gestore code gestito in remoto.
2. Un oggetto listener TCP/IP adatto deve essere in esecuzione su ogni gestore code remoto. Questo oggetto può essere il listener IBM MQ o, sui sistemi AIX and Linux , il daemon inetd.
3. Un canale di connessione server, per impostazione predefinita denominato SYSTEM.ADMIN.SVRCONN, esiste su tutti i gestori code remoti.

   È possibile creare il canale utilizzando il seguente comando MQSC:

   ```
   DEFINE CHANNEL(SYSTEM.ADMIN.SVRCONN) CHLTYPE(SVRCONN)
   ```

   Questo comando crea una definizione di canale di base. Se si desidera una definizione più sofisticata (per impostare la sicurezza, ad esempio), sono necessari ulteriori parametri. Per ulteriori informazioni, vedere DEFINE CHANNEL.
4. La coda di sistema, SYSTEM.MQEXPLORER.REPLY.MODEL, deve esistere.

Sicurezza per IBM MQ Explorer

Se si utilizza IBM MQ in un ambiente in cui è importante controllare l'accesso utente a determinati oggetti, potrebbe essere necessario considerare gli aspetti di sicurezza dell'utilizzo di IBM MQ Explorer.

Autorizzazione all'utilizzo di IBM MQ Explorer

Qualsiasi utente può utilizzare IBM MQ Explorer, ma sono necessarie determinate autorizzazioni per connettersi, accedere e gestire i gestori code.

Per eseguire le attività amministrative locali utilizzando IBM MQ Explorer, è necessario che un utente disponga dell'autorizzazione necessaria per eseguire le attività amministrative. Se l'utente è un membro del gruppo mqm , dispone dell'autorizzazione per eseguire tutte le attività amministrative locali.

Per connettersi a un gestore code remoto ed eseguire attività di amministrazione remota utilizzando IBM MQ Explorer, l'utente che esegue IBM MQ Explorer deve disporre delle seguenti autorizzazioni:

- Autorizzazione CONNECT sull'oggetto gestore code di destinazione
- Autorizzazione INQUIRE sull'oggetto gestore code di destinazione
- Autorizzazione DISPLAY per l'oggetto gestore code di destinazione
- Autorizzazione INQUIRE per la coda, SYSTEM.MQEXPLORER.REPLY.MODEL
- Autorizzazione DISPLAY alla coda, SYSTEM.MQEXPLORER.REPLY.MODEL
- Autorizzazione INPUT (get) per la coda, SYSTEM.MQEXPLORER.REPLY.MODEL
- Autorizzazione OUTPUT (inserimento) per la coda, SYSTEM.MQEXPLORER.REPLY.MODEL
- Autorizzazione OUTPUT (inserimento) per la coda, SYSTEM.ADMIN.COMMAND.QUEUE
- Autorizzazione INQUIRE sulla coda, SYSTEM.ADMIN.COMMAND.QUEUE
- Autorizzazione per eseguire l'azione selezionata

Nota: L'autorizzazione INPUT è relativa all'input per l'utente da una coda (un'operazione get). L'autorizzazione OUTPUT si riferisce all'output dall'utente ad una coda (un'operazione di inserimento).

Per connettersi a un gestore code remoto su IBM MQ for z/OS ed eseguire attività di gestione remota utilizzando IBM MQ Explorer, è necessario fornire quanto segue:

- Un profilo RACF per la coda di sistema, SYSTEM.MQEXPLORER.REPLY.MODEL
- Un profilo RACF per le code, AMQ.MQEXPLORER.*

Inoltre, l'utente che esegue IBM MQ Explorer deve disporre delle seguenti autorizzazioni:

- Autorizzazione RACF UPDATE per la coda di sistema, SYSTEM.MQEXPLORER.REPLY.MODEL
- RACF Autorizzazione UPDATE per le code, AMQ.MQEXPLORER.*
- Autorizzazione CONNECT sull'oggetto gestore code di destinazione
- Autorizzazione per eseguire l'azione selezionata
- Autorizzazione READ per tutti i profili hlq.DISPLAY.object nella classe MQCMDS

Per informazioni su come concedere l'autorizzazione agli oggetti IBM MQ , consultare Concessione dell'accesso a un oggetto IBM MQ su sistemi AIX, Linux, and Windows.

Se un utente tenta di eseguire un'operazione che non è autorizzato ad eseguire, il gestore code di destinazione richiama le procedure di errore di autorizzazione e l'operazione non riesce.

Il filtro predefinito in IBM MQ Explorer è visualizzare tutti gli oggetti IBM MQ . Se vi sono oggetti IBM MQ per i quali un utente non dispone dell'autorizzazione DISPLAY, vengono generati errori di autorizzazione. Se gli eventi di autorizzazione vengono registrati, limitare l'intervallo di oggetti visualizzati agli oggetti per i quali l'utente dispone dell'autorizzazione DISPLAY.

Sicurezza per la connessione a gestori code remoti da IBM MQ Explorer

È necessario proteggere il canale tra IBM MQ Explorer e ciascun gestore code remoto.

IBM MQ Explorer si collega ai gestori code remoti come applicazione client MQI. Ciò significa che ogni gestore code remoto deve avere una definizione di un canale di connessione server e un listener TCP/IP adatto. Se non si protegge il canale di connessione server, è possibile che un'applicazione dannosa si connetta allo stesso canale di connessione server e ottenga l'accesso agli oggetti del gestore code con autorizzazione illimitata. Per proteggere il canale di connessione del server, specificare un valore non vuoto per l'attributo MCAUSER del canale, utilizzare i record di autenticazione del canale o utilizzare un'uscita di sicurezza.

Il valore predefinito dell'attributo MCAUSER è l'ID utente locale. Se si specifica un nome utente non vuoto come attributo MCAUSER del canale di connessione del server, tutti i programmi che si collegano al gestore code utilizzando questo canale vengono eseguiti con l'identità dell'utente denominato e dispongono dello stesso livello di autorizzazione. Ciò non si verifica se si utilizzano i record di autenticazione di canale.

Utilizzo di un'uscita di sicurezza con IBM MQ Explorer

È possibile specificare un'uscita di sicurezza predefinita e uscite di sicurezza specifiche del gestore code utilizzando IBM MQ Explorer.

È possibile definire un'uscita di sicurezza predefinita, che può essere utilizzata per tutte le connessioni client nuove da IBM MQ Explorer. Questa uscita predefinita può essere sovrascritta al momento della connessione. È anche possibile definire un'uscita di sicurezza per un singolo gestore code o una serie di gestori code, che diventa effettiva quando viene effettuata una connessione. Le uscite vengono specificate utilizzando IBM MQ Explorer. Per ulteriori informazioni, vedere l'Aiuto di IBM MQ Explorer.

Utilizzo di IBM MQ Explorer per stabilire una connessione a un gestore code remoto utilizzando i canali MQI abilitati a TLS

IBM MQ Explorer si connette ai gestori code remoti utilizzando un canale MQI. Se si desidera proteggere il canale MQI utilizzando la sicurezza TLS, è necessario stabilire il canale utilizzando una tabella di definizione del canale client.

Per informazioni su come stabilire un canale MQI utilizzando una tabella di definizione del canale client, consultare IBM MQ MQI clients.

Una volta stabilito il canale utilizzando una tabella di definizione del canale del client, è possibile utilizzare IBM MQ Explorer per connettersi a un gestore code remoto utilizzando il canale MQI abilitato a TLS, come

descritto in “Attività sul sistema che ospita il gestore code remoto” a pagina 125 e “Attività sul sistema che ospita IBM MQ Explorer” a pagina 125.

Attività sul sistema che ospita il gestore code remoto

Sul sistema che ospita il gestore code remoto, effettuare le seguenti attività:

1. Definire una coppia di canali connessione server e client e specificare il valore appropriato per l'attributo *SSLCIPH* sulla connessione server su entrambi i canali. Per ulteriori informazioni sull'attributo *SSLCIPH* , consultare Protezione dei canali con TLS.
2. Inviare la tabella di definizione del canale `AMQCLCHL.TAB`, che si trova nella directory `@ipcc` del gestore code, al sistema su cui è presente IBM MQ Explorer.
3. Avviare un listener TCP/IP su una porta designata.
4. Collocare i certificati CA e TLS personali nella directory SSL del gestore code:
   - Linux AIX `/var/mqm/qmgrs/+QMNAME+/SSL` per sistemi AIX and Linux .
   - Windows `C:\Program Files\IBM\MQ\qmgrs\+QMNAME+\SSL` per sistemi Windows .

   Dove `+QMNAME+` è un token che rappresenta il nome del gestore code.
5. Creare un file database di chiavi di tipo CMS denominato `key.kdb`. Eseguire lo stash della password del database di chiavi in un file specificando il parametro `-stash` nel comando **runmqakm** utilizzato per creare il database di chiavi.
6. Aggiungere i certificati CA al database delle chiavi creato nel passo precedente.
7. Importare il certificato personale per il gestore code nel database delle chiavi.

Per informazioni più dettagliate sull'utilizzo di TLS su sistemi Windows , consulta Utilizzo di TLS su AIX, Linux, and Windows.

Attività sul sistema che ospita IBM MQ Explorer

Sul sistema che ospita IBM MQ Explorer, effettuare le seguenti attività:

1. Creare un file database delle chiavi di tipo JKS denominato `key.jks`. Impostare una password per questo file di database delle chiavi.

   Il keystore utilizzato da IBM MQ Explorer per la sicurezza TLS deve essere un file JKS (Java keystore).
2. Aggiungere i certificati CA al database delle chiavi creato nel passo precedente.
3. Importare il certificato personale per il gestore code nel database delle chiavi.
4. Su sistemi Windows e Linux , avviare IBM MQ Explorer utilizzando il menu di sistema, il file eseguibile `MQExplorer` o il comando **strmqcfg** .
5. Dalla barra degli strumenti di IBM MQ Explorer , fare clic su **Finestra -> Preferenze**, quindi espandere **IBM MQ Explorer** e selezionare **Archivi certificati client SSL**. Immettere il nome e la password per il file JKS creato nel passo 1 di “Attività sul sistema che ospita IBM MQ Explorer” a pagina 125, sia nell'archivio certificati attendibili che nell'archivio certificati personali, quindi fare clic su **OK**.
6. Chiudere la finestra **Preferenze** e fare clic con il tasto destro del mouse su **Gestori code**. Fare clic su **Mostra / nascondi gestori code**, quindi su **Aggiungi** nella schermata **Mostra / nascondi gestori code** .
7. Immettere il nome del gestore code e selezionare l'opzione **Connetti direttamente** . Fare clic su Avanti.
8. Selezionare **Utilizza tabella di definizione del canale client (CCDT)** e specificare l'ubicazione del file di tabella del canale trasferito dal gestore code remoto nel passo 2 in “Attività sul sistema che ospita il gestore code remoto” a pagina 125 sul sistema su cui si trova il gestore code remoto.
9. Fare clic su **Fine**. È ora possibile accedere al gestore code remoto da IBM MQ Explorer.

Connessione tramite un altro gestore code con IBM MQ Explorer

IBM MQ Explorer consente di connettersi a un gestore code tramite un gestore code intermedio, a cui IBM MQ Explorer è già connesso.

In questo caso, IBM MQ Explorer inserisce i messaggi di comando PCF nel gestore code intermedio, specificando quanto segue:

- Il parametro *ObjectQMgrName* nel descrittore oggetto (MQOD) come nome del gestore code di destinazione. Per ulteriori informazioni sulla risoluzione dei nomi delle code, consultare Risoluzione dei nomi.
- Il parametro *UserIdentifier* nel descrittore del messaggio (MQMD) come userIdlocale.

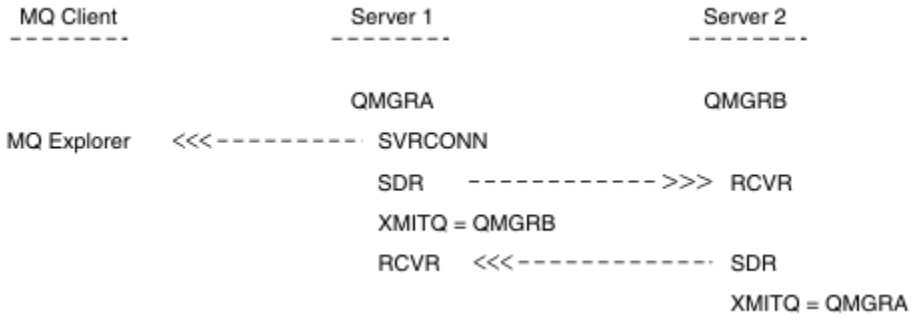
Se la connessione viene quindi utilizzata per connettersi al gestore code di destinazione tramite un gestore code intermedio, l' userId viene trasmesso nuovamente nel parametro *UserIdentifier* del descrittore del messaggio (MQMD). Per consentire al listener MCA sul gestore code di destinazione di accettare questo messaggio, è necessario impostare l'attributo MCAUSER oppure è necessario che userId esista già con l'autorizzazione di inserimento.

Il server dei comandi sul gestore code di destinazione inserisce i messaggi nella coda di trasmissione specificando l' userId nel parametro *UserIdentifier* nel descrittore del messaggio (MQMD). Perché questo inserimento abbia esito positivo, è necessario che l' userId esista già sul gestore code di destinazione con autorizzazione di inserimento.

Il seguente esempio mostra come connettere un gestore code, tramite un gestore code intermedio, a IBM MQ Explorer.

Stabilire una connessione di amministrazione remota a un gestore code. Verificare che:

- Il gestore code sul server è attivo e ha un canale di connessione server (SVRCONN) definito.
- Il listener è attivo.
- Il server dei comandi è attivo.
- SYSTEM.MQ EXPLORER.REPLY.MODEL è stata creata e si dispone dell'autorità sufficiente.
- I listener del gestore code, i server dei comandi e i canali mittente vengono avviati.



In questo esempio:

- IBM MQ Explorer è connesso al gestore code QMGRA (in esecuzione su Server1) utilizzando una connessione client.
- Il gestore code QMGRB su Server2 può ora essere connesso a IBM MQ Explorer tramite un gestore code intermedio (QMGRA)
- Quando ci si connette a QMGRB con IBM MQ Explorer, selezionare QMGRA come gestore code intermedio

In questa condizione, non esiste una connessione diretta a QMGRB da IBM MQ Explorer; la connessione a QMGRB avviene tramite QMGRA.

Il gestore code QMGRB su Server2 si connette a QMGRA su Server1 utilizzando i canali mittente - destinatario. Il canale tra QMGRA e QMGRB deve essere configurato in modo tale che l'amministrazione remota sia possibile; consultare “Configurazione dei gestori code per la gestione remota” a pagina 199.

Visualizzazione e nascondimento di gestori code e cluster in IBM MQ Explorer

IBM MQ Explorer può visualizzare più di un gestore code alla volta. Dal pannello Mostra / Nascondi gestore code (selezionabile dal menu per il nodo della struttura ad albero Gestori code), è possibile scegliere se visualizzare le informazioni su un'altra macchina (remota). I gestori code locali vengono rilevati automaticamente.

Per visualizzare un gestore code remoto:

1. Fare clic con il pulsante destro del mouse sul nodo della struttura **Gestori code** , quindi selezionare **Mostra / Nascondi gestori code**.
2. Fare clic su **Aggiungi**. Viene visualizzato il pannello Mostra / nascondi gestori code.
3. Immettere il nome del gestore code remoto e il nome host o l'indirizzo IP nei campi forniti.

   Il nome host o l'indirizzo IP viene utilizzato per stabilire una connessione client al gestore code remoto utilizzando il relativo canale di connessione server predefinito, SYSTEM.ADMIN.SVRCONNo un canale di connessione server definito dall'utente.
4. Fare clic su **Fine**.

Il pannello Mostra / nascondi gestori code visualizza anche un elenco di tutti i gestori code visibili. È possibile utilizzare questo pannello per nascondere i gestori code dalla vista di navigazione.

Se il IBM MQ Explorer visualizza un gestore code che è un membro di un cluster, il cluster viene rilevato e visualizzato automaticamente.

Per esportare l'elenco di gestori code remoti da questo pannello:

1. Chiudere il pannello Mostra / nascondi gestori code.
2. Fare clic con il pulsante destro del mouse sul nodo della struttura ad albero **IBM MQ** più alto nel riquadro di navigazione di IBM MQ Explorer, quindi selezionare **Esporta impostazioni IBM MQ Explorer**
3. Fare clic su **IBM MQ Explorer > IBM MQ Explorer Impostazioni**
4. Selezionare **Informazioni di connessione> Gestori code remoti**.
5. Selezionare un file in cui memorizzare le impostazioni esportate.
6. Infine, fare clic su **Fine** per esportare le informazioni di connessione del gestore code remoto nel file specificato.

Per importare un elenco di gestori code remoti:

1. Fare clic con il pulsante destro del mouse sul nodo della struttura ad albero **IBM MQ** più alto nel riquadro di navigazione di IBM MQ Explorer, quindi selezionare **Importa impostazioni IBM MQ Explorer**
2. Fare clic su **IBM MQ Explorer > IBM MQ Explorer Impostazioni**
3. Fare clic su **Sfoglia**e passare al percorso del file che contiene le informazioni di connessione del gestore code remoto.
4. Fare clic su **Apri**. Se il file contiene un elenco di gestori code remoti, la casella **Informazioni di connessione> Gestori code remoti** è selezionata.
5. Infine, fare clic su **Fine** per importare le informazioni di connessione del gestore code remoto in IBM MQ Explorer.

Appartenenza al cluster e IBM MQ Explorer

IBM MQ Explorer richiede informazioni sui gestori code che sono membri di un cluster.

Se un gestore code è membro di un cluster, il nodo della struttura ad albero del cluster verrà popolato automaticamente.

Se i gestori code diventano membri dei cluster mentre IBM MQ Explorer è in esecuzione, è necessario mantenere il IBM MQ Explorer con i dati di amministrazione aggiornati sui cluster in modo che possano

comunicare in modo efficace con essi e visualizzare le informazioni sul cluster corrette quando richiesto. Per fare ciò, IBM MQ Explorer ha bisogno delle seguenti informazioni:

- Il nome di un gestore code del repository
- Il nome della connessione del gestore code del repository se si trova su un gestore code remoto

Con queste informazioni, IBM MQ Explorer può:

- Utilizzare il gestore code del repository per ottenere un elenco di gestori code nel cluster.
- Gestire i gestori code che sono membri del cluster e si trovano su piattaforme e livelli di comando supportati.

La somministrazione non è possibile se:

- Il repository scelto diventa non disponibile. IBM MQ Explorer non passa automaticamente a un repository alternativo.
- Impossibile contattare il repository scelto tramite TCP/IP.
- Il repository scelto è in esecuzione su un gestore code in esecuzione su una piattaforma e a livello di comando non supportato da IBM MQ Explorer.

I membri del cluster che possono essere gestiti possono essere locali o remoti se possono essere contattati utilizzando TCP/IP. Il IBM MQ Explorer si connette ai gestori code locali che sono membri direttamente di un cluster, senza utilizzare una connessione client.

Conversione dati per IBM MQ Explorer

IBM MQ Explorer funziona in CCSID 1208 (UTF-8). Ciò consente a IBM MQ Explorer di visualizzare correttamente i dati provenienti dai gestori code remoti. Se ci si connette direttamente a un gestore code o utilizzando un gestore code intermedio, il IBM MQ Explorer richiede che tutti i messaggi in entrata siano convertiti in CCSID 1208 (UTF-8).

Viene emesso un messaggio di errore se si tenta di stabilire una connessione tra IBM MQ Explorer e un gestore code con un CCSID che IBM MQ Explorer non riconosce.

Le conversioni supportate sono descritte in Conversione codepage.

Windows Utilizzo dell'applicazione IBM MQ Taskbar (soloWindows)

L'applicazione della barra delle applicazioni IBM MQ visualizza un'icona nella barra delle applicazioni Windows sul server. L'icona fornisce lo stato corrente di IBM MQ e un menu da cui è possibile eseguire alcune semplici azioni.

Su Windows, l'icona IBM MQ si trova nella barra delle applicazioni del server e viene sovrapposta con un simbolo di stato codificato a colori, che può avere uno dei seguenti significati:

Verde
Funzionamento corretto; nessun avviso al momento

Blu
Indeterminato; IBM MQ è in fase di avvio o di arresto

Giallo
Avviso; uno o più servizi sono in errore o hanno già avuto esito negativo

Per visualizzare il menù, fare clic con il tasto destro del mouse su IBM MQ . Dal menu è possibile eseguire le seguenti operazioni:

- Fare clic su **Apri** per aprire il Controllo segnalazioni IBM MQ .
- Fare clic su **Esci** per uscire dall'applicazione della barra delle attività IBM MQ .
- Fare clic su **IBM MQ Explorer** per avviare IBM MQ Explorer.
- Fare clic su **Arresta IBM MQ** per arrestare IBM MQ.
- Fare clic su **Informazioni su IBM MQ** per visualizzare le informazioni su IBM MQ Controllo segnalazioni.

Windows L'applicazione di monitoraggio avvisi IBM MQ (solo Windows)

Il controllo degli avvisi IBM MQ è uno strumento di rilevamento degli errori che identifica e registra i problemi con IBM MQ su una macchina locale.

Il monitoraggio avvisi visualizza le informazioni sullo stato corrente dell'installazione locale di un server IBM MQ . Inoltre, monitora Windows ACPI (Advanced Configuration and Power Interface) e garantisce l'applicazione delle impostazioni ACPI.

Dal monitoraggio avvisi IBM MQ , è possibile:

- Accedi direttamente a IBM MQ Explorer
- Visualizza le informazioni relative a tutti gli avvisi in sospeso
- Arrestare il servizio IBM MQ sulla macchina locale
- Instradare i messaggi di avviso sulla rete a un account utente configurabile o a una workstation o a un server Windows

Utilizzo degli oggetti IBM MQ locali

È possibile gestire oggetti IBM MQ locali per supportare programmi applicativi che utilizzano MQI (Message Queue Interface).

Informazioni su questa attività

In questo contesto, amministrazione locale significa creare, visualizzare, modificare, copiare ed eliminare oggetti IBM MQ .

In aggiunta agli approcci descritti in questa sezione, è possibile utilizzare IBM MQ Explorer per gestire gli oggetti IBM MQ locali. Per ulteriori informazioni, consultare “Amministrazione mediante IBM MQ Explorer” a pagina 120.

Procedura

- Utilizzare le informazioni riportate nei seguenti argomenti per semplificare la gestione degli oggetti IBM MQ locali.
  - Programmi applicativi che utilizzano MQI
  - “Amministrazione di IBM MQ utilizzando i comandi MQSC” a pagina 12
  - “Visualizzazione e modifica degli attributi del gestore code” a pagina 137
  - “Gestione delle code locali” a pagina 140
  - “Gestione delle code alias” a pagina 153
  - “Utilizzo delle code modello” a pagina 155
  - “Gestione dei servizi” a pagina 184
  - “Gestione degli oggetti per il trigger” a pagina 192

Uso dei gestori code

È possibile utilizzare i comandi di controllo per avviare e arrestare un gestore code. È possibile utilizzare i comandi MQSC per la visualizzazione o la modifica degli attributi del gestore code.

Attività correlate

Creazione di gestori code su più piattaforme

Multi Avvio di un gestore code

Quando si crea un gestore code, è necessario avviarlo per consentirgli di elaborare comandi o chiamate MQI.

Informazioni su questa attività

È possibile avviare un gestore code utilizzando il comando **strmqm** . Per una descrizione del comando **strmqm** e delle relative opzioni, vedere strmqm.

Linux Windows In alternativa, sui sistemi Windows e Linux (x86 e x86-64), è possibile avviare un gestore code utilizzando IBM MQ Explorer.

Windows Su Windows , è possibile avviare automaticamente un gestore code quando il sistema inizia a utilizzare IBM MQ Explorer. Per ulteriori informazioni, consultare “Amministrazione mediante IBM MQ Explorer” a pagina 120.

Procedura

- Per avviare un gestore code utilizzando il comando **strmqm** , immettere il comando seguito dal nome del gestore code che si desidera avviare.

  Ad esempio, per avviare un gestore code denominato QMB, immettere il seguente comando:

  ```
  strmqm QMB
  ```

  Nota: È necessario utilizzare il comando **strmqm** dall'installazione associata al gestore code che si sta utilizzando. È possibile scoprire a quale installazione è associato un gestore code utilizzando il comando `dspmq -o installation` .

  Il comando `strmqm` non restituisce il controllo fino a quando il gestore code non viene avviato ed è pronto ad accettare le richieste di connessione.

- Linux Windows

  Per avviare un gestore code utilizzando IBM MQ Explorer, completare la seguente procedura:

  a) Apri IBM MQ Explorer.

  b) Nella vista Navigator , selezionare il gestore code.

  c) Fare clic su **Avvia**.

Risultati

Il gestore code viene avviato.

Se l'avvio del gestore code impiega più di pochi secondi, IBM MQ emette messaggi informativi che descrivono in modo intermittente l'avanzamento dell'avvio.

Multi Arresto di un gestore code

È possibile utilizzare il comando **endmqm** per interrompere un gestore code. Questo comando fornisce quattro modi per arrestare un gestore code: un arresto controllato o inattivo, un arresto immediato, un arresto preventivo e un arresto di attesa. In alternativa, su Windows e Linux, è possibile arrestare un gestore code utilizzando IBM MQ Explorer.

Informazioni su questa attività

Esistono quattro modi per arrestare un singolo gestore code dell'istanza con il comando **endmqm** :

Chiusura controllata (disattivata)
Per impostazione predefinita, il comando **endmqm** esegue un arresto inattivo del gestore code specificato. Una chiusura sospesa attende che tutte le applicazioni connesse si siano disconnesse, pertanto il completamento potrebbe richiedere del tempo.

arresto immediato
Per un arresto immediato, tutte le chiamate MQI correnti possono essere completate, ma le nuove chiamate non riescono. Questo tipo di arresto non attende la disconnessione delle applicazioni dal gestore code.

arresto preventivo
Il gestore code viene arrestato immediatamente. Utilizzare questo tipo di arresto solo in circostanze eccezionali, ad esempio quando un gestore code non si arresta come risultato di un normale comando **endmqm** .

Attendi arresto
Questo tipo di arresto è equivalente a un arresto controllato, ma il controllo viene restituito solo dopo l'arresto del gestore code.

Il comando **endmqm** arresta tutte le istanze di un gestore code a più istanze nello stesso modo in cui arresta un gestore code a istanza singola. È possibile emettere il comando **endmqm** sull'istanza attiva o su una delle istanze in standby di un gestore code a più istanze. Tuttavia, è necessario immettere **endmqm** sull'istanza attiva per terminare il gestore code.

È possibile terminare il gestore code entro un tempo di destinazione di un numero di secondi specificato, con o senza interrompere le attività di manutenzione del gestore code non essenziali, consultare “Chiusura di un gestore code entro un'ora di destinazione” a pagina 133.

Attenzione:

- I messaggi persistenti persisteranno indipendentemente dal tipo di arresto utilizzato (inclusi i processi IBM MQ che terminano manualmente), mentre non è possibile garantire che i messaggi non persistenti sopravvivano a qualsiasi tipo di arresto.

  Specificando la proprietà della coda NPMCLASS (HIGH) si salvano i messaggi non persistenti su una base ottimale. L'utilizzo di processi **endmqm -t**, **endmqm -tp**, **endmqm -p**o la chiusura manuale di IBM MQ riduce le possibilità che i messaggi NPMCLASS (HIGH) sopravvivano a un ciclo di arresto o riavvio IBM MQ rispetto a **endmqm -w** o **endmqm -i**
- Il tempo combinato per terminare e riavviare il gestore code può essere più lungo a causa dell'utilizzo di un metodo di arresto più brusco, in particolare quando si utilizzano le opzioni **-p** e **-tp** .

  Se il gestore code deve ricorrere all'arresto dei processi IBM MQ per terminare il gestore code, è probabile che sia necessaria una maggiore riconciliazione dello stato del gestore code quando il gestore code viene riavviato.

Per una descrizione dettagliata del comando **endmqm** e delle relative opzioni, vedere endmqm.

Suggerimento: I problemi con la chiusura di un gestore code sono spesso causati dalle applicazioni. Ad esempio, quando le applicazioni:

- Non controllare correttamente i codici di ritorno MQI
- Non richiedere la notifica di un quiesce
- Terminare senza disconnettersi dal gestore code (emettendo una chiamata `MQDISC`)

Se si verifica un problema quando si tenta di arrestare il gestore code, è possibile interrompere il comando **endmqm** utilizzando Ctrl-C. È quindi possibile immettere un altro comando **endmqm** , ma questa volta con un parametro che specifica il tipo di arresto richiesto.

Come alternativa all'utilizzo del comando **endmqm** , su Windows e Linux, è possibile arrestare un gestore code utilizzando IBM MQ Explorer per eseguire un arresto controllato o immediato.

Procedura

- Per arrestare il gestore code utilizzando il comando **endmqm** , immettere il comando seguito dal parametro appropriato, se richiesto, e il nome del gestore code che si desidera arrestare.

Nota: È necessario utilizzare il comando **endmqm** dall'installazione associata al gestore code che si sta utilizzando. Per individuare a quale installazione è associato un gestore code, utilizzare il comando **dspmq** :

```
dspmq -o installation
```

- Per eseguire un arresto controllato (inattivo), immettere il comando **endmqm** come mostrato nel seguente esempio, che arresta un gestore code denominato QMB:

  ```
  endmqm QMB
  ```

  In alternativa, immettere il comando **endmqm** con il parametro **-c** , come mostrato nel seguente esempio, equivale a un comando `endmqm QMB` .

  ```
  endmqm -c QMB
  ```

  In entrambi i casi, il controllo viene restituito immediatamente e non si riceve alcuna notifica quando il gestore code è stato arrestato. Se si desidera che il comando attenda l'arresto di tutte le applicazioni e che il gestore code sia terminato prima di restituire il controllo all'utente, utilizzare il parametro **-w** come mostrato nel seguente esempio.

  ```
  endmqm -w QMB
  ```

- Per eseguire un arresto immediato, immettere il comando **endmqm** con il parametro **-i** come mostrato nel seguente esempio:

  ```
  endmqm -i QMB
  ```

- Per eseguire un arresto preventivo, immettere il comando **endmqm** con il parametro **-p** come mostrato nel seguente esempio:

  ```
  endmqm -p QMB
  ```

  Attenzione: Un arresto preventivo può avere conseguenze imprevedibili per le applicazioni connesse. Non utilizzare questa opzione a meno che tutti gli altri tentativi di arrestare il gestore code utilizzando un normale comando **endmqm** non abbiano avuto esito negativo.
  ALW Se l'arresto preventivo non funziona, provare “Arresto manuale di un gestore code” a pagina 134 .

- Per richiedere la riconnessione client automatica, immettere il comando **endmqm** con il parametro `-r` . Questo parametro ha l'effetto di ristabilire la connessione dei client ad altri gestori code nel relativo gruppo di gestori code.

  Nota: L'arresto di un gestore code utilizzando il comando **endmqm** predefinito non attiva la riconnessione client automatica.

- Per trasferire a un'istanza in standby di un gestore code a più istanze dopo aver arrestato l'istanza attiva, immettere il comando **endmqm** con il parametro `-s` sull'istanza attiva del gestore code a più istanze.
- Per terminare l'istanza in standby di un gestore code a più istanze e lasciare in esecuzione l'istanza attiva, immettere il comando **endmqm** con il parametro `-x` nell'istanza in standby del gestore code a più istanze.
- Linux Windows

  In Windows e Linux, per arrestare il gestore code utilizzando IBM MQ Explorer, completare la seguente procedura:

  a) Apri IBM MQ Explorer.

  b) Selezionare il gestore code dalla vista Navigator .

c) Fare clic su **Arresta**.
   Viene visualizzato il pannello **Termina gestore code** .
d) Selezionare **Controllato**o **Immediato**.
e) Fare clic su **OK**.
   Il gestore code viene arrestato.

Attività correlate
Applicazione degli aggiornamenti del livello di manutenzione ai gestori code a più istanze su AIX
Applicazione degli aggiornamenti del livello di manutenzione ai gestori code a più istanze su Linux
Applicazione degli aggiornamenti del livello di manutenzione ai gestori code a più istanze su Windows
Riferimenti correlati
endmqm (fine gestore code)

Chiusura di un gestore code entro un'ora di destinazione

È possibile terminare il gestore code entro un tempo di destinazione di un numero di secondi specificato, con o senza interrompere le attività di manutenzione del gestore code non essenziali.

Esistono due metodi per specificare un'ora di destinazione quando si utilizza il comando **endmqm** . L'opzione **-t** consente il completamento di tutte le attività di manutenzione del gestore code, che potrebbero prolungare la fase di chiusura del gestore code. L'opzione **-tp** interrompe le attività di manutenzione del gestore code non essenziali, se necessario, per rispettare l'ora di destinazione specificata.

Le attività di manutenzione non essenziali includono: la compressione dei file di coda e la persistenza dei messaggi NPMCLASS (HIGH). Nel resto di questa pagina, viene utilizzata la parola "pulizia".

A seconda dei modelli di uso dell'applicazione, la compattazione dei file di coda può richiedere molto tempo, quindi se l'obiettivo primario è quello di terminare rapidamente il gestore code, utilizzare l'opzione **-tp** .

Quando si specifica un'ora di destinazione, il tipo di arresto **-w**, **-i**o **-p** indica il tipo di arresto iniziale.

Nota: Un arresto di `immediate` è ancora ordinato, differendo da un arresto di `controlled` principalmente nel modo in cui le applicazioni in esecuzione vengono disattivate. Un arresto `immediate` esegue ancora la manutenzione. Un arresto limitato nel tempo chiude queste azioni quando interferiscono con il raggiungimento dell'orario di destinazione.

Il gestore code esegue l'escalation del tipo di arresto come necessario, nel tentativo di soddisfare l'ora di destinazione. Ad esempio:

- Una destinazione **-t** di 10 secondi a partire da **-w** potrebbe essere sette secondi di quiesce, due secondi di arresto immediato del gestore code, incluse le operazioni di pulizia, quindi l'arresto immediato senza ulteriori operazioni di pulizia:

```
endmqm  -w -t 10 queue_manager
```

- Una destinazione **-tp** di 10 secondi potrebbe essere sette secondi di sospensione, due secondi di arresto immediato del gestore code, inclusa la manutenzione, un secondo di arresto immediato senza ulteriore manutenzione, quindi avviare la chiusura dei processi IBM MQ :

```
endmqm -c -tp 10 queue_manager
```

- Una destinazione **-tp** di due secondi in **-i** potrebbe essere un secondo di arresto immediato del gestore code, inclusa la manutenzione, un secondo di arresto immediato senza ulteriore manutenzione, quindi avviare la chiusura dei processi IBM MQ :

```
endmqm -i -tp 2 queue_manager
```

- Una destinazione di un secondo in **-w** potrebbe essere 0.1 secondi in `wait`, ad esempio, per un tempo sufficiente a inviare i codici di ritorno IBM MQ alle applicazioni connesse, 0.9 secondi per l'arresto

immediato del gestore code, incluso il servizio di pulizia, quindi l'arresto immediato senza ulteriori operazioni di manutenzione, quindi l'avvio della chiusura dei processi IBM MQ .

Riferimenti correlati

endmqm (fine gestore code)

ALW Arresto manuale di un gestore code

Se i metodi standard per l'arresto e la rimozione di un gestore code hanno esito negativo, è possibile provare ad arrestare manualmente il gestore code.

Informazioni su questa attività

Il modo standard per arrestare i gestori code consiste nell'utilizzare il comando **endmqm** , come descritto in “Arresto di un gestore code” a pagina 130. Se non è possibile arrestare un gestore code nel modo standard, è possibile provare ad arrestare manualmente un gestore code. Il modo in cui si fa ciò dipende dalla piattaforma che si sta utilizzando.

Procedura

- Windows
  Per arrestare un gestore code su Windows, consultare “Arresto manuale di un gestore code su Windows” a pagina 134.
- Linux AIX
  Per arrestare un gestore code su AIX o Linux, fare riferimento a “Arresto manuale di un gestore code su AIX and Linux” a pagina 135.

Attività correlate

Creazione e gestione di gestori code su più piattaforme

Riferimenti correlati

endmqm

Windows *Arresto manuale di un gestore code su Windows*

Se non è possibile arrestare un gestore code su Windows utilizzando il comando **endmqm** , è possibile provare ad arrestare manualmente il gestore code arrestando i processi in esecuzione e arrestando il servizio IBM MQ .

Informazioni su questa attività

Suggerimento: Windows Task Manager e il comando **tasklist** forniscono informazioni limitate sulle attività. Per ulteriori informazioni utili per determinare quali processi sono correlati a uno specifico gestore code, utilizzare uno strumento come *Process Explorer* (`procexp.exe`), disponibile per il download dal sito Web Microsoft all'indirizzo http://www.microsoft.com.

Per arrestare un gestore code su Windows, completare la seguente procedura.

Procedura

1. Elencare i nomi (ID) dei processi in esecuzione, utilizzando Windows Task Manager.
2. Terminare i processi utilizzando Windows Task Manager o il comando **taskkill** , nel seguente ordine (se sono in esecuzione):

Tabella 7. Processi Windows da arrestare se in esecuzione

Nome processo	Descrizione
AMQZMUC0	Gestore processi critici

Tabella 7. Processi Windows da arrestare se in esecuzione (Continua)	
Nome processo	**Descrizione**
AMQZXMA0	Controllore di Esecuzione
AMQZFUMA	processo OAM
AMQZLAA0	Agent LQM
AMQZLSA0	Agent LQM
AMQZMUF0	Gestore utilità
AMQZMGR0	Controller processo
AMQZMUR0	Gestore processi riavviabile
AMQFQPUB	Processo di pubblicazione - sottoscrizione
AMQFCXBA	Processo di lavoro broker
AMQRMPPA	processo di pooling del processo
AMQCRSTA	Processo di lavoro del responder non sottoposto a thread
AMQCRS6B	Canale ricevente LU62 e connessione client
AMQRRMFA	Il processo del repository (per i cluster)
AMQPCSEA	Il server comandi
RUNMQTRM	Richiama un controllo trigger per un server
RUNMQDLQ	Richiama gestore code di messaggi non recapitabili
RUNMQCHI	Il processo iniziatore del canale
RUNMQLSR	Il processo del listener del canale
VNXMQS	Server di memoria condivisa

3. Arrestare il servizio IBM MQ da **Strumenti di gestione** > **Servizi** nel Windows Pannello di controllo.
4. Se sono stati tentati tutti i metodi e il gestore code non è stato arrestato, riavviare il sistema.

Linux AIX *Arresto manuale di un gestore code su AIX and Linux*

Se non è possibile arrestare un gestore code su AIX o Linux utilizzando il comando **endmqm** , è possibile provare ad arrestare manualmente il gestore code arrestando tutti i processi in esecuzione e arrestando il servizio IBM MQ .

Informazioni su questa attività

Per arrestare un gestore code su AIX o Linux, completare la seguente procedura.

Se si arresta il gestore code manualmente, FFST potrebbe essere utilizzato e i file FDC posizionati in `/var/mqm/errors`. Non deve essere considerato come un difetto nel gestore code.

Il gestore code verrà riavviato normalmente, anche dopo averlo arrestato utilizzando questo metodo di arresto manuale.

Procedura

1. Individuare i PID (process identifier) dei programmi del gestore code ancora in esecuzione utilizzando il comando **ps** .

Ad esempio, se il gestore code è denominato QMNAME, utilizzare il seguente comando:

```
ps -ef | grep QMNAME
```

2. Terminare tutti i processi del gestore code ancora in esecuzione utilizzando il comando **kill** , specificando i PID rilevati utilizzando il comando **ps** .

   Per terminare un processo, utilizzare **kill -KILL <pid>** o il comando **kill -9 <pid>** equivalente.

   Devi lavorare attraverso i PID che vuoi uccidere, uno ad uno, immettendo quel comando ogni volta.

   Importante: Se si utilizza un segnale diverso da **9(SIGKILL)** , il processo probabilmente non si arresterà e si otterranno risultati imprevedibili.

   Terminare i processi nel seguente ordine:

Tabella 8. Processi AIX and Linux da arrestare se in esecuzione

Nome processo	**Descrizione**
amqzmuc0	Gestore processi critici
amqzxma0	Controllore di Esecuzione
amqzfuma	processo OAM
amqzlaa0	Agent LQM
amqzlsa0	Agent LQM
amqzmuf0	Gestore utilità
amqzmur0	Gestore processi riavviabile
amqzmgr0	Controller processo
amqfqpub	Processo di pubblicazione - sottoscrizione
amqfcxba	Processo di lavoro broker
amqrmppa	processo di pooling del processo
amqcrsta	Processo di lavoro del responder non sottoposto a thread
amqcrs6b	Canale ricevente LU62 e connessione client
amqrrmfa	Il processo del repository (per i cluster)
amqpcsea	Il server comandi
runmqtrm	Richiama un controllo trigger per un server
runmqdlq	Richiama gestore code di messaggi non recapitabili
runmqchi	Il processo iniziatore del canale
runmqlsr	Il processo del listener del canale

Attività correlate

“Arresto di un gestore code” a pagina 130
È possibile utilizzare il comando **endmqm** per interrompere un gestore code. Questo comando fornisce quattro modi per arrestare un gestore code: un arresto controllato o inattivo, un arresto immediato, un arresto preventivo e un arresto di attesa. In alternativa, su Windows e Linux, è possibile arrestare un gestore code utilizzando IBM MQ Explorer.

Multi Riavvio di un gestore code

È possibile utilizzare il comando **strmqm** per riavviare un gestore code oppure, su sistemi Windows e Linux x86-64 , è possibile riavviare un gestore code da IBM MQ Explorer.

Informazioni su questa attività

È possibile riavviare un gestore code utilizzando il comando **strmqm** . Per una descrizione del comando **strmqm** e delle relative opzioni, vedere strmqm.

Linux Windows Sui sistemi Windows e Linux x86-64 , è possibile riavviare un gestore code utilizzando IBM MQ Explorer nello stesso modo in cui si avvia un gestore code.

Procedura

- Per riavviare un gestore code utilizzando il comando **strmqm** , immettere il comando seguito dal nome del gestore code che si desidera riavviare.

  Ad esempio, per avviare un gestore code denominato `strmqm saturn.queue.manager`, immettere il seguente comando:

  ```
  strmqm saturn.queue.manager
  ```

- Linux Windows

  Per avviare un gestore code utilizzando IBM MQ Explorer, completare la seguente procedura:

  a) Apri IBM MQ Explorer.

  b) Nella vista Navigator , selezionare il gestore code.

  c) Fare clic su **Avvia**.

Risultati

Il gestore code viene riavviato.

Se il riavvio del gestore code impiega più di pochi secondi, IBM MQ emette messaggi informativi che descrivono in modo intermittente l'avanzamento dell'avvio.

Visualizzazione e modifica degli attributi del gestore code

Utilizzare il comando **DISPLAY QMGR** MQ per visualizzare i parametri del gestore code per un gestore code. Utilizzare il comando **ALTER QMGR** MQSC per modificare i parametri del gestore code per un gestore code locale.

Prima di iniziare

Nota: I passi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Procedura

- Per visualizzare gli attributi del gestore code specificato sul comando **runmqsc** , utilizzare il comando MQSC **DISPLAY QMGR** :

  ```
  DISPLAY QMGR
  ```

  Il seguente esempio mostra l'output tipico di questo comando:

  ```
  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 è un attributo del gestore code di sola lettura.

Il parametro **ALL** è il valore predefinito sul comando **DISPLAY QMGR** . Visualizza tutti gli attributi del gestore code. In particolare, l'output indica il nome del gestore code predefinito, il nome della coda di messaggi non recapitabili e il nome della coda comandi.

È possibile confermare l'esistenza di queste code immettendo il seguente comando:

```
DISPLAY QUEUE (SYSTEM.*)
```

Visualizza un elenco di code che corrispondono alla radice SYSTEM.*. Le parentesi sono obbligatorie.

- Per modificare gli attributi del gestore code specificato sul comando **runmqsc** , utilizzare il comando MQSC **ALTER QMGR**, specificando gli attributi e valori che si desidera modificare.

  Ad esempio, utilizzare i comandi seguenti per modificare gli attributi di `jupiter.queue.manager`:

  ```
  runmqsc jupiter.queue.manager
  ALTER QMGR DEADQ (ANOTHERDLQ) INHIBTEV (ENABLED)
  ```

  Il comando **ALTER QMGR** modifica la coda di messaggi non recapitabili utilizzata e abilita gli eventi di inibizione.

I parametri non specificati nel comando **ALTER QMGR** fanno sì che i valori esistenti per tali parametri non vengano modificati.

Attività correlate

Creazione di gestori code su più piattaforme

Riferimenti correlati

Attributi per il gestore code

runmqsc (esecuzione comandi MQSC)

VISUALIZZAZIONE QMGR

Gestore code ALTER

Multi Eliminazione di un gestore code

È possibile eliminare il gestore code utilizzando il comando di controllo **dltmqm** . In alternativa, su sistemi Windows e Linux , è possibile utilizzare IBM MQ Explorer per eliminare un gestore code.

Prima di iniziare

Attenzione:

- L'eliminazione di un gestore code è un passo drastico, poiché si eliminano anche le risorse associate al gestore code, incluse tutte le code e i relativi messaggi e tutte le definizioni di oggetti. Se si utilizza il comando di controllo **dltmqm** , non viene visualizzata alcuna richiesta che consente di cambiare idea; quando si preme il tasto Invio, tutte le risorse associate vengono perse.
- Windows Su Windows, l'eliminazione di un gestore code rimuove anche il gestore code dall'elenco di avvio automatico (descritto in “Avvio di un gestore code” a pagina 129). Quando il comando è stato completato, viene visualizzato un messaggio `IBM MQ queue manager ending` ; non viene indicato che il gestore code è stato eliminato.
- L'eliminazione di un gestore code del cluster non lo rimuove dal cluster. Per ulteriori informazioni, consultare le note di utilizzo in dltmqm.

Informazioni su questa attività

È possibile eliminare un gestore code utilizzando il comando di controllo **dltmqm** . Per una descrizione del comando **dltmqm** e delle relative opzioni, consultare dltmqm. Assicurarsi che solo gli amministratori attendibili abbiano l'autorità per utilizzare questo comando. (Per informazioni sulla sicurezza, vedere Impostazione della sicurezza su AIX, Linux, and Windows.)

Linux Windows In alternativa, su sistemi Windows e Linux (piattaformex86 e x86-64), è possibile eliminare un gestore code utilizzando IBM MQ Explorer.

Procedura

- Per eliminare un gestore code utilizzando il comando **dltmqm** , completare la seguente procedura:
  a) Chiudere il gestore code.
  b) Emetti il seguente comando:

  ```
  dltmqm QMB
  ```

  Nota: È necessario utilizzare il comando **dltmqm** dall'installazione associata al gestore code che si sta utilizzando. È possibile scoprire a quale installazione è associato un gestore code utilizzando il comando `dspmq -o installation` .
- Linux Windows
  Per eliminare un gestore code utilizzando IBM MQ Explorer, completare la seguente procedura:

a) Apri IBM MQ Explorer.

b) Nella vista Navigator , selezionare il gestore code.

c) Se il gestore code non è arrestato, arrestarlo.

Per arrestare il gestore code, fare clic con il pulsante destro del mouse su di esso e fare clic su **Arresta**.

d) Eliminare il gestore code.

Per eliminare il gestore code, fare clic con il tasto destro del mouse su di esso e fare clic su **Elimina**.

Risultati

Il gestore code è stato eliminato.

Arresto dei canali MQI

Quando si immette un comando STOP CHANNEL su un canale di connessione server, è possibile selezionare il metodo da utilizzare per arrestare il canale di connessione client. Ciò significa che un canale client che emette una chiamata di attesa MQGET può essere controllato ed è possibile decidere come e quando arrestare il canale.

Il comando STOP CHANNEL può essere emesso con tre modi, indicando come deve essere arrestato il canale:

Tempo di sospensione
Arresta il canale dopo che sono stati elaborati tutti i messaggi correnti.

Se la condivisione delle conversazioni è abilitata, IBM MQ MQI client viene a conoscenza della richiesta di arresto in modo tempestivo; questo tempo dipende dalla velocità della rete. L'applicazione client viene a conoscenza della richiesta di arresto come risultato dell'emissione di una chiamata successiva a IBM MQ.

Forza
Arresta immediatamente il canale.

Termina
Arresta immediatamente il canale. Se il canale è in esecuzione come un processo, può terminare il processo del canale o se il canale è in esecuzione come un thread, il suo thread.

Questo è un processo a più fasi. Se viene utilizzata la modalità terminate, viene effettuato un tentativo di arrestare il canale di connessione server, prima con la modalità quiesce, quindi con la modalità force e, se necessario, con la modalità terminate. Il client può ricevere codici di ritorno differenti durante le diverse fasi di terminazione. Se il processo o il thread viene terminato, il client riceve un errore di comunicazione.

I codici di ritorno restituiti all'applicazione variano in base alla chiamata MQI emessa e al comando STOP CHANNEL emesso. Il client riceverà un codice di ritorno MQRC_CONNECTION_QUIESCING o MQRC_CONNECTION_BROKEN. Se un client rileva MQRC_CONNECTION_QUIESCING, deve tentare di completare la transazione corrente e terminare. Ciò non è possibile con MQRC_CONNECTION_BROKEN. Se il client non completa la transazione e termina abbastanza velocemente, otterrà CONNECTION_BROKEN dopo pochi secondi. Un comando STOP CHANNEL con MODE (FORCE) o MODE (TERMINATE) è più probabile che abbia come risultato un CONNECTION_BROKEN rispetto a MODE (QUIESCE).

Concetti correlati

Canali

Gestione delle code locali

Questa sezione contiene esempi di alcuni comandi MQSC che è possibile utilizzare per gestire code locali, modello e alias.

Consultare Comandi MQSC per informazioni dettagliate su questi comandi.

Riferimenti correlati

Limitazioni di denominazione per code

Limitazioni di denominazione per altri oggetti

Definizione di una coda locale con DEFINE QLOCAL

Per un'applicazione, il gestore code locale è il gestore code a cui è connessa l'applicazione. Le code gestite dal gestore code locale vengono dette locali per tale gestore code. Utilizzare il comando MQSC **DEFINE QLOCAL** per creare una coda locale.

Prima di iniziare

Nota: I passi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Informazioni su questa attività

Utilizzare il comando MQSC **DEFINE QLOCAL** per creare una coda locale. È anche possibile utilizzare il valore predefinito definito nella definizione della coda locale predefinita oppure è possibile modificare le caratteristiche della coda da quelle della coda locale predefinita.

Nota: La coda locale predefinita è denominata SYSTEM.DEFAULT.LOCAL.QUEUE e viene creato sull'installazione del sistema.

Procedura

- Per creare una coda locale, immettere il comando **DEFINE QLOCAL** come mostrato nel seguente esempio.

  In questo esempio, il comando **DEFINE QLOCAL** definisce una coda denominata ORANGE.LOCAL.QUEUE con queste caratteristiche:

  – È abilitato per le ricezioni, abilitato per gli inserimenti e opera su una base di ordine di priorità.

  – Si tratta di una coda *normale* ; non è una coda di iniziazione o di trasmissione e non genera messaggi trigger.

  – La grandezza massima della coda è 5000 messaggi; la lunghezza massima del messaggio è 4194304 byte.

```
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)
```

  Note:

  1. Ad eccezione del valore per la descrizione, tutti i valori di attributo mostrati nell'esempio sono i valori predefiniti. Questi esempi sono inclusi a scopo illustrativo. È possibile ometterli se si è certi che i valori predefiniti sono quelli desiderati o non sono stati modificati. Consultare anche “Visualizzazione degli attributi dell'oggetto predefiniti con DISPLAY QUEUE” a pagina 142.
  2. **USAGE(NORMAL)** indica che questa coda non è una coda di trasmissione.
  3. Se si dispone già di una coda locale sullo stesso gestore code con il nome ORANGE.LOCAL.QUEUE, questo comando ha esito negativo. Utilizzare l'attributo **REPLACE** se si desidera sovrascrivere la definizione esistente di una coda, ma consultare anche “Modifica degli attributi della coda locale con ALTER QLOCAL o DEFINE QLOCAL” a pagina 143.

Riferimenti correlati
DEFINE QLOCAL

Visualizzazione degli attributi dell'oggetto predefiniti con DISPLAY QUEUE

È possibile utilizzare il comando **DISPLAY QUEUE** MQSC per visualizzare gli attributi che sono stati presi dall'oggetto predefinito quando è stato definito un oggetto IBM MQ .

Prima di iniziare

Nota: I passi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Informazioni su questa attività

Quando si definisce un oggetto IBM MQ , vengono utilizzati tutti gli attributi non specificati dall'oggetto predefinito. Ad esempio, quando si definisce una coda locale, la coda eredita tutti gli attributi omessi nella definizione dalla coda locale predefinita, denominata SYSTEM.DEFAULT.LOCAL.QUEUE. È possibile utilizzare il comando **DISPLAY QUEUE** per visualizzare esattamente questi attributi.

Procedura

- Per visualizzare gli attributi oggetto predefiniti per una coda locale, utilizzare il seguente comando:

  ```
  DISPLAY QUEUE (SYSTEM.DEFAULT.LOCAL.QUEUE)
  ```

  La sintassi del comando **DISPLAY** è diversa da quella del comando **DEFINE** corrispondente. Sul comando **DISPLAY** è possibile fornire solo il nome della coda, mentre sul comando **DEFINE** è necessario specificare il tipo di coda, ovvero QLOCAL, QALIAS, QMODEL o QREMOTE.

  È possibile visualizzare in modo selettivo gli attributi specificandoli singolarmente. Ad esempio:

  ```
  DISPLAY QUEUE (ORANGE.LOCAL.QUEUE) +
  MAXDEPTH +
  MAXMSGL +
  CURDEPTH;
  ```

  Questo comando visualizza i tre attributi specificati nel modo seguente:

  ```
  AMQ8409: Display Queue details.
  QUEUE(ORANGE.LOCAL.QUEUE)        TYPE(QLOCAL)
  CURDEPTH(0)                MAXDEPTH(5000)
  MAXMSGL(4194304)
  ```

  CURDEPTH è la profondità della coda corrente, ossia il numero di messaggi sulla coda. Questo è un attributo utile da visualizzare, perché monitorando la profondità della coda è possibile assicurarsi che la coda non diventi piena.

Riferimenti correlati
VISUALIZZAZIONE CODA
code DEFINE

Copia di una definizione di coda locale con DEFINE QLOCAL

È possibile copiare una definizione di coda utilizzando l'attributo **LIKE** nel comando MQSC **DEFINE QLOCAL** .

Prima di iniziare

Nota: I passi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Informazioni su questa attività

È possibile utilizzare il comando **DEFINE** con l'attributo **LIKE** per creare una coda con gli stessi attributi della coda specificata, piuttosto che quelli della coda locale predefinita del sistema. È anche possibile utilizzare questo formato del comando **DEFINE** per copiare una definizione di coda, ma sostituire una o più modifiche agli attributi dell'originale.

Note:

1. Quando si utilizza l'attributo **LIKE** su un comando **DEFINE** , si stanno copiando solo gli attributi della coda. Non si stanno copiando i messaggi sulla coda.
2. Se si definisce una coda locale, senza specificare **LIKE**, è uguale a:

```
DEFINE LIKE(SYSTEM.DEFAULT.LOCAL.QUEUE)
```

Procedura

- Per creare una coda con gli stessi attributi della coda specificata, piuttosto che quelli della coda locale predefinita di sistema, immettere il comando **DEFINE** come mostrato nel seguente esempio.

  Immettere il nome della coda da copiare esattamente come è stata immessa al momento della creazione della coda. Se il nome contiene caratteri minuscoli, racchiuderlo tra virgolette singole.

  Questo esempio crea una coda con gli stessi attributi della coda ORANGE.LOCAL.QUEUE, piuttosto che quelli della coda locale predefinita del sistema:

```
DEFINE QLOCAL (MAGENTA.QUEUE) +
LIKE (ORANGE.LOCAL.QUEUE)
```

- Per copiare una definizione di coda, ma sostituire una o più modifiche agli attributi dell'originale, immettere il comando **DEFINE** come mostrato nel seguente esempio.

  Questo comando copia gli attributi della coda ORANGE.LOCAL.QUEUE alla coda THIRD.QUEUE, ma specifica che la lunghezza massima del messaggio nella nuova coda deve essere di 1024 byte, invece di 4194304:

```
DEFINE QLOCAL (THIRD.QUEUE) +
LIKE (ORANGE.LOCAL.QUEUE) +
MAXMSGL(1024);
```

Riferimenti correlati

code DEFINE

Modifica degli attributi della coda locale con ALTER QLOCAL o DEFINE QLOCAL

È possibile modificare gli attributi della coda in due modi, utilizzando il comando MQSC **ALTER QLOCAL** o **DEFINE QLOCAL** con l'attributo **REPLACE** .

Prima di iniziare

Nota: I passi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Informazioni su questa attività

È possibile utilizzare l'attributo **REPLACE** del comando **ALTER** e **DEFINE** per sostituire una definizione esistente con la nuova definizione specificata. La differenza tra l'utilizzo di **ALTER** e **DEFINE** è che **ALTER** con **REPLACE** non modifica parametri non specificati, ma **DEFINE** con **REPLACE** imposta tutti i parametri.

Procedura

- Per modificare gli attributi della coda, utilizzare il comando **ALTER** o il comando **DEFINE** come mostrato nei seguenti esempi.

  In questi esempi, la lunghezza massima del messaggio sulla coda ORANGE.LOCAL.QUEUE viene ridotto a 10.000 byte.

  – Utilizzando il comando **ALTER** :

  ```
  ALTER QLOCAL (ORANGE.LOCAL.QUEUE) MAXMSGL(10000)
  ```

  Questo comando modifica un singolo attributo, quello della lunghezza massima del messaggio; tutti gli altri attributi rimangono uguali.

  – Utilizzando il comando **DEFINE** con l'opzione **REPLACE** , ad esempio:

  ```
  DEFINE QLOCAL (ORANGE.LOCAL.QUEUE) MAXMSGL(10000) REPLACE
  ```

  Questo comando modifica non solo la lunghezza massima del messaggio, ma anche tutti gli altri attributi, ai quali vengono assegnati valori predefiniti. Quindi, ad esempio, se la coda è stata precedentemente inibita, viene modificata in abilitata all'inserimento poiché l'inserimento abilitato è il valore predefinito, come specificato dalla coda SYSTEM.DEFAULT.LOCAL.QUEUE.

  Se si diminuisce la lunghezza massima del messaggio su una coda esistente, i messaggi esistenti non vengono influenzati. Qualsiasi nuovo messaggio, tuttavia, deve soddisfare i nuovi criteri.

Riferimenti correlati
Code ALTER
MODIFICA QLOCAL
code DEFINE
DEFINE QLOCAL

Cancellazione di una coda locale con CLEAR QLOCAL

È possibile utilizzare il comando **CLEAR QLOCAL** MQSC per cancellare una coda locale.

Prima di iniziare

Nota: I passi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Non è possibile cancellare una coda se:

- Ci sono messaggi senza commit che sono stati inseriti nella coda nel punto di sincronizzazione.
- Un'applicazione ha attualmente la coda aperta.

Informazioni su questa attività

Se si desidera cancellare una coda locale utilizzando il comando **CLEAR QLOCAL** , il nome della coda deve essere definito sul gestore code locale.

Nota: Non vi è alcuna richiesta che consente di cambiare idea; una volta premuto il tasto Invio, i messaggi vengono persi.

Procedura

Per cancellare i messaggi da una coda locale, utilizzare **CLEAR QLOCAL** come mostrato nel seguente esempio.

In questo esempio, tutti i messaggi vengono eliminati da una coda locale denominata MAGENTA.QUEUE:

```
CLEAR QLOCAL (MAGENTA.QUEUE)
```

Riferimenti correlati
CANCELLA QLOCAL

Eliminazione di una coda locale con DELETE QLOCAL

È possibile utilizzare il comando MQSC **DELETE QLOCAL** per eliminare una coda locale.

Prima di iniziare

Nota: I passi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Una coda non può essere eliminata se contiene messaggi di cui non è stato eseguito il commit.

Se una coda ha uno o più messaggi di cui è stato eseguito il commit e nessun messaggio di cui non è stato eseguito il commit, può essere eliminata solo se si specifica l'opzione **PURGE** . L'eliminazione viene quindi eseguita anche se ci sono messaggi di cui è stato eseguito il commit sulla coda denominata e anche questi messaggi vengono eliminati.

Specificando **NOPURGE** invece di **PURGE** si garantisce che la coda non venga eliminata se contiene messaggi di cui è stato eseguito il commit.

Procedura

- Per eliminare una coda locale, utilizzare il comando **DELETE QLOCAL** come mostrato nel seguente esempio.

  Questo esempio elimina la coda PINK.QUEUE se nella coda non sono presenti messaggi di cui è stato eseguito il commit:

  ```
  DELETE QLOCAL (PINK.QUEUE) NOPURGE
  ```

  Questo esempio elimina la coda PINK.QUEUE anche se sulla coda sono presenti messaggi di cui è stato eseguito il commit:

  ```
  DELETE QLOCAL (PINK.QUEUE) PURGE
  ```

Riferimenti correlati
ELIMINA QLOCALE

Ricerca delle code con il programma di esempio

IBM MQ fornisce un browser della coda di esempio che è possibile utilizzare per esaminare il contenuto dei messaggi su una coda.

Informazioni su questa attività

Il browser viene fornito nei formati di origine ed eseguibile nelle seguenti ubicazioni, dove *MQ_INSTALLATION_PATH* rappresenta la directory di alto livello in cui è installato IBM MQ .

Windows Su Windows, i nomi file e i percorsi per il browser della coda di esempio sono i seguenti:

Origine
MQ_INSTALLATION_PATH\tools\c\samples\

Eseguibile
MQ_INSTALLATION_PATH\tools\c\samples\bin\amqsbcg.exe

Linux AIX In AIX and Linux, i nomi file e i percorsi sono i seguenti:

Origine
MQ_INSTALLATION_PATH/samp/amqsbcg0.c

Eseguibile
MQ_INSTALLATION_PATH/samp/bin/amqsbcg

Procedura

- Per eseguire il programma di esempio, immettere un comando come mostrato nel seguente esempio.

  Il programma di esempio richiede due parametri di immissione, il nome della coda in cui verranno esaminati i messaggi e il gestore code proprietario di tale coda. Ad esempio:

```
amqsbcg SYSTEM.ADMIN.QMGREVENT.tpp01 saturn.queue.manager
```

Risultati

I tipici risultati di questo comando vengono mostrati nel seguente esempio:

```
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
```

Riferimenti correlati

Il programma di esempio Browser

Abilitazione di code grandi

IBM MQ supporta code più grandi di 2 TB.

Windows Sui sistemi Windows , il supporto per file di grandi dimensioni è disponibile senza alcuna ulteriore abilitazione.

Linux AIX Su sistemi AIX and Linux , è necessario abilitare esplicitamente il supporto per file di grandi dimensioni prima di poter creare file di coda di più gigabyte o terabyte. Per informazioni su come eseguire questa operazione, consultare la documentazione del sistema operativo.

Alcuni programmi di utilità, come tar, non sono in grado di gestire file di più gigabyte o terabyte. Prima di attivare il supporto file di grandi dimensioni, consultare la documentazione del sistema operativo per informazioni sulle restrizioni relative ai programmi di utilità utilizzati.

Per informazioni sulla pianificazione della quantità di memoria necessaria per le code, consultare MQ Performance documents per report sulle prestazioni specifici della piattaforma.

È possibile controllare la dimensione dei file della coda utilizzando un nuovo attributo sulle code locali e modello. Per ulteriori informazioni, consultare “Modifica dei file di coda IBM MQ” a pagina 147.

Multi Modifica dei file di coda IBM MQ

È possibile controllare la dimensione dei file della coda utilizzando un attributo sulle code locali e modello. È possibile visualizzare la dimensione corrente di un file di coda e la dimensione massima a cui è attualmente in grado di aumentare (in base alla dimensione del blocco attualmente in uso in tale file), utilizzando due attributi di stato della coda.

Attributo utilizzato per modificare i file della coda

L'attributo sulle code locali e modello è:

MAXFSIZE
Indica la dimensione massima del file di coda utilizzato dalla coda, in megabyte.

È possibile impostare o visualizzare il valore di questo attributo utilizzando i comandi MQSC, IBM MQ Explorero administrative REST API. È anche possibile visualizzare il valore di questo attributo in IBM MQ Console.

Consultare MAXFSIZE e “Modifica della dimensione di un file di coda IBM MQ” a pagina 148 per ulteriori informazioni.

L'equivalente PCF di questo attributo è **MQIA_MAX_Q_FILE_SIZE**. Consultare Modifica, copia e crea coda.

I due attributi sullo stato della coda sono:

CURFSIZE
Visualizza la dimensione corrente del file di coda in megabyte, arrotondato al megabyte più vicino.

È possibile impostare o visualizzare il valore di questo attributo utilizzando i comandi MQSC, IBM MQ Explorero administrative REST API.

Per ulteriori informazioni, consultare CURFSIZE .

L'equivalente PCF di questo attributo è **MQIA_CUR_Q_FILE_SIZE**. Vedere Coda di interrogazione e Coda di interrogazione (risposta).

CURMAXFS
Indica la dimensione massima corrente che può raggiungere il file della coda, arrotondata al megabyte più vicino, data la dimensione del blocco corrente in uso su una coda.

È possibile impostare o visualizzare il valore di questo attributo utilizzando i comandi MQSC, IBM MQ Explorero administrative REST API.

Per ulteriori informazioni, consultare CURMAXFS .

L'equivalente PCF di questo attributo è **MQIA_CUR_MAX_FILE_SIZE**. Vedere Coda di interrogazione e Coda di interrogazione (risposta).

Dimensione e granularità del blocco

I file di coda sono divisi in segmenti denominati blocchi. Per aumentare la dimensione massima di un file di coda, potrebbe essere necessario che il gestore code modificasse la dimensione del blocco o la granularità della coda.

Se una coda appena definita viene creata con un valore **MAXFSIZE** elevato, la coda viene creata con una dimensione di blocco adatta. Tuttavia, se il valore **MAXFSIZE** di una coda esistente è stato aumentato, ad esempio utilizzando il comando MQSC **ALTER QLOCAL** , potrebbe essere necessario consentire lo svuotamento della coda per consentire al gestore code di riconfigurare la coda.

Per ulteriori informazioni, consultare “Calcolo della quantità di dati che un file di coda IBM MQ può memorizzare” a pagina 149.

Attenzione: Alcuni file system e sistemi operativi hanno dei limiti sulla dimensione dell'intero file system o sulla dimensione di un singolo file. È necessario verificare i limiti sui sistemi utilizzati dall'azienda.

Riferimenti correlati
MODIFICA CODE
VISUALIZZAZIONE CODA
VISUALIZZAZIONE QSTATUS

Multi *Modifica della dimensione di un file di coda IBM MQ*

È possibile aumentare o diminuire la dimensione massima di un file di coda.

Prima di iniziare

Nota: La procedura in questa attività richiede l'esecuzione dei comandi MQSC:

- ALW Su AIX, Linux, and Windows, si immettono comandi MQSC da un prompt dei comandi **runmqsc** . Consultare Esecuzione interattiva dei comandi MQSC in **runmqsc** e Esecuzione dei comandi MQSC dai file di testo in **runmqsc**.
- IBM i Su IBM i, si crea un elenco di comandi in un file Script, quindi si esegue il file utilizzando il comando **STRMQMMQSC** . Consultare Gestione utilizzando i comandi MQSC su IBM i.

Prima di impostare una nuova dimensione per un file di coda, utilizzare il comando MQSC DISPLAY QLOCAL per visualizzare la dimensione del file di coda che si desidera modificare. Ad esempio, emettere il seguente comando:

```
DISPLAY QLOCAL(SYSTEM.DEFAULT.LOCAL.QUEUE) MAXFSIZE
```

Si riceve il seguente output:

```
AMQ8409I: Display queue details
```

```
   QUEUE(SYSTEM.DEFAULT.LOCAL.QUEUE)        TYPE(QLOCAL)
   MAXFSIZE(DEFAULT)
```

che mostra che la dimensione massima del file di coda è il valore predefinito di 2.088.960 MB.

Informazioni su questa attività

Le seguenti procedure mostrano come:

- Ridurre la dimensione massima che un file di coda può raggiungere.
- Aumentare la dimensione massima di un file di coda.

Attenzione: È necessario essere prudenti nell'aumentare la dimensione dei file della coda senza considerare il modo in cui le applicazioni vengono scritte e il possibile effetto sulle prestazioni. L'accesso ai messaggi in modo casuale in un file di coda molto grande può essere molto lento.

Se si sta considerando di aumentare la dimensione massima di un file di coda oltre il valore predefinito, è necessario essere prudenti nell'utilizzo dei selettori di messaggi come gli ID di correlazione e le stringhe del selettore IBM MQ classes for JMS JM 3.0 o IBM MQ classes for Jakarta Messaging . I file di coda più grandi sono più adatti per il primo accesso alla coda.

La presenza di grandi quantità di dati in singoli file di coda deve essere eseguita solo su gestori code configurati per la registrazione circolare o in cui l'imaging del supporto non è stato abilitato per la singola coda.

Non è necessario limitare la dimensione delle code SYSTEM poiché ciò potrebbe influire sul funzionamento del gestore code.

Procedura

1. Ridurre la dimensione massima del file di coda
   a) Immettere il seguente comando MQSC per creare un file locale denominato SMALLQUEUE, con una dimensione di 500 gigabyte:

```
DEFINE QLOCAL(SMALLQUEUE) MAXFSIZE(512000)
     2 : DEFINE QLOCAL(SMALLQUEUE) MAXFSIZE(512000)
AMQ8006I: IBM MQ queue created
```

   e si riceve il messaggio: AMQ8006I:

   Nota: Se si configura una coda con un valore inferiore alla quantità di dati già presente nel file, i nuovi messaggi non possono essere inseriti nella coda.

   Se un'applicazione tenta di inserire un messaggio in un file di coda che non dispone di spazio sufficiente, l'applicazione riceve il codice di ritorno MQRC_Q_SPACE_NOT_AVAILABLE. Quando un numero sufficiente di messaggi viene letto in modo distruttivo dalla coda, le applicazioni possono iniziare a inserire nuovi messaggi nella coda.
2. Aumentare la dimensione massima del file della coda.
   a) Immettere il seguente comando MQSC per creare un file locale denominato LARGEQUEUE, con una dimensione di 5 terabyte:

```
DEFINE QLOCAL(LARGEQUEUE) MAXFSIZE(5242880)
     3 : DEFINE QLOCAL(LARGEQUEUE) MAXFSIZE(5242880)
AMQ8006I: IBM MQ queue created
```

Multi ***Calcolo della quantità di dati che un file di coda IBM MQ può memorizzare***

La quantità di dati che possono essere memorizzati su una coda è limitata dalla dimensione dei singoli blocchi in cui è divisa la coda. Utilizzare i comandi MQSC per confermare la dimensione del blocco e la granularità e verificare la dimensione di un file della coda.

Prima di iniziare

Nota: La procedura in questa attività richiede l'esecuzione dei comandi MQSC:

- ALW Su AIX, Linux, and Windows, si immettono comandi MQSC da un prompt dei comandi **runmqsc** . Consultare Esecuzione interattiva dei comandi MQSC in **runmqsc** e Esecuzione dei comandi MQSC dai file di testo in **runmqsc**.
- IBM i Su IBM i, si crea un elenco di comandi in un file Script, quindi si esegue il file utilizzando il comando **STRMQMMQSC** . Consultare Gestione utilizzando i comandi MQSC su IBM i.

Procedura

- Confermare la dimensione del blocco e la granularità.

  La dimensione blocco predefinita è 512 byte. Per supportare file di coda superiori a due terabyte, il gestore code dovrà aumentare la dimensione del blocco.

  La dimensione del blocco viene calcolata automaticamente quando si configura MAXFSIZE per una coda, ma la dimensione del blocco modificata non può essere applicata alla coda se la coda contiene già messaggi. Quando una coda è vuota, il gestore code modifica automaticamente la dimensione del blocco per supportare il **MAXFSIZE**configurato.

  Il comando **DISPLAY QSTATUS** ha un nuovo attributo, CURMAXFS, che consente di confermare che una coda è stata modificata per utilizzare una nuova dimensione blocco.

  Nel seguente esempio, il valore **CURMAXFS** di 4177920 conferma che la dimensione del file della coda è attualmente di circa quattro terabyte. Se il valore di **MAXFSIZE** configurato nella coda è maggiore del valore di **CURMAXFS**, il gestore code è ancora in attesa che la coda venga svuotata prima di riconfigurare la dimensione del blocco del file della coda.

```
DISPLAY QSTATUS(LARGEQUEUE)  CURMAXFS
     2 : DISPLAY QSTATUS(LARGEQUEUE)  CURMAXFS
AMQ8450I: Display queue status details
  QUEUE(LARGEQUEUE)                       TYPE(QUEUE)
  CURMAXFS(4177920)                       CURDEPTH(100000)
```

- Controllare la dimensione di un file di coda.

  È possibile visualizzare la dimensione corrente di un file di coda su disco, in megabyte, utilizzando l'attributo CURFSIZE nel comando **DISPLAY QSTATUS** . Ciò può essere utile su piattaforme come IBM MQ Appliance, dove non è possibile accedere direttamente al file system.

```
DISPLAY QSTATUS(SMALLQUEUE)  CURFSIZE
     1 : DISPLAY QSTATUS(SMALLQUEUE)  CURFSIZE
AMQ8450I: Display queue status details
  QUEUE(SMALLQUEUE)                       TYPE(QUEUE)
  CURDEPTH(4024)                          CURFSIZE(10)
```

  Nota: Quando una coda ha dei messaggi rimossi, l'attributo **CURFSIZE** non diminuisce immediatamente.

  In genere, lo spazio in un file di coda viene rilasciato solo quando nessuna applicazione ha la coda aperta e non ci sono messaggi in dubbio memorizzati sulla coda. Qualsiasi troncamento o compattazione necessaria di un file di coda caricato dal gestore code si verifica durante il checkpoint, l'arresto del gestore code o durante la registrazione di un'immagine del supporto della coda.

Riferimenti correlati

MODIFICA CODE

VISUALIZZAZIONE QSTATUS

Gestione delle code remote

Una coda remota è una definizione su un gestore code locale che fa riferimento a un gestore code remoto. Non è necessario definire una coda remota da una posizione locale, ma in questo caso le applicazioni possono fare riferimento alla coda remota in base al nome definito localmente invece di dover specificare un nome qualificato dall'ID del gestore code su cui si trova la coda remota.

Prima di iniziare

Nota: I passi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Informazioni su questa attività

Un'applicazione si connette a un gestore code locale ed emette una chiamata MQOPEN . Nella chiamata di apertura, il nome coda specificato è quello di una definizione di coda remota sul gestore code locale. La definizione della coda remota fornisce i nomi della coda di destinazione, del gestore code di destinazione e, facoltativamente, di una coda di trasmissione. Per inserire un messaggio sulla coda remota, l'applicazione emette una chiamata MQPUT , specificando l'handle restituito dalla chiamata MQOPEN . Il gestore code utilizza il nome coda remota e il nome gestore code remoto in un'intestazione di trasmissione all'inizio del messaggio. Queste informazioni vengono utilizzate per instradare il messaggio alla destinazione corretta nella rete.

Come amministratore, è possibile controllare la destinazione del messaggio modificando la definizione della coda remota.

Procedura

- Inserire un messaggio in una coda di proprietà di un gestore code remoto.

  L'applicazione si connette a un gestore code, ad esempio `saturn.queue.manager`. La coda di destinazione è di proprietà di un altro gestore code.

  Nella chiamata MQOPEN , l'applicazione specifica questi campi:

Valore del campo	Descrizione
ObjectName CYAN.REMOTE.QUEUE	Specifica il nome locale dell'oggetto coda remota. Definisce la coda di destinazione e il gestore code di destinazione.
ObjectType (coda)	Identifica questo oggetto come coda.
ObjectQmgrName Vuoto o `saturn.queue.manager`	Questo campo è facoltativo. Se vuoto, viene utilizzato il nome del gestore code locale. (Questo è il gestore code su cui esiste la definizione della coda remota).

  In seguito, l'applicazione emette una chiamata MQPUT per inserire un messaggio in questa coda.

  Sul gestore code locale, è possibile creare una definizione locale di una coda remota utilizzando i seguenti comandi 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)
```

  dove:

QREMOTE (CYAN.REMOTE.QUEUE)
Specifica il nome locale dell'oggetto coda remota. Questo è il nome che le applicazioni connesse a questo gestore code devono specificare nella chiamata MQOPEN per aprire la coda AUTOMOBILE.INSURANCE.QUOTE.QUEUE sul gestore code remoto `jupiter.queue.manager`.

DESCR ('Queue for auto insurance requests from the branches')
Fornisce ulteriore testo che descrive l'utilizzo della coda.

RNAME (AUTOMOBILE.INSURANCE.QUOTE.QUEUE)
Specifica il nome della coda di destinazione sul gestore code remoto. È la coda di destinazione reale per i messaggi inviati dalle applicazioni che specificano il nome della coda CYAN.REMOTE.QUEUE. La coda AUTOMOBILE.INSURANCE.QUOTE.QUEUE deve essere definito come coda locale sul Gestore code remoto.

RQMNAME (jupiter.queue.manager)
Specifica il nome del gestore code remoto che possiede la coda di destinazione AUTOMOBILE.INSURANCE.QUOTE.QUEUE.

XMITQ (INQUOTE.XMIT.QUEUE)
Specifica il nome della coda di trasmissione. Questa opzione è facoltativa; se il nome di una coda di trasmissione non viene specificato, viene utilizzata una coda con lo stesso nome del gestore code remoto.

In entrambi i casi, la coda di trasmissione appropriata deve essere definita come una coda locale con un attributo **Usage** che specifica che è una coda di trasmissione (USAGE (XMITQ) nei comandi MQSC).

- Inserire i messaggi su una coda remota (metodo alternativo).

  L'uso di una definizione locale di una coda remota non è l'unico modo per inserire messaggi in una coda remota. Le applicazioni possono specificare il nome completo della coda, incluso il nome gestore code remoto, come parte della chiamata MQOPEN . In questo caso, non è necessaria una definizione locale di una coda remota. Tuttavia, ciò significa che le applicazioni devono conoscere o avere accesso al nome del gestore code remoto in fase di runtime.
- Utilizzare altri comandi con le code remote.

  È possibile utilizzare i comandi MQSC per visualizzare o modificare gli attributi di un oggetto coda remota oppure è possibile eliminare l'oggetto coda remota. Ad esempio:

  – Per visualizzare gli attributi della coda remota:

  ```
  DISPLAY QUEUE (CYAN.REMOTE.QUEUE)
  ```

  – Per modificare la coda remota per abilitare gli inserimenti. Ciò non influisce sulla coda di destinazione, ma solo sulle applicazioni che specificano questa coda remota:

  ```
  ALTER QREMOTE (CYAN.REMOTE.QUEUE) PUT(ENABLED)
  ```

  – Per eliminare questa coda remota. Ciò non influisce sulla coda di destinazione, ma solo sulla sua definizione locale:

  ```
  DELETE QREMOTE (CYAN.REMOTE.QUEUE)
  ```

  Nota: Quando si elimina una coda remota, si elimina solo la rappresentazione locale della coda remota. Non si elimina la coda remota stessa o alcun messaggio su di essa.

Le definizioni di coda remota possono essere utilizzate come alias

Oltre a individuare una coda su un altro gestore code, è anche possibile utilizzare una definizione locale di una coda remota per gli alias del gestore code e gli alias della coda di risposta. Entrambi i tipi di alias vengono risolti tramite la definizione locale di una coda remota. È necessario impostare i canali appropriati affinché il messaggio arrivi a destinazione.

Alias del gestore code

Un alias è il processo con cui il nome del gestore code di destinazione, come specificato in un messaggio, viene modificato da un gestore code sull'instradamento del messaggio. Gli alias dei gestori code sono importanti perché è possibile utilizzarli per controllare la destinazione dei messaggi all'interno di una rete di gestori code.

A tale scopo, modificare la definizione della coda remota sul gestore code nel punto di controllo. L'applicazione mittente non è a conoscenza del fatto che il nome del gestore code specificato è un alias.

Per ulteriori informazioni sugli alias dei gestori code, consultare Cosa sono gli alias?

Alias coda di risposta

Facoltativamente, un'applicazione può specificare il nome di una coda di risposta quando inserisce un *messaggio di richiesta* in una coda.

Se l'applicazione che elabora il messaggio estrae il nome della coda di risposta, sa dove inviare il *messaggio di risposta*, se necessario.

Un alias della coda di risposta è il processo mediante il quale una coda di risposta, come specificato in un messaggio di richiesta, viene modificata da un gestore code sull'instradamento del messaggio. L'applicazione mittente non riconosce che il nome della coda di risposta specificato è un alias.

Un alias della coda di risposta consente di modificare il nome della coda di risposta e, facoltativamente, il gestore code. Questo a sua volta consente di controllare quale instradamento viene utilizzato per i messaggi di risposta.

Per ulteriori informazioni sui messaggi di richiesta, sui messaggi di risposta e sulle code di risposta, consultare Tipi di messaggio e Coda di risposta e gestore code.

Per ulteriori informazioni sugli alias della coda reply - to, consultare Cluster e alias della coda reply - to.

Gestione delle code alias

È possibile definire una coda alias per fare riferimento indirettamente ad un'altra coda o argomento.

Prima di iniziare

Nota: I passi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Attenzione: Gli elenchi di distribuzione non supportano l'utilizzo di code alias che puntano agli oggetti argomento. Se una coda alias punta a un oggetto argomento in un elenco di distribuzione, IBM MQ restituisce `MQRC_ALIAS_BASE_Q_TYPE_ERROR`.

Informazioni su questa attività

La coda a cui fa riferimento una coda alias può essere una delle seguenti:

- Una coda locale (consultare “Definizione di una coda locale con DEFINE QLOCAL” a pagina 141).
- Una definizione locale di una coda remota (consultare “Gestione delle code remote” a pagina 151).
- Un argomento.

Una coda alias non è una coda reale, ma una definizione che si risolve in una coda reale (o di destinazione) al runtime. La definizione della coda alias specifica la coda di destinazione. Quando un'applicazione effettua una chiamata `MQOPEN` a una coda alias, il gestore code risolve l'alias nel nome coda di destinazione.

Una coda alias non può essere risolta in un'altra coda alias definita localmente. Tuttavia, una coda alias può risolversi in code alias definite altrove nei cluster di cui è membro il gestore code locale. Per ulteriori informazioni, consultare Risoluzione dei nomi .

Le code alias sono utili per:

- Fornire alle diverse applicazioni diversi livelli di autorizzazioni di accesso alla coda di destinazione.
- Consentire a diverse applicazioni di gestire la stessa coda in modi diversi. (È possibile che si desideri assegnare priorità predefinite differenti o valori di persistenza predefiniti differenti.)
- Semplificazione della manutenzione, della migrazione e del bilanciamento del carico di lavoro. (Forse si desidera modificare il nome della coda di destinazione senza dover modificare l'applicazione, che continua a utilizzare l'alias.)

Ad esempio, si supponga che un'applicazione sia stata sviluppata per inserire i messaggi su una coda denominata MY.ALIAS.QUEUE. Specifica il nome di questa coda quando effettua una richiesta MQOPEN e, indirettamente, se inserisce un messaggio su questa coda. L'applicazione non è consapevole che la coda è una coda alias. Per ogni chiamata MQI che utilizza questo alias, il gestore code risolve il nome della coda reale, che potrebbe essere una coda locale o una coda remota definita su questo gestore code.

Modificando il valore dell'attributo TARGET, è possibile reindirizzare chiamate MQI a un'altra coda, possibilmente su un altro gestore code. È utile per la manutenzione, la migrazione e il bilanciamento del carico.

Procedura

- Definire una coda alias.

  Il seguente comando MQSC crea una coda alias:

  ```
  DEFINE QALIAS (MY.ALIAS.QUEUE) TARGET (YELLOW.QUEUE)
  ```

  Questo comando reindirizza le chiamate MQI che specificano MY.ALIAS.QUEUE alla coda YELLOW.QUEUE. Il comando non crea la coda di destinazione; le chiamate MQI hanno esito negativo se la coda è YELLOW.QUEUE non esiste in fase di runtime.

  Se si modifica la definizione dell'alias, è possibile reindirizzare le chiamate MQI a un'altra coda. Ad esempio:

  ```
  ALTER QALIAS (MY.ALIAS.QUEUE) TARGET (MAGENTA.QUEUE)
  ```

  Questo comando reindirizza le chiamate MQI a un'altra coda, MAGENTA.QUEUE.

  È inoltre possibile utilizzare le code alias per fare in modo che una singola coda (la coda di destinazione) sembri avere attributi differenti per applicazioni differenti. Questa operazione viene eseguita definendo due alias, uno per ogni applicazione. Si supponga che vi siano due applicazioni:

  – L'applicazione ALPHA può inserire messaggi su YELLOW.QUEUE, ma non è consentito richiamare messaggi da esso.
  – L'applicazione BETA può ricevere messaggi da YELLOW.QUEUE, ma non è consentito inserire messaggi su di esso.

  Il comando MQSC riportato di seguito definisce un alias inserito abilitato e disabilitato per l'applicazione ALPHA:

  ```
  DEFINE QALIAS (ALPHAS.ALIAS.QUEUE) +
  TARGET (YELLOW.QUEUE) +
  PUT (ENABLED) +
  GET (DISABLED)
  ```

  Il seguente comando definisce un alias che viene inserito disabilitato e abilitato per l'applicazione BETA:

  ```
  DEFINE QALIAS (BETAS.ALIAS.QUEUE) +
  TARGET (YELLOW.QUEUE) +
  ```

```
PUT (DISABLED) +
GET (ENABLED)
```

ALPHA utilizza il nome coda ALPHAS.ALIAS.QUEUE nelle sue chiamate MQI; BETA utilizza il nome della coda BETAS.ALIAS.QUEUE. Entrambi accedono alla stessa coda, ma in modi diversi.

È possibile utilizzare gli attributi LIKE e REPLACE quando si definiscono gli alias della coda, nello stesso modo in cui si utilizzano questi attributi con code locali.

- Utilizzare altri comandi con le code alias.

  È possibile utilizzare i comandi MQSC appropriati per la visualizzazione o la modifica degli attributi della coda alias o per eliminare l'oggetto della coda alias. Ad esempio:

  Utilizzare il comando **DISPLAY QALIAS** per visualizzare gli attributi della coda alias:

  ```
  DISPLAY QALIAS (ALPHAS.ALIAS.QUEUE)
  ```

  Utilizzare il comando **ALTER QALIAS** per modificare il nome della coda di base, in cui l'alias si risolve, in cui l'opzione `force` forza la modifica anche se la coda è aperta:

  ```
  ALTER QALIAS (ALPHAS.ALIAS.QUEUE) TARGET(ORANGE.LOCAL.QUEUE) FORCE
  ```

  Utilizzare il comando **DELETE QALIAS** per eliminare questo alias della coda:

  ```
  DELETE QALIAS (ALPHAS.ALIAS.QUEUE)
  ```

  Non è possibile eliminare una coda alias se un'applicazione ha attualmente la coda aperta.

Concetti correlati
Liste di distribuzione
Riferimenti correlati
MODIFICA QALIAS
DEFINE QALIAS
DELETE QALIAS

Utilizzo delle code modello

Le code modello forniscono un metodo conveniente per le applicazioni per creare le code come richiesto.

Prima di iniziare

Nota: I passi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Informazioni su questa attività

Un gestore code crea una *coda dinamica* se riceve una chiamata MQI da un'applicazione che specifica un nome coda definito come coda modello. Il nome della nuova coda dinamica viene generato dal gestore code quando viene creata la coda. Una *coda modello* è un modello che specifica gli attributi di tutte le code dinamiche da essa create.

Procedura

- Definire una coda modello.

  Utilizzare il comando MQSC **DEFINE QMODEL** per definire una coda modello con una serie di attributi nello stesso modo in cui si definisce una coda locale. Le code modello e le code locali hanno la stessa serie di attributi, tranne per il fatto che sulle code modello è possibile specificare se le code

dinamiche create sono temporanee o permanenti. Le code permanenti vengono gestite durante i riavvii del gestore code, mentre quelle temporanee non lo sono. Ad esempio:

```
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)
```

Questo comando crea una definizione di coda modello. Dall'attributo **DEFTYPE** , è possibile vedere che le code effettive create da questo modello sono code dinamiche permanenti. Gli attributi non specificati vengono automaticamente copiati da SYSYTEM SYSYTEM.DEFAULT.MODEL.QUEUE predefinita.

È possibile utilizzare gli attributi **LIKE** e **REPLACE** quando si definiscono code modello, nello stesso modo in cui si utilizzano le code locali.

- Utilizzare altri comandi con le code modello.

  È possibile utilizzare i comandi MQSC appropriati per visualizzare o modificare gli attributi di una coda modello o per eliminare l'oggetto coda modello. Ad esempio:

  Utilizzare il comando **DISPLAY QUEUE** per visualizzare gli attributi della coda modello:

  ```
  DISPLAY QUEUE (GREEN.MODEL.QUEUE)
  ```

  Utilizzare il comando **ALTER QMODEL** per modificare il modello per abilitare gli inserimenti su qualsiasi coda dinamica creata da questo modello:

  ```
  ALTER QMODEL (BLUE.MODEL.QUEUE) PUT(ENABLED)
  ```

  Utilizzare il comando **DELETE QMODEL** per eliminare questa coda modello:

  ```
  DELETE QMODEL (RED.MODEL.QUEUE)
  ```

Riferimenti correlati
MODIFICA QMODEL
DEFINE QMODEL
DELETE QMODEL
VISUALIZZAZIONE CODA

Gestione delle code di messaggi non recapitabili

Ogni gestore code generalmente ha una coda locale da utilizzare come coda di messaggi non instradabili, in modo che i messaggi che non possono essere consegnati alla destinazione corretta possano essere memorizzati per un successivo richiamo. Si indica al gestore code la coda di messaggi non instradabili e si specifica il modo in cui devono essere elaborati i messaggi rilevati su una coda di messaggi non instradabili. L'utilizzo di code di messaggi non recapitabili può influire sulla sequenza in cui vengono consegnati i messaggi, pertanto è possibile scegliere di non utilizzarle.

Prima di iniziare

Nota: I passi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Informazioni su questa attività

Una coda di messaggi non recapitabili di esempio denominata SYSTEM.DEAD.LETTER.QUEUE è disponibile con il prodotto. Questa coda viene creata automaticamente quando si crea il gestore code. Se necessario, è possibile modificare questa definizione e ridenominarla.

Una coda di messaggi non recapitabili non ha requisiti speciali tranne che:

- Deve essere una coda locale
- Il suo attributo MAXMSGL (lunghezza massima del messaggio) deve abilitare la coda ad accogliere i messaggi più grandi che il gestore code deve gestire **più** la dimensione dell'intestazione dei messaggi non instradabili (MQDLH)

L'utilizzo di code di messaggi non recapitabili può influire sulla sequenza in cui vengono consegnati i messaggi, pertanto è possibile scegliere di non utilizzarle.

Procedura

- Comunicare al gestore code la coda di messaggi non recapitabili.

  A tale scopo, specificare un nome coda di messaggi non recapitabili nel comando **crtmqm** (`crtmqm -u DEAD.LETTER.QUEUE`, ad esempio) oppure utilizzando l'attributo **DEADQ** nel comando **ALTER QMGR** per specificarne uno in un secondo momento. È necessario definire la coda di messaggi non recapitabili prima di utilizzarla.
- Specificare in che modo devono essere elaborati i messaggi trovati su una coda di messaggi non instradabili.

  Impostare l'attributo del canale USEDLQ per determinare se la coda di messaggi non recapitabili viene utilizzata quando i messaggi non possono essere consegnati. Questo attributo può essere configurato in modo che alcune funzioni del gestore code utilizzino la coda di messaggi non recapitabili, mentre altre non lo utilizzano. Per ulteriori informazioni sull'utilizzo dell'attributo del canale USEDLQ in diversi comandi MQSC, consultare DEFINE CHANNEL, DISPLAY CHANNEL, ALTER CHANNELe DISPLAY CLUSQMGR.

  Si utilizza il gestore code di messaggi non instradabili IBM MQ per specificare il modo in cui i messaggi trovati su una coda di messaggi non instradabili devono essere elaborati o rimossi. Consultare “Elaborazione di messaggi su una coda di messaggi non instradabili IBM MQ” a pagina 157.

Concetti correlati

Code di messaggi non recapitabili

Attività correlate

Risoluzione dei problemi dei messaggi non recapitati

Riferimenti correlati

Gestore code ALTER

crtmqm (crea gestore code)

Elaborazione di messaggi su una coda di messaggi non instradabili IBM MQ

Per elaborare i messaggi su una DLQ (dead - letter queue), utilizzare il gestore DLQ predefinito fornito da IBM MQ. Il gestore mette in corrispondenza i messaggi sulla DLQ con le voci in una tabella delle regole definita.

Informazioni su questa attività

I messaggi possono essere inseriti su una DLQ dai gestori code, dagli MCA (message channel agent) e dalle applicazioni. Tutti i messaggi sulla DLQ devono avere come prefisso una struttura *dead-letter header* , MQDLH. I messaggi inseriti nella DLQ da un gestore code o da un agente del canale dei messaggi hanno sempre questa intestazione; le applicazioni che inserendo i messaggi nella DLQ devono fornire questa intestazione. Il campo *Motivo* della struttura MQDLH contiene un codice motivo che identifica il motivo per cui il messaggio si trova sulla DLQ.

Tutti gli ambienti IBM MQ richiedono una routine per elaborare regolarmente i messaggi sulla DLQ. IBM MQ fornisce una routine predefinita, denominata *gestore code di messaggi non instradabili (DLQ)* , richiamata utilizzando il comando MQSC **runmqdlq** .

Le istruzioni per l'elaborazione dei messaggi sul DLQ vengono fornite al gestore DLQ mediante una *tabella di regole*scritta dall'utente. Ciò significa che il gestore DLQ mette in corrispondenza i messaggi sulla DLQ con le voci nella tabella delle regole; quando un messaggio DLQ corrisponde a una voce nella tabella delle regole, il gestore DLQ esegue l'azione associata a tale voce.

Attività correlate

Risoluzione dei problemi dei messaggi non recapitati

Riferimenti correlati

Code di messaggi non recapitabili

z/OS ALW *Richiamo del gestore code di messaggi non recapitabili*

Richiamare il gestore DLQ (dead - letter queue) utilizzando il comando di controllo **runmqdl**q. È possibile denominare la DLQ che si desidera elaborare e il gestore code che si desidera utilizzare in due modi.

Prima di iniziare

Per eseguire il gestore DLQ è necessario essere autorizzati ad accedere sia alla DLQ stessa che a qualsiasi coda messaggi a cui vengono inoltrati i messaggi sulla DLQ. Per consentire al gestore DLQ di inserire i messaggi nelle code con l'autorità dell'ID utente nel contesto del messaggio, è inoltre necessario essere autorizzati ad assumere l'identità di altri utenti.

Informazioni su questa attività

I seguenti esempi si applicano alla DLQ denominata `ABC1.DEAD.LETTER.QUEUE`, di proprietà del gestore code `ABC1.QUEUE.MANAGER`.

Se non si specifica la DLQ o il gestore code come mostrato, il gestore code predefinito per l'installazione viene utilizzato insieme alla DLQ che appartiene a tale gestore code.

Il comando **runmqdlq** prende il suo input da `stdin`. Associare la tabella delle regole a **runmqdlq** reindirizzando `stdin` dalla tabella delle regole.

Per ulteriori informazioni sul comando **runmqdlq** , consultare runmqdlq.

Procedura

- È possibile denominare la DLQ e il gestore code come parametri del comando **runmqdlq** .

  Ad esempio, dal prompt dei comandi:

  ```
  runmqdlq ABC1.DEAD.LETTER.QUEUE ABC1.QUEUE.MANAGER <qrule.rul
  ```

- È possibile denominare il DLQ e il gestore code nella tabella delle regole.

  Ad esempio:

  ```
  INPUTQ(ABC1.DEAD.LETTER.QUEUE) INPUTQM(ABC1.QUEUE.MANAGER)
  ```

Concetti correlati

Code di messaggi non recapitabili

Attività correlate

Risoluzione dei problemi dei messaggi non recapitati

z/OS ALW *Il gestore DLQ di esempio **amqsdlq***

Oltre al gestore code di messaggi non recapitabili richiamato mediante il comando **runmqdlq** , IBM MQ fornisce l'origine di un gestore DLQ di esempio **amqsdlq** con una funzione simile a quella fornita da **runmqdlq**.

È possibile personalizzare **amqsdlq** in modo da fornire un gestore DLQ che soddisfi i requisiti. Ad esempio, è possibile decidere che si desidera un gestore DLQ che possa elaborare i messaggi senza intestazioni di messaggi non recapitabili. (Sia il gestore DLQ predefinito che l'esempio, **amqsdlq**, elaborano solo i messaggi sul DLQ che cominciano con un'intestazione di messaggi non recapitabili, MQDLH. I messaggi che non iniziano con MQDLH vengono identificati come in errore e rimangono sulla DLQ per un tempo indefinito.)

MQ_INSTALLATION_PATH rappresenta la directory di livello superiore in cui è installato IBM MQ .

In IBM MQ for Windows, l'origine di amqsdlq viene fornita nella directory:

```
MQ_INSTALLATION_PATH\tools\c\samples\dlq
```

e la versione compilata viene fornita nella directory:

```
MQ_INSTALLATION_PATH\tools\c\samples\bin
```

Nei sistemi IBM MQ for UNIX e Linux , l'origine di **amqsdlq** viene fornita nella directory:

MQ_INSTALLATION_PATH/samp/dlq

e la versione compilata viene fornita nella directory:

MQ_INSTALLATION_PATH/samp/bin

Viene inclusa una versione integrata del programma di esempio, denominata **amqsdlqc**. È possibile utilizzarlo per connettersi a un gestore code remoto in modalità client. Per utilizzare **amqsdlqc** è necessario impostare una delle variabili di ambiente *MQSERVER* , **MQCHLLIB** o **MQCHLTAB** per identificare la modalità di connessione al gestore code. Ad esempio:

```
export MQSERVER="SYSTEM.DEF.SVRCONN/TCP/myappliance.co.uk(1414)"
```

z/OS ALW *Tabella regole gestore DLQ*

La tabella delle regole del gestore code di messaggi non instradabili definisce il modo in cui il gestore DLQ elabora i messaggi che arrivano sulla DLQ.

Ci sono due tipi di voce in una tabella di regole:

- La prima voce nella tabella, che è facoltativa, contiene *dati di controllo*.
- Tutte le altre voci nella tabella sono *regole* per il gestore DLQ da seguire. Ogni regola è composta da un *pattern* (una serie di caratteristiche del messaggio) rispetto al quale viene confrontato un messaggio e da un' *azione* da eseguire quando un messaggio sulla DLQ corrisponde al pattern specificato. In una tabella di regole deve essere presente almeno una regola.

Ciascuna voce nella tabella delle regole è composta da una o più parole chiave.

Concetti correlati

Code di messaggi non recapitabili

Attività correlate

Risoluzione dei problemi dei messaggi non recapitati

z/OS ALW *Dati di controllo DLQ*

È possibile includere parole chiave in una voce di dati di controllo in una tabella di regole del gestore code di messaggi non recapitabili.

Nota:

- La linea verticale (|) separa le alternative, solo una delle quali può essere specificata.
- Tutte le parole chiave sono facoltative.

INPUTQ (*QueueName*| '' (valore predefinito))
Il nome della DLQ che si desidera elaborare:

1. Qualsiasi valore INPUTQ fornito come parametro per il comando `runmqdlq` sovrascrive qualsiasi valore INPUTQ nella tabella delle regole.
2. Se non si specifica un valore INPUTQ come parametro nel comando `runmqdlq` , ma **si** specifica un valore nella tabella delle regole, viene utilizzato il valore INPUTQ nella tabella delle regole.
3. Se non viene specificato alcun DLQ o si specifica INPUTQ ('') nella tabella delle regole, viene utilizzato il nome del DLQ appartenente al gestore code con il nome fornito come parametro per il comando `runmqdlq` .
4. Se non si specifica un valore INPUTQ come parametro per il comando `runmqdlq` o come valore nella tabella delle regole, viene utilizzato il DLQ che appartiene al gestore code denominato nella parola chiave INPUTQM nella tabella delle regole.

INPUTQM (*QueueManagerNome*| '' (valore predefinito))
Il nome del gestore code che possiede il DLQ denominato nella parola chiave INPUTQ:

1. Qualsiasi valore INPUTQM fornito come parametro al comando `runmqdlq` sovrascrive qualsiasi valore INPUTQM nella tabella delle regole.
2. Se non si specifica un valore INPUTQM come parametro del comando `runmqdlq` , viene utilizzato il valore INPUTQM nella tabella delle regole.
3. Se non viene specificato alcun gestore code oppure se si specifica INPUTQM ('') nella tabella delle regole, viene utilizzato il gestore code predefinito per l'installazione.

RETRYINT (*Intervallo*| 60 (predefinito))
L'intervallo, in secondi, con cui il gestore DLQ deve rielaborare i messaggi sulla DLQ che non è stato possibile elaborare al primo tentativo e per cui sono stati richiesti tentativi ripetuti. Per impostazione predefinita, l'intervallo tra i tentativi è 60 secondi.

WAIT (YES (predefinito) |NO|*nnn*)
Indica se il gestore DLQ deve attendere l'arrivo di ulteriori messaggi sulla DLQ quando rileva che non sono presenti ulteriori messaggi che può elaborare.

SÌ
Il gestore DLQ attende indefinitamente.

No
Il gestore DLQ termina quando rileva che il DLQ è vuoto o non contiene messaggi che può elaborare.

nnn
Il gestore DLQ attende per *nnn* secondi l'arrivo di nuovo lavoro prima di terminare, dopo aver rilevato che la coda è vuota o non contiene messaggi che è possibile elaborare.

Specificare WAIT (YES) per le DLQ occupate e WAIT (NO) o WAIT (*nnn*) per le DLQ che hanno un basso livello di attività. Se il gestore DLQ è autorizzato a terminare, richiamarlo di nuovo utilizzando il trigger. Per ulteriori informazioni sull'attivazione, consultare Avvio delle applicazioni IBM MQ utilizzando i trigger.

Un'alternativa all'inclusione dei dati di controllo nella tabella di regole è quella di fornire i nomi della DLQ e del relativo gestore code come parametri di input per il comando `runmqdlq` . Se si specifica un valore sia nella tabella delle regole che come input per il comando `runmqdlq` , il valore specificato nel comando `runmqdlq` ha la precedenza.

Se si include una voce di dati di controllo nella tabella delle regole, deve essere la **prima** voce nella tabella.

z/OS ALW *Regole DLQ (modelli e azioni)*

Una descrizione delle parole chiave corrispondenti al modello (quelle rispetto alle quali i messaggi sulla coda di messaggi non recapitabili corrispondono) e le parole chiave di azione (quelle che determinano come il gestore DLQ deve elaborare un messaggio corrispondente). Viene fornita anche una regola di esempio.

Le parole chiave corrispondenti al modello

Le parole chiave di corrispondenza del modello, che si utilizzano per specificare i valori rispetto ai quali i messaggi sulla DLQ corrispondono, sono le seguenti. (Tutte le parole chiave corrispondenti al modello sono facoltative):

APPLIDAT (*ApplIdentityData*|* (predefinito))
Il valore *ApplIdentityData* specificato nel descrittore del messaggio, MQMD, del messaggio sulla DLQ.

APPLNAME (*PutApplName*|* (predefinito))
Il nome dell'applicazione che ha emesso la chiamata MQPUT o MQPUT1 , come specificato nel campo *PutApplName* del descrittore del messaggio, MQMD, del messaggio sulla DLQ.

APPLTYPE (*PutApplTipo*|* (predefinito))
Il valore *PutApplType* , specificato nel descrittore del messaggio, MQMD, del messaggio sulla DLQ.

DESTQ (*QueueName*|* (valore predefinito))
Il nome della coda messaggi a cui è destinato il messaggio.

DESTQM (Nome *QueueManager*|* (predefinito))
Il nome del gestore code della coda messaggi a cui è destinato il messaggio.

FEEDBACK (*Feedback*|* (Predefinito))
Quando il valore *MsgType* è MQFB_REPORT, *Feedback* descrive la natura del report.

È possibile utilizzare nomi simbolici. Ad esempio, è possibile utilizzare il nome simbolico MQFB_COA per identificare i messaggi sulla DLQ che necessitano di conferma del relativo arrivo sulle code di destinazione.

FORMAT (*Formato*|* (predefinito))
Il nome utilizzato dal mittente del messaggio per descrivere il formato dei dati del messaggio.

MSGTYPE (*MsgType*|* (valore predefinito))
Il tipo di messaggio del DLQ.

È possibile utilizzare nomi simbolici. Ad esempio, è possibile utilizzare il nome simbolico MQMT_REQUEST per identificare i messaggi sulla DLQ che necessitano di risposte.

PERSIST (*Persistenza*|* (predefinito)
Il valore di persistenza del messaggio. La persistenza di un messaggio determina se sopravvive ai riavvii del gestore code.

È possibile utilizzare nomi simbolici. Ad esempio, è possibile utilizzare il nome simbolico MQPER_PERSISTENT per identificare i messaggi sulla DLQ persistenti.

REASON (*ReasonCode*|* (predefinito)
Il codice di errore che descrive il motivo per cui il messaggio è stato inserito nella DLQ.

È possibile utilizzare nomi simbolici. Ad esempio, è possibile utilizzare il nome simbolico MQRC_Q_FULL per identificare i messaggi collocati nella DLQ perché le code di destinazione erano piene.

REPLYQ (*QueueName*|* (predefinito))
Il nome della coda di risposta specificato nel descrittore del messaggio, MQMD, del messaggio sulla DLQ.

REPLYQM (*QueueManagerNome*|* (predefinito))
Il nome del gestore code della coda di risposta, come specificato nel descrittore del messaggio, MQMD, del messaggio sulla DLQ.

USERID (*UserIdentifier*|* (predefinito))
L'ID utente dell'utente che ha creato il messaggio sul DLQ, come specificato nel descrittore del messaggio, MQMD, del messaggio sul DLQ.

Le parole chiave di azione

Le parole chiave dell'azione, utilizzate per descrivere il modo in cui deve essere elaborato un messaggio corrispondente, sono le seguenti:

AZIONE (DISCARD | IGNORE | RETRY | FWD)
L'azione da intraprendere per qualsiasi messaggio sulla DLQ che corrisponde al modello definito in questa regola.

DISCARD
Cancellare il messaggio dalla DLQ.

IGNORE
Lasciare il messaggio sulla DLQ.

RIPROVA
Se il primo tentativo di inserire il messaggio nella relativa coda di destinazione ha esito negativo, riprovare. La parola chiave RETRY imposta il numero di tentativi effettuati per implementare un'azione. La parola chiave RETRYINT dei dati di controllo controlla l'intervallo tra i tentativi.

FWD
Inoltrare il messaggio alla coda denominata sulla parola chiave FWDQ.

È necessario specificare la parola chiave ACTION.

FWDQ (*QueueName*| & DESTQ | & REPLYQ)
Il nome della coda messaggi a cui inoltrare il messaggio quando viene richiesto ACTION (FWD).

QueueName
Il nome di una coda messaggi. FWDQ ('') non è valido.

& DESTQ
Prendere il nome della coda dal campo *DestQName* nella struttura MQDLH.

& REPLYQ
Prendere il nome della coda dal campo *ReplyToQ* nel descrittore del messaggio, MQMD.

Per evitare i messaggi di errore quando una regola che specifica FWDQ (& REPLYQ) corrisponde a un messaggio con un campo *ReplyToQ* vuoto, specificare REPLYQ (? *) nel modello del messaggio.

FWDQM (*QueueManagerNome*| & DESTQM | & REPLYQM | '' (valore predefinito))
Il gestore code della coda a cui inoltrare un messaggio.

QueueManagerName
Il nome del gestore code della coda a cui inoltrare un messaggio quando viene richiesto ACTION (FWD).

& DESTQM
Prendere il nome gestore code dal campo *DestQMgrNome* nella struttura MQDLH.

& REPLYQM
Prendere il nome del gestore code dal campo *ReplyToReplyTo* nel descrittore del messaggio, MQMD.

' '
FWDQM (''), che è il valore predefinito, identifica il gestore code locale.

HEADER (YES (predefinito) |NO)
Se MQDLH deve rimanere su un messaggio per cui è richiesto ACTION (FWD). Per impostazione predefinita, MQDLH rimane sul messaggio. La parola chiave HEADER non è valida per azioni diverse da FWD.

PUTAUT (DEF (predefinito) | CTX)
L'autorità con cui i messaggi devono essere inseriti dal gestore DLQ:

DEF
Inserire i messaggi con l'autorità del gestore DLQ.

CTX
Inserire i messaggi con l'autorizzazione dell'ID utente nel contesto del messaggio. Se si specifica PUTAUT (CTX), è necessario essere autorizzati ad assumere l'identità di altri utenti.

RETRY (*RetryCount*|1 (predefinito)
Il numero di volte, nell'intervallo compreso tra 1 e 999.999.999, per tentare un'azione (all'intervallo specificato nella parola chiave RETRYINT dei dati di controllo). Il conteggio dei tentativi effettuati dal gestore DLQ per implementare una particolare regola è specifico per l'istanza corrente del gestore DLQ; il conteggio non persiste durante i riavvii. Se il gestore DLQ viene riavviato, il numero di tentativi effettuati per applicare una regola viene reimpostato su zero.

Regola di esempio

Di seguito è riportata una regola di esempio da una tabella di regole del gestore DLQ:

```
PERSIST(MQPER_PERSISTENT) REASON (MQRC_PUT_INHIBITED) +
ACTION (RETRY) RETRY (3)
```

Questa regola indica al gestore DLQ di effettuare tre tentativi di consegnare alla coda di destinazione qualsiasi messaggio persistente inserito nella DLQ perché MQPUT e MQPUT1 non erano consentiti.

Tutte le parole chiave che è possibile utilizzare su una regola sono descritte nel resto di questa sezione. Tieni presente quanto segue:

- Il valore predefinito per una parola chiave, se presente, è sottolineato. Per la maggior parte delle parole chiave, il valore predefinito è * (asterisco), che corrisponde a qualsiasi valore.
- La linea verticale (|) separa le alternative, solo una delle quali può essere specificata.
- Tutte le parole chiave tranne ACTION sono facoltative.

z/OS ALW *Convenzioni della tabella regole DLQ*

La sintassi, la struttura e il contenuto della tabella delle regole del gestore code di messaggi non recapitabili devono rispettare queste convenzioni.

La tabella delle regole deve rispettare le convenzioni seguenti:

- Una tabella di regole deve contenere almeno una regola.
- Le parole chiave possono essere presenti in qualsiasi ordine.
- Una parola chiave può essere inclusa solo una volta in una regola.
- Le parole chiave non sono sensibili al maiuscolo / minuscolo.
- Una parola chiave e il relativo valore di parametro devono essere separati da altre parole chiave da almeno uno spazio vuoto o da una virgola.
- Ci può essere qualsiasi numero di spazi vuoti all'inizio o alla fine di una regola e tra parole chiave, punteggiatura e valori.
- Ogni regola deve iniziare su una nuova riga.
- Sui sistemi Windows , l'ultima regola della tabella deve terminare con un carattere di ritorno a capo / avanzamento riga. È possibile ottenere ciò assicurandosi di premere il tasto Invio alla fine della regola, in modo che l'ultima riga della tabella sia una riga vuota.
- Per motivi di portabilità, la lunghezza significativa di una riga non deve essere superiore a 72 caratteri.
- Utilizzare il segno più (+) come ultimo carattere non vuoto su una riga per indicare che la regola continua dal primo carattere non vuoto nella riga successiva. Utilizzare il segno meno (-) come ultimo carattere non vuoto su una riga per indicare che la regola continua dall'inizio della riga successiva. I caratteri di continuazione possono verificarsi all'interno di parole chiave e parametri.

Ad esempio:

```
APPLNAME('ABC+
D')
```

si traduce in "ABCD", e

```
APPLNAME('ABC-
D')
```

risultati in ' ABC D'.

- Le righe di commento, che iniziano con un asterisco (*), possono trovarsi in qualsiasi punto della tabella delle regole.
- Le righe vuote vengono ignorate.
- Ogni voce nella tabella delle regole del programma di gestione DLQ comprende una o più parole chiave e i relativi parametri associati. I parametri devono seguire queste regole di sintassi:
  - Ogni valore di parametro deve includere almeno un carattere significativo. Le virgolette singole delimitate in valori racchiusi tra virgolette non sono considerate significative. Ad esempio, questi parametri sono validi:

`FORMAT('ABC')`	3 caratteri significativi
`FORMAT(ABC)`	3 caratteri significativi
`FORMAT('A')`	1 carattere significativo
`FORMAT(A)`	1 carattere significativo
`FORMAT(' ')`	1 carattere significativo

    Questi parametri non sono validi perché non contengono caratteri significativi:

    ```
    FORMAT('')
    FORMAT( )
    FORMAT()
    FORMAT
    ```

  - sono supportati. È possibile utilizzare il punto interrogativo (?) invece di qualsiasi carattere singolo, tranne uno spazio finale; è possibile utilizzare l'asterisco (*) invece di zero o più caratteri adiacenti. L'asterisco (*) e il punto interrogativo (?) sono ***sempre*** interpretati come caratteri jolly nei valori dei parametri.
  - I caratteri jolly non possono essere inclusi nei parametri di queste parole chiave: ACTION, HEADER, RETRY, FWDQ, FWDQM e PUTAUT.
  - Gli spazi vuoti finali nei valori di parametro e nei campi corrispondenti nel messaggio sulla DLQ, non sono significativi quando si eseguono corrispondenze di caratteri jolly. Tuttavia, gli spazi vuoti iniziali e incorporati nelle stringhe racchiuse tra virgolette singole sono significativi per le corrispondenze di caratteri jolly.
  - I parametri numerici non possono includere il carattere jolly punto interrogativo (?). È possibile utilizzare l'asterisco (*) invece di un intero parametro numerico, ma non come parte di un parametro numerico. Ad esempio, questi sono parametri numerici validi:

`MSGTYPE(2)`	Sono idonei solo i messaggi di risposta
`MSGTYPE(*)`	Qualsiasi tipo di messaggio è idoneo
`MSGTYPE('*')`	Qualsiasi tipo di messaggio è idoneo

Tuttavia, `MSGTYPE('2*')` non è valido, poiché include un asterisco (*) come parte di un parametro numerico.

- I parametri numerici devono essere compresi tra 0 e 999 999 999. Se il valore del parametro è compreso in questo intervallo, viene accettato, anche se non è attualmente valido nel campo a cui si riferisce la parola chiave. È possibile utilizzare nomi simbolici per i parametri numerici.
- Se un valore stringa è più breve del campo in MQDLH o MQMD a cui si riferisce la parola chiave, il valore viene riempito con spazi vuoti fino alla lunghezza del campo. Se il valore, esclusi gli asterischi, è più lungo del campo, viene diagnosticato un errore. Ad esempio, questi sono tutti valori stringa validi per un campo di 8 caratteri:

  ```
  'ABCDEFGH'          8 caratteri
  'A*C*E*G*I'         5 caratteri esclusi gli asterischi
  '*A*C*E*G*I*K*M*O   8 caratteri esclusi gli asterischi
  *'
  ```

- Racchiudere le stringhe che contengono spazi, caratteri minuscoli o caratteri speciali diversi da punto (.), barra (?), carattere di sottolineatura (_) e segno di percentuale (%) tra virgolette singole. I caratteri minuscoli non racchiusi tra virgolette singole vengono piegati in maiuscolo. Se la stringa include una virgoletta, utilizzare due virgolette singole per indicare sia l'inizio che la fine della virgoletta. Quando viene calcolata la lunghezza della stringa, ogni ricorrenza delle virgolette viene conteggiata come un singolo carattere.

z/OS ALW *Modalità di elaborazione della tabella delle regole DLQ*

Il gestore code di messaggi non instradabili ricerca nella tabella delle regole una regola in cui il pattern corrisponde a un messaggio sulla DLQ.

La ricerca inizia con la prima regola nella tabella e continua in modo sequenziale attraverso la tabella. Quando il gestore DLQ trova una regola con un modello corrispondente, prende l'azione da tale regola. Il gestore DLQ incrementa il numero di tentativi per una regola di 1 ogni volta che applica tale regola. Se il primo tentativo ha esito negativo, il gestore DLQ tenta di nuovo fino a quando il numero di tentativi non corrisponde al numero specificato nella parola chiave RETRY. Se tutti i tentativi hanno esito negativo, il gestore DLQ ricerca la regola di corrispondenza successiva nella tabella.

Questo processo viene ripetuto per le regole di corrispondenza successive fino a quando un'azione ha esito positivo. Quando ogni regola di corrispondenza è stata tentata il numero di volte specificato nella relativa parola chiave RETRY e tutti i tentativi hanno avuto esito negativo, viene assunto ACTION (IGNORE). ACTION (IGNORE) viene anche assunto se non viene trovata alcuna regola corrispondente.

Nota:

1. I modelli di regole corrispondenti vengono ricercati solo per i messaggi sulla DLQ che iniziano con un MQDLH. I messaggi che non iniziano con MQDLH vengono riportati periodicamente come in errore e rimangono nella DLQ per un periodo di tempo indefinito.
2. Tutte le parole chiave del modello possono essere impostate come predefinite, in modo che una regola possa contenere solo un'azione. Tenere presente, tuttavia, che le regole di sola azione vengono applicate a tutti i messaggi nella coda che hanno MQDLH e che non sono già stati elaborati in conformità con altre regole nella tabella.
3. La tabella delle regole viene convalidata all'avvio del gestore DLQ e gli errori vengono contrassegnati in quel momento. È possibile apportare modifiche alla tabella delle regole in qualsiasi momento, ma tali modifiche non diventano effettive fino al riavvio del gestore DLQ.
4. Il gestore DLQ non modifica il contenuto dei messaggi, MQDLH o il descrittore del messaggio. Il gestore DLQ inserisce sempre i messaggi in altre code con l'opzione MQPMO_PASS_ALL_CONTEXT.
5. Gli errori di sintassi consecutivi nella tabella delle regole potrebbero non essere riconosciuti perché la tabella delle regole è progettata per eliminare la creazione di errori ripetitivi durante la convalida.
6. Il gestore DLQ apre il DLQ con l'opzione MQOO_INPUT_AS_Q_DEF.

7. È possibile eseguire simultaneamente più istanze del gestore DLQ sulla stessa coda, utilizzando la stessa tabella delle regole. Tuttavia, è più usuale che ci sia una relazione uno - a - uno tra una DLQ e un gestore DLQ.

Concetti correlati

Code di messaggi non recapitabili

Attività correlate

Risoluzione dei problemi dei messaggi non recapitati

z/OS ALW *Una tabella di regole gestore DLQ di esempio*

Una tabella di regole della coda di messaggi non instradati di esempio per il comando **runmqdlq** , contenente una singola immissione di dati di controllo e diverse regole.

```
************************************************************************
*    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)
```

Concetti correlati

Code di messaggi non recapitabili

Attività correlate

Risoluzione dei problemi dei messaggi non recapitati

Riferimenti correlati

runmqdlq (esecuzione gestore code di messaggi non instradabili)

IBM i *Richiamo del gestore code di messaggi non recapitabili su IBM i*

Su IBM MQ for IBM i, si richiama il gestore DLQ impostando il comando **STRMQMDLQ** .

Prima di iniziare

Per eseguire il gestore DLQ, è necessario essere autorizzati ad accedere sia alla DLQ stessa, sia a tutte le code di messaggi a cui vengono inoltrati i messaggi sulla DLQ. È inoltre necessario essere autorizzati ad assumere l'identità di altri utenti, affinché la DLQ possa inserire i messaggi nelle code con l'autorizzazione dell'ID utente nel contesto del messaggio.

Nota: È spesso preferibile evitare di inserire messaggi in una DLQ. Per informazioni sull'utilizzo e la prevenzione dei DLQ, consultare “Gestione delle code di messaggi non recapitabili” a pagina 156.

Informazioni su questa attività

Una *coda di messaggi non recapitabili* (DLQ), a volte indicata come *coda di messaggi non recapitabili*, è una coda di attesa per i messaggi che non possono essere consegnati alle relative code di destinazione. Ogni gestore code in una rete deve avere una DLQ associata.

I gestori code, gli agent del canale dei messaggi e le applicazioni possono inserire messaggi nella DLQ. Tutti i messaggi sulla DLQ devono avere come prefisso una struttura *dead-letter header* , MQDLH. I messaggi inseriti nella DLQ da un gestore code o da un agente del canale dei messaggi hanno sempre un MQDLH. Per le applicazioni che immettono messaggi nella DLQ, è necessario fornire un MQDLH.

Il campo *Motivo* della struttura MQDLH contiene un codice motivo che identifica il motivo per cui il messaggio si trova sulla DLQ.

In tutti gli ambienti di IBM MQ , deve essere presente una routine che viene eseguita regolarmente per elaborare i messaggi sulla DLQ. IBM MQ fornisce una routine predefinita, denominata *gestore code di messaggi non instradabili* (il gestore DLQ), richiamata utilizzando il comando **STRMQMDLQ** . Una *tabella regole* scritta dall'utente fornisce istruzioni al gestore DLQ, per l'elaborazione dei messaggi sulla DLQ. Ovvero, il gestore DLQ mette in corrispondenza i messaggi sulla DLQ con le voci nella tabella delle regole. Quando un messaggio DLQ corrisponde a una voce nella tabella delle regole, il gestore DLQ esegue l'azione associata a tale voce.

Procedura

- Richiama il gestore DLQ

  Utilizzare il comando **STRMQMDLQ** per richiamare il gestore DLQ. È possibile denominare la DLQ che si desidera elaborare e il gestore code che si desidera utilizzare in due modi:

  – Come parametri per **STRMQMDLQ** dal prompt dei comandi. Ad esempio:

  ```
  STRMQMDLQ UDLMSGQ(ABC1.DEAD.LETTER.QUEUE) SRCMBR(QRULE) SRCFILE(library/QTXTSRC)
  MQMNAME(MY.QUEUE.MANAGER)
  ```

  – Nella tabella delle regole. Ad esempio:

  ```
  INPUTQ(ABC1.DEAD.LETTER.QUEUE)
  ```

  Nota: La tabella delle regole è un membro all'interno di un file fisico di origine che può assumere qualsiasi nome.

  Gli esempi si applicano alla DLQ denominata `ABC1.DEAD.LETTER.QUEUE`, di proprietà del gestore code predefinito.

  Se non si specifica la DLQ o il gestore code come mostrato, il gestore code predefinito per l'installazione viene utilizzato insieme alla DLQ che appartiene a tale gestore code.

Concetti correlati

Code di messaggi non recapitabili

Attività correlate

Risoluzione dei problemi dei messaggi non recapitati

IBM i *La tabella delle regole del gestore DLQ su IBM i*

La tabella delle regole del gestore code di messaggi non recapitabili definisce il modo in cui il gestore DLQ elabora i messaggi che arrivano sulla DLQ IBM i .

La tabella delle regole del gestore DLQ definisce il modo in cui il gestore DLQ elabora i messaggi che arrivano sul DLQ. Ci sono due tipi di voce in una tabella di regole:

- La prima voce nella tabella, che è facoltativa, contiene *dati di controllo*.
- Tutte le altre voci nella tabella sono *regole* per il gestore DLQ da seguire. Ogni regola è composta da un *pattern* (una serie di caratteristiche del messaggio) rispetto al quale viene confrontato un messaggio e da un' *azione* da eseguire quando un messaggio sulla DLQ corrisponde al pattern specificato. In una tabella di regole deve essere presente almeno una regola.

Ciascuna voce nella tabella delle regole è composta da una o più parole chiave.

Dati di controllo

Questa sezione descrive le parole chiave che è possibile includere in una voce di dati di controllo in una tabella di regole gestore DLQ. Tieni presente quanto segue:

- Il valore predefinito per una parola chiave, se presente, è sottolineato.
- La linea verticale (|) separa le alternative. È possibile specificare solo uno di questi.
- Tutte le parole chiave sono facoltative.

INPUTQ (*QueueName*| '' (valore predefinito))

Il nome della DLQ che si desidera elaborare:

1. Qualsiasi valore UDLMSGQ (o *DFT) specificato come parametro per il comando **STRMQMDLQ** sostituisce qualsiasi valore INPUTQ nella tabella delle regole.
2. Se si specifica un valore UDLMSGQ vuoto come parametro per il comando **STRMQMDLQ** , viene utilizzato il valore INPUTQ nella tabella delle regole.

3. Se si specifica un valore UDLMSGQ vuoto come parametro per il comando **STRMQMDLQ** e un valore INPUTQ vuoto nella tabella delle regole, viene utilizzata la coda di messaggi non recapitabili predefinita di sistema.

INPUTQM (*QueueManagerNome*| '' (valore predefinito))
Il nome del gestore code proprietario della DLQ denominata nella parola chiave INPUTQ.

Se non si specifica un gestore code o se si specifica INPUTQM ('') nella tabella delle regole, il sistema utilizza il gestore code predefinito per l'installazione.

RETRYINT (*Intervallo*| 60 (predefinito))
L'intervallo, in secondi, con cui il gestore DLQ deve tentare di rielaborare i messaggi sulla DLQ che non è stato possibile elaborare al primo tentativo e per cui sono stati richiesti ripetuti tentativi. Per impostazione predefinita, l'intervallo tra i tentativi è 60 secondi.

WAIT (YES (predefinito) |NO|*nnn*)
Indica se il gestore DLQ deve attendere l'arrivo di ulteriori messaggi sulla DLQ quando rileva che non sono presenti ulteriori messaggi che può elaborare.

SÌ
Fa sì che il gestore DLQ attenda indefinitamente.

No
Causa la chiusura del gestore DLQ quando rileva che il DLQ è vuoto o non contiene alcun messaggio che può elaborare.

nnn
Fa in modo che il gestore DLQ attenda per *nnn* secondi l'arrivo di nuovo lavoro prima di terminare, dopo aver rilevato che la coda è vuota o non contiene alcun messaggio che può elaborare.

Specificare WAIT (YES) per le DLQ occupate e WAIT (NO) o WAIT (*nnn*) per le DLQ che hanno un basso livello di attività. Se il gestore DLQ può terminare, richiamarlo nuovamente utilizzando il trigger.

È possibile fornire il nome della DLQ come parametro di input al comando **STRMQMDLQ** , come alternativa all'inclusione dei dati di controllo nella tabella delle regole. Se viene specificato un valore sia nella tabella delle regole che nell'input del comando **STRMQMDLQ** , il valore specificato nel comando **STRMQMDLQ** ha la precedenza.

Nota: Se una voce di dati di controllo è inclusa nella tabella delle regole, deve essere la prima voce nella tabella.

IBM i *Regole DLQ (modelli e azioni) su IBM i*
Una descrizione dei modelli e delle azioni per ciascuna delle regole della coda di messaggi non instradabili IBM i .

Di seguito è riportata una regola di esempio da una tabella di regole del gestore DLQ:

```
PERSIST(MQPER_PERSISTENT) REASON (MQRC_PUT_INHIBITED) +
ACTION (RETRY) RETRY (3)
```

Questa regola indica al gestore DLQ di effettuare 3 tentativi per consegnare alla coda di destinazione qualsiasi messaggio persistente inserito nella DLQ perché MQPUT e MQPUT1 non erano consentiti.

Questa sezione descrive le parole chiave che è possibile includere in una regola. Tieni presente quanto segue:

- Il valore predefinito per una parola chiave, se presente, è sottolineato. Per la maggior parte delle parole chiave, il valore predefinito è * (asterisco), che corrisponde a qualsiasi valore.
- La linea verticale (|) separa le alternative. È possibile specificare solo uno di questi.
- Tutte le parole chiave tranne ACTION sono facoltative.

Questa sezione inizia con una descrizione delle parole chiave corrispondenti al modello (quelle con cui i messaggi sulla DLQ vengono messi in corrispondenza). Descrive quindi le parole chiave dell'azione (quelle che determinano in che modo il gestore DLQ deve elaborare un messaggio corrispondente).

IBM i *Modello DLQ - parole chiave corrispondenti su IBM i*

Le parole chiave corrispondenti al modello sono descritte in un esempio. Utilizzare queste parole chiave per specificare i valori a cui corrispondono i messaggi sulla coda di messaggi non instradabili IBM i . Tutte le parole chiave corrispondenti al modello sono facoltative.

APPLIDAT (*ApplIdentityData*|* (predefinito))
Il valore *ApplIdentityData* del messaggio sulla DLQ, specificato nel descrittore del messaggio, MQMD.

APPLNAME (*PutApplName*|* (predefinito))
Il nome dell'applicazione che ha emesso la chiamata MQPUT o MQPUT1 , come specificato nel campo *PutApplName* del descrittore del messaggio, MQMD, del messaggio sulla DLQ.

APPLTYPE (*PutApplTipo*|* (predefinito))
Il valore *PutApplType* specificato nel descrittore del messaggio, MQMD, del messaggio sulla DLQ.

DESTQ (*QueueName*|* (valore predefinito))
Il nome della coda messaggi a cui è destinato il messaggio.

DESTQM (Nome *QueueManager*|* (predefinito))
Il nome del gestore code per la coda messaggi a cui è destinato il messaggio.

FEEDBACK (*Feedback*|* (Predefinito))
Quando il valore *MsgType* è MQMT_REPORT, *Feedback* descrive la natura del report.

È possibile utilizzare nomi simbolici. Ad esempio, è possibile utilizzare il nome simbolico MQFB_COA per identificare i messaggi sulla DLQ che richiedono la conferma del relativo arrivo sulle code di destinazione.

FORMAT (*Formato*|* (predefinito))
Il nome utilizzato dal mittente del messaggio per descrivere il formato dei dati del messaggio.

MSGTYPE (*MsgType*|* (valore predefinito))
Il tipo di messaggio del DLQ.

È possibile utilizzare nomi simbolici. Ad esempio, è possibile utilizzare il nome simbolico MQMT_REQUEST per identificare quei messaggi sulla DLQ che richiedono risposte.

PERSIST (*Persistenza*|* (predefinito)
Il valore di persistenza del messaggio. La persistenza di un messaggio determina se sopravvive ai riavvii del gestore code.

È possibile utilizzare nomi simbolici. Ad esempio, è possibile utilizzare il nome simbolico MQPER_PERSISTENT per identificare quei messaggi nella DLQ che sono persistenti.

REASON (*ReasonCode*|* (predefinito)
Il codice di errore che descrive il motivo per cui il messaggio è stato inserito nella DLQ.

È possibile utilizzare nomi simbolici. Ad esempio, è possibile utilizzare il nome simbolico MQRC_Q_FULL per identificare i messaggi collocati nella DLQ perché le code di destinazione erano piene.

REPLYQ (*QueueName*|* (predefinito))
Il nome della coda di risposta specificato nel descrittore del messaggio, MQMD, del messaggio sulla DLQ.

REPLYQM (*QueueManagerNome*|* (predefinito))
Il nome del gestore code della coda di risposta specificato nella parola chiave REPLYQ.

USERID (*UserIdentifier*|* (predefinito))
L'ID utente dell'utente che ha creato il messaggio sulla DLQ, come specificato nel descrittore del messaggio, MQMD.

IBM i *Parole chiave di azione DLQ su IBM i*

Utilizzare queste parole chiave di azione della coda di messaggi non recapitabili per determinare in che modo viene elaborato un messaggio corrispondente sulla coda di messaggi non recapitabili IBM i .

AZIONE (DISCARD | IGNORE | RETRY | FWD)
L'azione eseguita per qualsiasi messaggio sul DLQ che corrisponde al pattern definito in questa regola.

DISCARD
Causa la cancellazione del messaggio dalla DLQ.

IGNORE
Fa sì che il messaggio venga conservato nella DLQ.

RIPROVA
Fa in modo che il gestore DLQ tenti nuovamente di inserire il messaggio nella coda di destinazione.

FWD
Il gestore DLQ inoltra il messaggio alla coda denominata sulla parola chiave FWDQ.

È necessario specificare la parola chiave ACTION. Il numero di tentativi effettuati per implementare un'azione è regolato dalla parola chiave RETRY. La parola chiave RETRYINT dei dati di controllo controlla l'intervallo tra i tentativi.

FWDQ (*QueueName*| & DESTQ | & REPLYQ)
Il nome della coda messaggi a cui viene inoltrato il messaggio quando si seleziona la parola chiave ACTION.

QueueName
Il nome di una coda messaggi. FWDQ ('') non è valido.

& DESTQ
Prendere il nome della coda dal campo *DestQName* nella struttura MQDLH.

& REPLYQ
Prendere il nome della coda dal campo *ReplyToQ* nel descrittore del messaggio, MQMD.

È possibile specificare REPLYQ (? *) nel modello di messaggio per evitare messaggi di errore, quando una regola che specifica FWDQ (& REPLYQ) corrisponde a un messaggio con un campo *ReplyToQ* vuoto.

FWDQM (*QueueManagerNome*| & DESTQM | & REPLYQM | '' (valore predefinito))
Il gestore code della coda a cui viene inoltrato un messaggio.

QueueManagerName
Il nome del gestore code per la coda a cui viene inoltrato il messaggio quando si seleziona la parola chiave ACTION (FWD).

& DESTQM
Prendere il nome gestore code dal campo *DestQMgrNome* nella struttura MQDLH.

& REPLYQM
Prendere il nome del gestore code dal campo *ReplyToReplyTo* nel descrittore del messaggio, MQMD.

' '
FWDQM (''), che è il valore predefinito, identifica il gestore code locale.

HEADER (YES (predefinito) |NO)
Se MQDLH deve rimanere su un messaggio per cui è richiesto ACTION (FWD). Per impostazione predefinita, MQDLH rimane sul messaggio. La parola chiave HEADER non è valida per azioni diverse da FWD.

PUTAUT (DEF (predefinito) | CTX)
L'autorità con cui i messaggi devono essere inseriti dal gestore DLQ:

DEF
Inserisce i messaggi con l'autorità del gestore DLQ stesso.

CTX
Fa in modo che i messaggi vengano inseriti con l'autorizzazione dell'ID utente nel contesto del messaggio. È necessario essere autorizzati ad assumere l'identità di altri utenti, se si specifica PUTAUT (CTX).

RETRY (*RetryCount*|1 (predefinito)
Il numero di volte, nell'intervallo compreso tra 1 e 999.999.999, per tentare un'azione (all'intervallo specificato nella parola chiave RETRYINT dei dati di controllo).

Nota: Il conteggio dei tentativi effettuati dal gestore DLQ per implementare una particolare regola è specifico per l'istanza corrente del gestore DLQ; il conteggio non persiste durante i riavvii. Se si riavvia il gestore DLQ, il numero di tentativi effettuati per applicare una regola viene reimpostato su zero.

IBM i *Convenzioni della tabella regole DLQ su IBM i*

La tabella delle regole della coda di messaggi non recapitabili IBM i deve rispettare convenzioni specifiche relative a sintassi, struttura e contenuto.

- Una tabella di regole deve contenere almeno una regola.
- Le parole chiave possono essere presenti in qualsiasi ordine.
- Una parola chiave può essere inclusa una sola volta in qualsiasi regola.
- Le parole chiave non sono sensibili al maiuscolo / minuscolo.
- Una parola chiave e il relativo valore di parametro devono essere separati da altre parole chiave da almeno uno spazio vuoto o da una virgola.
- Qualsiasi numero di spazi vuoti può verificarsi all'inizio o alla fine di una regola e tra parole chiave, punteggiatura e valori.
- Ogni regola deve iniziare su una nuova riga.
- Per la portabilità, la lunghezza significativa di una riga non deve superare i 72 caratteri.
- Utilizzare il segno più (+) come ultimo carattere non vuoto su una riga per indicare che la regola continua dal primo carattere non vuoto nella riga successiva. Utilizzare il segno meno (-) come ultimo carattere non vuoto su una riga per indicare che la regola continua dall'inizio della riga successiva. I caratteri di continuazione possono verificarsi all'interno di parole chiave e parametri.

  Ad esempio:

  ```
  APPLNAME('ABC+
  D')
  ```

  risultati in 'ABCD'.

  ```
  APPLNAME('ABC-
  D')
  ```

  risultati in ' ABC D'.
- Le righe di commento, che iniziano con un asterisco (*), possono trovarsi in qualsiasi punto della tabella delle regole.
- Le righe vuote vengono ignorate.
- Ogni voce nella tabella delle regole del programma di gestione DLQ comprende una o più parole chiave e i relativi parametri associati. I parametri devono seguire queste regole di sintassi:
  - Ogni valore di parametro deve includere almeno un carattere significativo. Le virgolette di delimitazione nei valori racchiusi tra virgolette non sono considerate significative. Ad esempio, questi parametri sono validi:

`FORMAT('ABC')`	3 caratteri significativi
`FORMAT(ABC)`	3 caratteri significativi
`FORMAT('A')`	1 carattere significativo
`FORMAT(A)`	1 carattere significativo

`FORMAT(' ')`	1 carattere significativo

Questi parametri non sono validi perché non contengono caratteri significativi:

```
FORMAT('')
FORMAT( )
FORMAT()
FORMAT
```

- sono supportati. È possibile utilizzare il punto interrogativo (?) al posto di qualsiasi carattere singolo, ad eccezione di uno spazio vuoto finale. È possibile utilizzare l'asterisco (*) al posto di zero o più caratteri adiacenti. L'asterisco (*) e il punto interrogativo (?) sono ***sempre*** interpretati come caratteri jolly nei valori dei parametri.
- Non è possibile includere caratteri jolly nei parametri di queste parole chiave: ACTION, HEADER, RETRY, FWDQ, FWDQM e PUTAUT.
- Gli spazi vuoti finali nei valori di parametro e nei campi corrispondenti nel messaggio sulla DLQ, non sono significativi quando si eseguono corrispondenze di caratteri jolly. Tuttavia, gli spazi vuoti iniziali e incorporati all'interno delle stringhe tra virgolette sono significativi per le corrispondenze di caratteri jolly.
- I parametri numerici non possono includere il carattere jolly punto interrogativo (?). È possibile includere l'asterisco (*) al posto di un intero parametro numerico, ma l'asterisco non può essere incluso come parte di un parametro numerico. Ad esempio, questi sono parametri numerici validi:

`MSGTYPE(2)`	Sono idonei solo i messaggi di risposta
`MSGTYPE(*)`	Qualsiasi tipo di messaggio è idoneo
`MSGTYPE('*')`	Qualsiasi tipo di messaggio è idoneo

  Tuttavia, `MSGTYPE('2*')` non è valido, poiché include un asterisco (*) come parte di un parametro numerico.
- I parametri numerici devono essere compresi tra 0 e 999 999 999. Se il valore del parametro è compreso in questo intervallo, viene accettato, anche se non è attualmente valido nel campo a cui si riferisce la parola chiave. È possibile utilizzare nomi simbolici per i parametri numerici.
- Se un valore stringa è più breve del campo in MQDLH o MQMD a cui si riferisce la parola chiave, il valore viene riempito con spazi vuoti fino alla lunghezza del campo. Se il valore, esclusi gli asterischi, è più lungo del campo, viene diagnosticato un errore. Ad esempio, questi sono tutti valori di stringa validi per un campo di 8 caratteri:

`'ABCDEFGH'`	8 caratteri
`'A*C*E*G*I'`	5 caratteri esclusi gli asterischi
`'*A*C*E*G*I*K*M*O*'`	8 caratteri esclusi gli asterischi

- Le stringhe che contengono spazi, caratteri minuscoli o caratteri speciali diversi da punto (.), barra (?), sottolineatura (_) e segno di percentuale (%) devono essere racchiuse tra virgolette singole. I caratteri minuscoli non racchiusi tra virgolette vengono piegati in maiuscolo. Se la stringa include una virgoletta, è necessario utilizzare due virgolette singole per indicare sia l'inizio che la fine della virgoletta. Quando viene calcolata la lunghezza della stringa, ogni ricorrenza delle virgolette viene conteggiata come un singolo carattere.

IBM i *Modalità di elaborazione della tabella di regole DLQ su IBM i*

Il gestore code di messaggi non instradabili ricerca nella tabella delle regole una regola con un modello che corrisponde a un messaggio nella coda di messaggi non instradabili IBM i .

La ricerca inizia con la prima regola nella tabella e continua in modo sequenziale attraverso la tabella. Quando viene trovata una regola con un modello corrispondente, la tabella delle regole tenta l'azione da

tale regola. Il gestore DLQ incrementa il numero di tentativi per una regola di 1 ogni volta che tenta di applicare tale regola. Se il primo tentativo ha esito negativo, il tentativo viene ripetuto fino a quando il numero di tentativi effettuati non corrisponde al numero specificato nella parola chiave RETRY. Se tutti i tentativi hanno esito negativo, il gestore DLQ ricerca la regola di corrispondenza successiva nella tabella.

Questo processo viene ripetuto per le regole di corrispondenza successive fino a quando un'azione ha esito positivo. Quando ogni regola di corrispondenza è stata tentata il numero di volte specificato nella relativa parola chiave RETRY e tutti i tentativi hanno avuto esito negativo, viene assunto ACTION (IGNORE). ACTION (IGNORE) viene anche assunto se non viene trovata alcuna regola corrispondente.

Nota:

1. I modelli di regole corrispondenti vengono ricercati solo per i messaggi sulla DLQ che iniziano con un MQDLH. I messaggi che non iniziano con MQDLH vengono riportati periodicamente come in errore e rimangono nella DLQ per un periodo di tempo indefinito.
2. Tutte le parole chiave del modello possono essere predefinite, in modo che una regola possa essere composta solo da un'azione. Tenere presente, tuttavia, che le regole di sola azione vengono applicate a tutti i messaggi nella coda che hanno MQDLH e che non sono già stati elaborati in conformità con altre regole nella tabella.
3. La tabella delle regole viene convalidata all'avvio del programma di gestione DLQ e gli errori vengono contrassegnati in quel momento. (I messaggi di errore emessi dal gestore DLQ sono descritti in Messaggi e codici di errore.) È possibile apportare delle modifiche alla tabella delle regole in qualsiasi momento, ma tali modifiche non diventano effettive fino a quando il gestore DLQ non viene riavviato.
4. Il gestore DLQ non modifica il contenuto dei messaggi, di MQDLH o del descrittore del messaggio. Il gestore DLQ inserisce sempre i messaggi in altre code con l'opzione **MQPMO_PASS_ALL_CONTEXT**.
5. Gli errori di sintassi consecutivi nella tabella delle regole potrebbero non essere riconosciuti, poiché la convalida della tabella delle regole elimina la creazione di errori ripetitivi.
6. Il gestore DLQ apre il DLQ con l'opzione **MQOO_INPUT_AS_Q_DEF** .
7. È possibile eseguire simultaneamente più istanze del gestore DLQ sulla stessa coda, utilizzando la stessa tabella delle regole. Tuttavia, è più usuale che ci sia una relazione uno - a - uno tra una DLQ e un gestore DLQ.

IBM i *Una tabella di regole del gestore DLQ di esempio su IBM i*

Codice di esempio per una tabella di regole del gestore code di messaggi non recapitabili su IBM i. Questa tabella di regole di esempio contiene una singola immissione dati di controllo e diverse regole.

```
*********************************************************************
*    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)
```

Verifica dell'elaborazione di tutti i messaggi DLQ

Il gestore code di messaggi non instradabili conserva un record di tutti i messaggi sulla DLQ che sono stati visualizzati ma non rimossi. Assicurarsi che la DLQ contenga il minor numero di messaggi possibile.

Informazioni su questa attività

Se si utilizza il gestore DLQ come filtro per estrarre un piccolo sottoinsieme di messaggi dal DLQ, il gestore DLQ conserva ancora un record di tali messaggi sul DLQ che non ha elaborato. Inoltre, il gestore DLQ non può garantire che vengano visualizzati nuovi messaggi in arrivo sul DLQ, anche se il DLQ è definito come FIFO (first - in - first - out). Se la coda non è vuota, la DLQ viene periodicamente riesaminata per controllare tutti i messaggi.

Per questi motivi, è necessario assicurarsi che la DLQ contenga il minor numero di messaggi possibile. Se i messaggi che non possono essere eliminati o inoltrati ad altre code (per qualsiasi motivo) possono accumularsi sulla coda, il carico di lavoro del gestore DLQ aumenta e il DLQ stesso rischia di riempirsi.

Per abilitare il gestore DLQ a svuotarlo, effettuare le seguenti operazioni:

Procedura

- Per i messaggi che altrimenti verrebbero ignorati, utilizzare un'azione che sposta i messaggi su un'altra coda.

  Provare a non utilizzare il comando **ACTION (IGNORE)** , che lascia i messaggi sulla DLQ - e ricordare che **ACTION (IGNORE)** viene utilizzato per i messaggi non esplicitamente indirizzati da altre regole nella tabella. Utilizzare invece un'azione che sposta i messaggi in un'altra coda. Ad esempio:

  ```
  ACTION (FWD) FWDQ (IGNORED.DEAD.QUEUE) HEADER (YES)
  ```

- Rendere la regola finale nella tabella un "catch-all" per elaborare i messaggi che non sono stati indirizzati da regole precedenti nella tabella.

  Ad esempio, la regola finale nella tabella potrebbe essere simile alla seguente:

  ```
  ACTION (FWD) FWDQ (REALLY.DEAD.QUEUE) HEADER (YES)
  ```

  Questo inoltra i messaggi che rientrano nella regola finale nella tabella alla coda REALLY.DEAD.QUEUE, dove possono essere elaborati manualmente. Se non si dispone di una regola di questo tipo, è probabile che i messaggi rimangano nella DLQ per un periodo di tempo indefinito.

Utilizzo degli argomenti di gestione

Utilizzare i comandi MQSC per gestire gli argomenti di amministrazione.

Consultare Comandi MQSC per informazioni dettagliate su questi comandi.

Concetti correlati

Oggetti argomento di gestione

Attività correlate

“Definizione di un argomento di amministrazione” a pagina 176
Utilizzare il comando MQSC **DEFINE TOPIC** per creare un argomento di gestione. Quando si definisce un argomento di gestione, è possibile impostare facoltativamente ogni attributo dell'argomento.

“Visualizzazione degli attributi dell'oggetto argomento di gestione” a pagina 177
Utilizzare il comando MQSC **DISPLAY TOPIC** per visualizzare un oggetto argomento di gestione.

“Modifica degli attributi dell'argomento di amministrazione” a pagina 178
È possibile modificare gli attributi dell'argomento in due modi, utilizzando il comando **ALTER TOPIC** o il comando **DEFINE TOPIC** con l'attributo **REPLACE** .

“Copia di una definizione di argomento di gestione” a pagina 179
È possibile copiare una definizione di argomento utilizzando l'attributo LIKE sul comando **DEFINE** .

“Eliminazione di una definizione di argomento di gestione” a pagina 179
È possibile utilizzare il comando MQSC **DELETE TOPIC** per eliminare un argomento di gestione.

Definizione di un argomento di amministrazione

Utilizzare il comando MQSC **DEFINE TOPIC** per creare un argomento di gestione. Quando si definisce un argomento di gestione, è possibile impostare facoltativamente ogni attributo dell'argomento.

Prima di iniziare

Nota: L'esempio in questa attività richiede l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Informazioni su questa attività

Qualsiasi attributo dell'argomento non esplicitamente impostato viene ereditato dall'argomento di gestione predefinito, SYSTEM.DEFAULT.TOPIC, che è stato creato quando è stata installata l'installazione del sistema.

Esempio

Ad esempio, il comando **DEFINE TOPIC** che segue, definisce un argomento denominato ORANGE.TOPIC con queste caratteristiche:

- Si risolve nella stringa di argomenti ORANGE. Per informazioni su come utilizzare le stringhe argomento, consultare Combinazione di stringhe argomento.
- Qualsiasi attributo impostato su ASPARENT utilizza l'attributo come definito dall'argomento principale di questo argomento. Questa operazione viene ripetuta nella struttura ad albero degli argomenti fino all'argomento principale, SYSTEM.BASE.TOPIC trovato. Per ulteriori informazioni, consultare Alberi degli argomenti.

```
DEFINE TOPIC (ORANGE.TOPIC) +
TOPICSTR (ORANGE) +
DEFPRTY(ASPARENT) +
NPMSGDLV(ASPARENT)
```

Nota:

- Tranne che per il valore della stringa di argomenti, tutti i valori di attributo visualizzati sono i valori predefiniti. Sono mostrati qui solo come un'illustrazione. È possibile ometterli se si è certi che i valori predefiniti sono quelli desiderati o non sono stati modificati. Consultare anche “Visualizzazione degli attributi dell'oggetto argomento di gestione” a pagina 177.
- Se si dispone già di un argomento di gestione sullo stesso gestore code con il nome ORANGE.TOPIC, questo comando non riesce. Utilizzare l'attributo REPLACE se si desidera sovrascrivere la definizione esistente di un argomento, ma consultare anche “Modifica degli attributi dell'argomento di amministrazione” a pagina 178.

Riferimenti correlati

DEFINISCI ARGOMENTO

Visualizzazione degli attributi dell'oggetto argomento di gestione

Utilizzare il comando MQSC **DISPLAY TOPIC** per visualizzare un oggetto argomento di gestione.

Prima di iniziare

Nota: Gli esempi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Esempio

Questo comando visualizza tutti gli argomenti:

```
DISPLAY TOPIC(ORANGE.TOPIC)
```

È possibile visualizzare selettivamente gli attributi specificandoli singolarmente con il comando **DISPLAY TOPIC** . Ad esempio:

```
DISPLAY TOPIC(ORANGE.TOPIC) +
TOPICSTR +
DEFPRTY +
NPMSGDLV
```

Questo comando visualizza i tre attributi specificati:

```
AMQ8633: Display topic details.
     TOPIC(ORANGE.TOPIC)                                     TYPE(LOCAL)
     TOPICSTR(ORANGE)                                        DEFPRTY(ASPARENT)
     NPMSGDLV(ASPARENT)
```

Per visualizzare i valori ASPARENT dell'argomento come vengono utilizzati al runtime, utilizzare il comando **DISPLAY TPSTATUS** . Ad esempio, utilizzare:

```
DISPLAY TPSTATUS(ORANGE) DEFPRTY NPMSGDLV
```

Il comando visualizza i seguenti dettagli:

```
AMQ8754: Display topic status details.
TOPICSTR(ORANGE)            DEFPRTY(0)
NPMSGDLV(ALLAVAIL)
```

Quando si definisce un argomento di gestione, vengono utilizzati tutti gli attributi non specificati esplicitamente dall'argomento di gestione predefinito, denominato SYSTEM.DEFAULT.TOPIC. Per visualizzare questi attributi predefiniti, utilizzare il seguente comando:

```
DISPLAY TOPIC (SYSTEM.DEFAULT.TOPIC)
```

Riferimenti correlati
VISUALIZZA ARGOMENTO
VISUALIZZA TPSTATUS

Modifica degli attributi dell'argomento di amministrazione

È possibile modificare gli attributi dell'argomento in due modi, utilizzando il comando **ALTER TOPIC** o il comando **DEFINE TOPIC** con l'attributo **REPLACE** .

Prima di iniziare

Nota: Gli esempi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Esempio

Se, ad esempio, si desidera modificare la priorità predefinita dei messaggi consegnati ad un argomento denominato ORANGE.TOPIC, deve essere 5, utilizzare uno dei seguenti comandi:

- Utilizzando il comando **ALTER** :

  ```
  ALTER TOPIC(ORANGE.TOPIC) DEFPRTY(5)
  ```

  Questo comando modifica un singolo attributo, quello della priorità predefinita del messaggio consegnato a questo argomento in 5; tutti gli altri attributi rimangono gli stessi.
- Utilizzando il comando **DEFINE** :

  ```
  DEFINE TOPIC(ORANGE.TOPIC) DEFPRTY(5) REPLACE
  ```

  Questo comando modifica la priorità predefinita dei messaggi consegnati a questo argomento. A tutti gli altri attributi vengono assegnati valori predefiniti.

Se si modifica la priorità dei messaggi inviati a questo argomento, i messaggi esistenti non vengono influenzati. Qualsiasi nuovo messaggio, tuttavia, utilizza la priorità specificata se non fornita dall'applicazione di pubblicazione.

Riferimenti correlati

ALTER TOPIC

VISUALIZZA ARGOMENTO

Copia di una definizione di argomento di gestione

È possibile copiare una definizione di argomento utilizzando l'attributo LIKE sul comando **DEFINE** .

Prima di iniziare

Nota: Gli esempi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Esempio

Il seguente comando crea un argomento, MAGENTA.TOPIC, con gli stessi attributi dell'argomento originale, ORANGE.TOPIC, piuttosto che quelli dell'argomento di gestione predefinito del sistema. Immettere il nome dell'argomento da copiare esattamente come è stato immesso quando è stato creato l'argomento. Se il nome contiene caratteri minuscoli, racchiuderlo tra virgolette singole.

```
DEFINE TOPIC (MAGENTA.TOPIC) +
LIKE (ORANGE.TOPIC)
```

È anche possibile utilizzare questo modulo del comando **DEFINE** per copiare una definizione di argomento, ma apportare modifiche agli attributi dell'originale. Ad esempio:

```
DEFINE TOPIC(BLUE.TOPIC) +
TOPICSTR(BLUE) +
LIKE(ORANGE.TOPIC)
```

È anche possibile copiare gli attributi dell'argomento BLUE.TOPIC all'argomento GREEN.TOPIC e specificare che quando le pubblicazioni non possono essere consegnate alla coda del sottoscrittore corretta, non vengono inserite nella coda di messaggi non recapitabili. Ad esempio:

```
DEFINE TOPIC(GREEN.TOPIC) +
TOPICSTR(GREEN) +
LIKE(BLUE.TOPIC) +
USEDLQ(NO)
```

Riferimenti correlati

DEFINISCI ARGOMENTO

Eliminazione di una definizione di argomento di gestione

È possibile utilizzare il comando MQSC **DELETE TOPIC** per eliminare un argomento di gestione.

Prima di iniziare

Nota: L'esempio in questa attività richiede l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Esempio

```
DELETE TOPIC(ORANGE.TOPIC)
```

Le applicazioni non saranno più in grado di aprire l'argomento per la pubblicazione o effettuare nuove sottoscrizioni utilizzando il nome oggetto, ORANGE.TOPIC. Le applicazioni di pubblicazione che hanno l'argomento aperto possono continuare a pubblicare la stringa di argomenti risolta. Le sottoscrizioni già effettuate a questo argomento continuano a ricevere pubblicazioni dopo che l'argomento è stato eliminato.

Le applicazioni che non fanno riferimento a questo oggetto argomento ma che utilizzano la stringa di argomenti risolta rappresentata da questo oggetto argomento, 'ORANGE' in questo esempio, continuano a funzionare. In questo caso, ereditano le proprietà da un oggetto argomento più in alto nella struttura ad albero degli argomenti. Per ulteriori informazioni, consultare Alberi degli argomenti.

Riferimenti correlati

Elimina argomento

Utilizzo delle sottoscrizioni

Utilizzare i comandi MQSC per gestire le sottoscrizioni.

Informazioni su questa attività

Le sottoscrizioni possono essere di tre tipi, definiti nell'attributo **SUBTYPE** :

ADMIN
Definito amministrativamente da un utente.

PROXY
Una sottoscrizione creata internamente per instradare le pubblicazioni tra i gestori code.

API
Creato in modo programmatico, ad esempio, utilizzando la chiamata MQI MQSUB.

Consultare Comandi MQSC per informazioni dettagliate su questi comandi.

Definizione di una sottoscrizione di gestione

Utilizzare il comando MQSC **DEFINE SUB** per creare una sottoscrizione di gestione. È anche possibile utilizzare il valore predefinito definito nella definizione della sottoscrizione locale predefinita. Oppure, è possibile modificare le caratteristiche della sottoscrizione da quelle della sottoscrizione locale predefinita, SYSTEM.DEFAULT.SUB che è stato creato quando il sistema è stato installato.

Prima di iniziare

Nota: Gli esempi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Esempio

Il seguente comando **DEFINE SUB** definisce una sottoscrizione denominata ORANGE con queste caratteristiche:

- Sottoscrizione durevole, che indica che persiste al riavvio del gestore code, con scadenza illimitata.
- Ricevere le pubblicazioni effettuate sulla stringa di argomenti ORANGE, con le priorità dei messaggi impostate dalle applicazioni di pubblicazione.
- Le pubblicazioni distribuite per questa sottoscrizione vengono inviate alla coda locale SUBQ, questa coda deve essere definita prima della definizione della sottoscrizione.

```
DEFINE SUB (ORANGE) +
TOPICSTR (ORANGE) +
DESTCLAS (PROVIDED) +
DEST (SUBQ) +
EXPIRY (UNLIMITED) +
PUBPRTY (ASPUB)
```

Nota:

- Il nome della sottoscrizione e della stringa di argomenti non devono corrispondere.
- Tranne che per i valori della destinazione e della stringa argomento, tutti i valori di attributo visualizzati sono i valori predefiniti. Sono mostrati qui solo come un'illustrazione. È possibile ometterli se si è certi che i valori predefiniti sono quelli desiderati o non sono stati modificati. Consultare anche “Visualizzazione degli attributi delle sottoscrizioni” a pagina 181.
- Se si dispone già di una sottoscrizione locale sullo stesso gestore code con il nome ORANGE, questo comando non riesce. Utilizzare l'attributo **REPLACE** se si desidera sovrascrivere la definizione esistente di una coda, ma consultare anche “Modifica degli attributi della sottoscrizione locale” a pagina 182.
- Se la coda SUBQ non esiste, questo comando ha esito negativo.

Riferimenti correlati

DEFINE SUB

Visualizzazione degli attributi delle sottoscrizioni

È possibile utilizzare il comando **DISPLAY SUB** per visualizzare gli attributi configurati di qualsiasi sottoscrizione nota al gestore code.

Prima di iniziare

Nota: Gli esempi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Esempio

```
DISPLAY SUB(ORANGE)
```

È possibile visualizzare in modo selettivo gli attributi specificandoli singolarmente. Ad esempio:

```
DISPLAY SUB(ORANGE) +
        SUBID +
        TOPICSTR +
        DURABLE
```

Questo comando visualizza i tre attributi specificati nel modo seguente:

```
AMQ8096: IBM MQ subscription inquired.
     SUBID(414D5120414141202020202020202020EE921E4E20002A03)
     SUB(ORANGE)                                                    TOPICSTR(ORANGE)
     DURABLE(YES)
```

TOPICSTR è la stringa argomento risolta su cui questo sottoscrittore è operativo. Quando una richiesta viene definita per utilizzare un oggetto argomento, la stringa di argomento da tale oggetto viene utilizzata come prefisso alla stringa argomento fornita durante l'esecuzione della richiesta. SUBID è un identificativo univoco assegnato dal gestore code quando viene creata una sottoscrizione. Questo è un attributo utile da visualizzare perché alcuni nomi di sottoscrizioni potrebbero essere lunghi o in una serie di caratteri diversa per cui potrebbero diventare impraticabili.

Un metodo alternativo per visualizzare le sottoscrizioni consiste nell'utilizzare il SUBID:

```
DISPLAY SUB +
        SUBID(414D5120414141202020202020202020EE921E4E20002A03) +
        TOPICSTR +
        DURABLE
```

Questo comando fornisce lo stesso output di prima:

```
AMQ8096: IBM MQ subscription inquired.
 SUBID(414D5120414141202020202020202020EE921E4E20002A03)
   SUB(ORANGE)                                                          TOPICSTR(ORANGE)
   DURABLE(YES)
```

Le sottoscrizioni proxy su un gestore code non vengono visualizzate per default. Per visualizzarli specificare un **SUBTYPE** di PROXY o ALL.

È possibile utilizzare il comando DISPLAY SBSTATUS per visualizzare gli attributi Runtime. Ad esempio, utilizzare il comando:

```
DISPLAY SBSTATUS(ORANGE) NUMMSGS
```

Viene visualizzato il seguente output:

```
AMQ8099: IBM MQ subscription status inquired.
   SUB(ORANGE)
   SUBID(414D5120414141202020202020202020EE921E4E20002A03)
   NUMMSGS(0)
```

Quando si definisce una sottoscrizione di gestione, vengono utilizzati tutti gli attributi non specificati esplicitamente dalla sottoscrizione predefinita, denominata SYSTEM.DEFAULT.SUB. Per visualizzare questi attributi predefiniti, utilizzare il seguente comando:

```
DISPLAY SUB (SYSTEM.DEFAULT.SUB)
```

Riferimenti correlati

VISUALIZZA SECONDARIO

Modifica degli attributi della sottoscrizione locale

È possibile modificare gli attributi di sottoscrizione in due modi, utilizzando il comando **ALTER SUB** o il comando **DEFINE SUB** con l'attributo **REPLACE** .

Prima di iniziare

Nota: Gli esempi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Esempio

Se si desidera modificare la priorità dei messaggi consegnati a una sottoscrizione denominata ORANGE in modo che sia 5, utilizzare uno dei seguenti comandi:

- Utilizzando il comando **ALTER** :

  ```
  ALTER SUB(ORANGE) PUBPRTY(5)
  ```

  Questo comando modifica un singolo attributo, quello della priorità dei messaggi consegnati a questa sottoscrizione a 5; tutti gli altri attributi rimangono gli stessi.

- Utilizzando il comando **DEFINE** :

```
DEFINE SUB(ORANGE) PUBPRTY(5) REPLACE
```

  Questo comando modifica non solo la priorità dei messaggi consegnati a questa sottoscrizione, ma anche tutti gli altri attributi a cui vengono assegnati valori predefiniti.

Se si modifica la priorità dei messaggi inviati a questa sottoscrizione, i messaggi esistenti non vengono influenzati. Tutti i nuovi messaggi, tuttavia, hanno la priorità specificata.

Riferimenti correlati

MODIFICA SUB

DEFINE SUB

Copia di una definizione di sottoscrizione locale

È possibile copiare una definizione di sottoscrizione utilizzando l'attributo **LIKE** sul comando **DEFINE** .

Prima di iniziare

Nota: Gli esempi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Esempio

```
DEFINE SUB(BLUE) +
       LIKE(ORANGE)
```

È inoltre possibile copiare gli attributi del REAL secondario nel THIRD.SUBe specificare che **correlID** delle pubblicazioni fornite è THIRD, piuttosto che i publisher **correlID**. Ad esempio:

```
DEFINE SUB(THIRD.SUB) +
       LIKE(BLUE) +
       DESTCORL(ORANGE)
```

Riferimenti correlati

DEFINE SUB

Eliminazione di una sottoscrizione locale

È possibile utilizzare il comando MQSC **DELETE SUB** per eliminare una sottoscrizione locale.

Prima di iniziare

Nota: Gli esempi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Esempio

```
DELETE SUB(ORANGE)
```

È anche possibile eliminare una sottoscrizione utilizzando il SUBID:

```
DELETE SUB SUBID(414D5120414141202020202020202020EE921E4E20002A03)
```

Riferimenti correlati
ELIMINA SUB

Controllo dei messaggi su una sottoscrizione

Quando una sottoscrizione viene definita, viene associata a una coda. I messaggi pubblicati corrispondenti a questa sottoscrizione vengono inseriti in questa coda. Utilizzare i comandi MQSC per controllare i messaggi attualmente accodati per una sottoscrizione.

Prima di iniziare

Nota: I passi in questa attività richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Informazioni su questa attività

Tenere presente che i seguenti comandi MQSC mostrano solo le sottoscrizioni che hanno ricevuto messaggi.

Per controllare i messaggi attualmente accodati per una sottoscrizione, effettuare le seguenti operazioni:

Procedura

1. Per controllare i messaggi accodati per un tipo sottoscrizione DISPLAY SBSTATUS(*sub_name*) NUMMSGS, consultare "Visualizzazione degli attributi delle sottoscrizioni" a pagina 181.
2. Se il valore di **NUMMSGS** è maggiore di zero, identificare la coda associata alla sottoscrizione immettendo DISPLAY SUB(*sub_name*)DEST.
3. Utilizzando il nome della coda restituita è possibile visualizzare i messaggi seguendo la tecnica descritta in "Ricerca delle code con il programma di esempio" a pagina 145.

Riferimenti correlati
VISUALIZZAZIONE STATO SB

Gestione dei servizi

Gli oggetti di servizio sono un mezzo con cui è possibile gestire ulteriori processi come parte di un gestore code. Con i servizi, è possibile definire programmi che vengono avviati e arrestati all'avvio e alla fine del gestore code. I servizi IBM MQ vengono sempre avviati con l'ID utente dell'utente che ha avviato il gestore code.

Informazioni su questa attività

Gli oggetti di servizio possono essere uno dei seguenti tipi:

Server
Un server è un oggetto di servizio con il parametro **SERVTYPE** specificato come SERVER. Un oggetto servizio server è la definizione di un programma eseguito quando viene avviato un gestore code specificato. Gli oggetti di servizio del server definiscono i programmi che generalmente vengono eseguiti per un lungo periodo di tempo. Ad esempio, un oggetto servizio del server può essere utilizzato per eseguire un processo di controllo dei trigger, come **runmqtrm**.

Solo un'istanza di un oggetto servizio server può essere eseguita simultaneamente. Lo stato degli oggetti del servizio server in esecuzione può essere monitorato utilizzando il comando MQSC, **DISPLAY SVSTATUS**.

Comando
Un comando è un oggetto di servizio che ha il parametro **SERVTYPE** specificato come COMANDO. Gli oggetti servizio comandi sono simili agli oggetti servizio server, tuttavia più istanze di un oggetto servizio comandi possono essere eseguite simultaneamente e il loro stato non può essere monitorato utilizzando il comando MQSC **DISPLAY SVSTATUS**.

Se il comando MQSC, **STOP SERVICE**, viene eseguito, non viene eseguito alcun controllo per determinare se il programma avviato dal comando MQSC, **START SERVICE**, è ancora attivo prima di arrestare il programma.

Riferimenti correlati

Definisci servizio

VISUALIZZA SVSTATUS

Avvia servizio

Arresta servizio

Definizione di un oggetto servizio

Definire un oggetto servizio con il comando MQSC **DEFINE SERVICE**.

Prima di iniziare

Nota: Questa attività richiede l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Procedura

- Definire un oggetto di servizio con il comando MQSC **DEFINE SERVICE**.

  Gli attributi da definire sono i seguenti:

  SERVTYPE
  Definisce il tipo di oggetto servizio. I valori possibili sono i seguenti:

  SERVER
  Un oggetto servizio server.

  È possibile eseguire una sola istanza di un oggetto servizio server alla volta. Lo stato degli oggetti del servizio server può essere monitorato utilizzando il comando MQSC, **DISPLAY SVSTATUS**.

  COMANDO
  Un oggetto servizio comandi.

  È possibile eseguire contemporaneamente più istanze di un oggetto servizio comandi. Lo stato di un oggetto servizio comandi non può essere monitorato.

  STARTCMD
  Il programma eseguito per avviare il servizio. È necessario specificare un percorso completo per il programma.

  STARTARG
  Argomenti passati al programma di avvio.

  STDERR
  Specifica il percorso di un file a cui reindirizzare l'errore standard (stderr) del programma di servizio.

  STDOUT
  Specifica il percorso di un file a cui reindirizzare l'emissione standard (stdout) del programma di servizio.

  STOPCMD
  Il programma eseguito per arrestare il servizio. È necessario specificare un percorso completo per il programma.

  STOPARG
  Argomenti passati al programma di arresto.

  CONTROL
  Specifica il modo in cui il servizio deve essere avviato e arrestato:

MANUAL
Il servizio non deve essere avviato automaticamente o arrestato automaticamente. Viene controllato utilizzando i comandi **START SERVICE** e **STOP SERVICE** . Questo è il valore predefinito.

QMGR
Il servizio da definire deve essere avviato e arrestato contemporaneamente all'avvio e all'arresto del gestore code.

SOLO
Il servizio deve essere avviato contemporaneamente all'avvio del gestore code, ma non è richiesto l'arresto quando il gestore code viene arrestato.

Attività correlate

"Gestione dei servizi" a pagina 186
Un'istanza di un oggetto servizio può essere avviata e arrestata automaticamente dal gestore code oppure avviata e arrestata utilizzando i comandi MQSC **START SERVICE** e **STOP SERVICE**.

Riferimenti correlati

Definisci servizio
VISUALIZZA SVSTATUS
Avvia servizio
Arresta servizio

Gestione dei servizi

Un'istanza di un oggetto servizio può essere avviata e arrestata automaticamente dal gestore code oppure avviata e arrestata utilizzando i comandi MQSC **START SERVICE** e **STOP SERVICE**.

Prima di iniziare

Nota: Questa attività richiede l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Procedura

- Impostare il parametro **CONTROL** sul gestore code per avviare o arrestare automaticamente un'istanza di un oggetto servizio oppure utilizzare i comandi MQSC **START SERVICE** e **STOP SERVICE** per eseguire questa operazione manualmente.

  Quando viene avviata un'istanza di un oggetto servizio, viene scritto un messaggio nel log degli errori del gestore code contenente il nome dell'oggetto servizio e l'ID processo del processo avviato. Di seguito viene riportato un esempio di voce di log per un oggetto servizio server in fase di avvio:

```
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.
```

  Di seguito è riportato un esempio di voce di log per un oggetto servizio comandi in fase di avvio:

```
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.
```

Quando un servizio del server delle istanze si arresta, viene scritto un messaggio nei log degli errori del gestore code contenenti il nome del servizio e l'ID processo del processo finale. Di seguito è riportato un esempio di voce di log per un oggetto servizio server in fase di arresto:

```
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.
```

Attività correlate

“Definizione di variabili di ambiente aggiuntive nel file service.env” a pagina 187
Quando un servizio viene avviato, l'ambiente in cui viene avviato il processo del servizio viene ereditato dall'ambiente del gestore code. È possibile definire ulteriori variabili di ambiente da impostare nell'ambiente del processo del servizio aggiungendo le variabili che si desidera definire a uno dei file di sovrascrittura dell'ambiente `service.env` .

Riferimenti correlati

STOP SERVICE (arrestare un servizio) su Multiplatforms

START SERVICE (avvio di un servizio) su Multiplatforms

Definizione di variabili di ambiente aggiuntive nel file service.env

Quando un servizio viene avviato, l'ambiente in cui viene avviato il processo del servizio viene ereditato dall'ambiente del gestore code. È possibile definire ulteriori variabili di ambiente da impostare nell'ambiente del processo del servizio aggiungendo le variabili che si desidera definire a uno dei file di sovrascrittura dell'ambiente `service.env` .

Informazioni su questa attività

Esistono due possibili file a cui è possibile aggiungere le variabili di ambiente:

- Il file `service.env` dell'ambito macchina
- Il file `service.env` dell'ambito del gestore code

Entrambi i file vengono elaborati, se disponibili, con le definizioni nel file di ambito del gestore code che hanno la precedenza su quelle nel file di ambito della macchina.

È possibile specificare qualsiasi variabile di ambiente nel file `service.env` . Ad esempio, se il servizio IBM MQ esegue una serie di comandi, potrebbe essere utile impostare la variabile utente **PATH** nel file `service.env` .

Nota: I valori su cui si imposta la variabile non possono essere variabili di ambiente; ad esempio CLASSPATH= *%CLASSPATH%* non è corretto. Allo stesso modo, su Linux PATH= *$PATH* `:/opt/mqm/bin` si ottengono risultati imprevisti.

CLASSPATH deve essere in maiuscolo e l'istruzione del percorso classe può contenere solo valori letterali. Alcuni servizi (ad esempio Telemetria) impostano il proprio percorso di classe. Il **CLASSPATH** definito in `service.env` viene aggiunto ad esso.

Il formato delle variabili definite nel file `service.env` è un elenco di coppie di variabili nome e valore. Ogni variabile deve essere definita su una nuova linea e ogni variabile viene presa come è esplicitamente definita, incluso lo spazio.

Procedura

- Aggiungere le variabili di ambiente al file di ambito macchina `service.env` .
  Questo file si trova in:
  - Linux AIX `/var/mqm` su sistemi AIX and Linux .
  - Windows La directory di dati selezionata durante l'installazione su sistemi Windows .
- Aggiungere le variabili di ambiente al file `service.env` dell'ambito gestore code.
  Questo file si trova nella directory dei dati del gestore code. Ad esempio, l'ubicazione del file di sovrascrittura ambiente per un gestore code denominato QMNAME è:
  - Linux AIX Su sistemi AIX and Linux , `/var/mqm/qmgrs/QMNAME/service.env`
  - Windows Su sistemi Windows , `C:\ProgramData\IBM\MQ\qmgrs\QMNAME\service.env`

Esempio di un file `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
```

Attività correlate

“Utilizzo di inserimenti sostituibili nelle definizioni servizio” a pagina 188
È possibile sostituire i token nella definizione di un oggetto servizio. I token sostituiti vengono sostituiti automaticamente con il relativo testo espanso quando viene eseguito il programma di servizio.

Riferimenti correlati

Descrizioni delle variabili di ambiente

Utilizzo di inserimenti sostituibili nelle definizioni servizio

È possibile sostituire i token nella definizione di un oggetto servizio. I token sostituiti vengono sostituiti automaticamente con il relativo testo espanso quando viene eseguito il programma di servizio.

Informazioni su questa attività

I token di sostituzione possono essere presi dal seguente elenco di token comuni o da qualsiasi variabile definita nel file, `service.env`.

Procedura

- Per utilizzare gli inserimenti sostituibili, inserire il token all'interno dei caratteri + in una delle stringhe **STARTCMD**, **STARTARG**, **STOPCMD**, **STOPARG**, **STDOUT** o **STDERR** .

Per esempi, consultare “Utilizzo di un oggetto servizio server” a pagina 189 e “Utilizzo di un oggetto servizio comandi” a pagina 191.

I seguenti sono token comuni che possono essere utilizzati per sostituire i token nella definizione di un oggetto servizio:

PERCORSO MQ_INSTALL_
L'ubicazione in cui è installato IBM MQ .

MQ_DATA_PATH
L'ubicazione della directory di dati IBM MQ :

- Linux AIX Su sistemi AIX and Linux , l'ubicazione della directory di dati IBM MQ è `/var/mqm/`
- Windows Su sistemi Windows , l'ubicazione della directory di dati IBM MQ è la directory di dati selezionata durante l'installazione di IBM MQ

QMNAME
Il nome del gestore code corrente.

NOME_SERVIZIO_MQ
Il nome del servizio.

PID MQ_SERVER_
Questo token può essere utilizzato solo dagli argomenti **STOPARG** e **STOPCMD** .

Per gli oggetti servizio server questo token viene sostituito con l'ID processo del processo avviato dagli argomenti **STARTCMD** e **STARTARG** . Altrimenti, questo token viene sostituito con 0.

MQ_Q_MGR_DATA_PATH
L'ubicazione della directory dei dati del gestore code.

MQ_Q_MGR_DATA_NAME
Il nome trasformato del gestore code. Per ulteriori informazioni sulla trasformazione dei nomi, consultare Informazioni sui nomi file IBM MQ.

Utilizzo di un oggetto servizio server

Questi esempi mostrano come definire, utilizzare e modificare un oggetto servizio server per avviare un controllo trigger o un altro programma.

Prima di iniziare

Nota: Questi esempi richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Questi esempi sono scritti con i caratteri separatori del percorso di stile UNIX , tranne dove diversamente indicato.

Procedura

1. **Definire un oggetto servizio del server, utilizzando il comando MQSC `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+')
```

dove:

+MQ_INSTALL_PATH+ è un token che rappresenta la directory di installazione.
+QMNAME+ è un token che rappresenta il nome del gestore code.

ACCOUNTS.INITIATION.QUEUE è la coda di avvio.

amqsstop è un programma di esempio fornito con IBM MQ che richiede al gestore code di interrompere tutte le connessioni per l'ID processo. amqsstop genera comandi PCF, quindi il server dei comandi deve essere in esecuzione.

+MQ_SERVER_PID+ è un token che rappresenta l'ID processo passato al programma di arresto.

Consultare “Utilizzo di inserimenti sostituibili nelle definizioni servizio” a pagina 188 per un elenco dei token comuni.

2. **Un'istanza dell'oggetto del servizio server viene eseguita al successivo avvio del gestore code. Tuttavia, è possibile avviare immediatamente un'istanza dell'oggetto servizio server con il comando MQSC START SERVICE :**

```
START SERVICE(S1)
```

3. **Visualizzare lo stato del processo del servizio server, utilizzando il comando MQSC DISPLAY SVSTATUS :**

```
DISPLAY SVSTATUS(S1)
```

4. **Modificare l'oggetto servizio server e fare in modo che gli aggiornamenti siano raccolti riavviando manualmente il processo servizio server, utilizzando il comando MQSC ALTER SERVICE .**

   L'oggetto servizio del server viene modificato in modo che la coda di iniziazione sia specificata come JUPITER.INITIATION.QUEUE.

```
ALTER SERVICE(S1) +
STARTARG('-m +QMNAME+ -q JUPITER.INITIATION.QUEUE')
```

   Nota: Un servizio in esecuzione non prende alcun aggiornamento alla sua definizione di servizio fino a quando non viene riavviato.

5. **Riavviare il processo del servizio server in modo che la modifica venga rilevata, utilizzando i comandi MQSC STOP SERVICE e START SERVICE**

```
STOP SERVICE(S1)
```

   seguito da:

```
START SERVICE(S1)
```

   Il processo di servizio del server viene riavviato e prende le modifiche apportate in “4” a pagina 190.

   Nota: Il comando MQSC, **STOP SERVICE**, può essere utilizzato solo se nella definizione del servizio è specificato un argomento **STOPCMD** .

Altri esempi di argomenti di passaggio

- **Definire un oggetto servizio server per avviare un programma denominato `runserv` quando viene avviato un gestore code.**

  Eseguire questa operazione utilizzando il comando MQSC **DEFINE SERVICE** .

  Questo esempio viene scritto con i caratteri separatori del percorso di stile Windows .

  Uno degli argomenti passati al programma iniziale è una stringa contenente uno spazio. Questo argomento deve essere passato come una singola stringa. Per ottenere questo risultato, le virgolette doppie vengono utilizzate come mostrato nel seguente comando per definire l'oggetto servizio comando.

```
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')
```

dove:

+QMNAME+ è un token che rappresenta il nome del gestore code.

`"C:\Program Files\Tools\"` è una stringa contenente uno spazio, che verrà passato come una singola stringa.

- **Definire un oggetto servizio del server che può essere utilizzato per avviare automaticamente il monitoraggio trigger all'avvio del gestore code.**

  Eseguire questa operazione utilizzando il comando MQSC **DEFINE SERVICE** .

```
DEFINE SERVICE(TRIG_MON_START) +
CONTROL(QMGR) +
SERVTYPE(SERVER) +
STARTCMD('runmqtrm') +
STARTARG('-m +QMNAME+ -q +IQNAME+')
```

  dove:

  +QMNAME+ è un token che rappresenta il nome del gestore code.

  +IQNAME+ è una variabile di ambiente definita dall'utente in uno dei file service.env che rappresentano il nome della coda di iniziazione.

Riferimenti correlati

MODIFICA SERVIZIO

Definisci servizio

VISUALIZZA SVSTATUS

Avvia servizio

Arresta servizio

Utilizzo di un oggetto servizio comandi

Questi esempi mostrano come definire un oggetto servizio comandi per avviare un programma che scrive le voci nel log di sistema del sistema operativo quando un gestore code viene avviato o arrestato.

Prima di iniziare

Nota: Questi esempi richiedono l'esecuzione del comando MQSC **DEFINE SERVICE** . La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Questi esempi sono scritti con caratteri separatori di percorso in stile UNIX .

Informazioni su questa attività

Nei seguenti esempi:

`logger` è un programma di esempio fornito con IBM MQ che può scrivere voci nel log del sistema operativo.

+QMNAME+ è un token che rappresenta il nome del gestore code.

Procedura

- Definire un oggetto del servizio comandi per avviare un programma che scrive le voci nel log di sistema del sistema operativo quando un gestore code viene avviato o arrestato:

```
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')
```

- Definire un oggetto servizio comandi per avviare un programma che scrive voci nel log di sistema del sistema operativo solo quando un gestore code viene arrestato:

```
DEFINE SERVICE(S3) +
CONTROL(QMGR) +
SERVTYPE(COMMAND) +
STOPCMD('/usr/bin/logger') +
STOPARG('Queue manager +QMNAME+ stopping')
```

Riferimenti correlati
Definisci servizio

Gestione degli oggetti per il trigger

Questi esempi mostrano come avviare un'applicazione automaticamente quando vengono soddisfatte determinate condizioni su una coda. Ad esempio, si potrebbe voler avviare un'applicazione quando il numero di messaggi su una coda raggiunge un numero specificato. Questa funzione è denominata *attivazione*. È necessario definire gli oggetti che supportano il trigger.

Prima di iniziare

Nota: Questi esempi richiedono l'esecuzione di comandi MQSC. La modalità di tale operazione varia in base alla piattaforma. Consultare Amministrazione IBM MQ utilizzando i comandi MQSC.

Questi esempi sono scritti con caratteri separatori di percorso in stile UNIX .

Informazioni su questa attività

Per una descrizione dettagliata del trigger, consultare Avvio delle applicazioni IBM MQ utilizzando i trigger.

Procedura

- Definire una coda dell'applicazione per il trigger.

  Una coda dell'applicazione è una coda locale utilizzata dalle applicazioni per la messaggistica, tramite MQI. L'attivazione richiede la definizione di un certo numero di attributi della coda sulla coda dell'applicazione.

  L'attivazione è abilitata dall'attributo **Trigger** (TRIGGER nei comandi MQSC). In questo esempio, un evento trigger deve essere generato quando ci sono 100 messaggi di priorità 5 o superiore sulla coda locale MOTOR.INSURANCE.QUEUE, come segue:

```
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)
```

dove:

QLOCAL (MOTOR.INSURANCE.QUEUE)
Indica il nome della coda dell'applicazione che si sta definendo.

PROCESS (MOTOR.INSURANCE.QUOTE.PROCESS)
Indica il nome della definizione del processo che definisce l'applicazione che deve essere avviata da un programma di controllo trigger.

MAXMSGL (2000)
Indica la lunghezza massima dei messaggi sulla coda.

DEFPSIST (YES)
Specifica che i messaggi su questa coda sono persistenti per impostazione predefinita.

INITQ (MOTOR.INS.INIT.QUEUE)
Indica il nome della coda di iniziazione su cui il gestore code deve inserire il messaggio trigger.

TRIGGER
Indica il valore dell'attributo trigger.

TRIGTYPE (DEPTH)
Specifica che un evento trigger viene generato quando il numero di messaggi con la priorità richiesta (TRIGMPRI) raggiunge il numero specificato in TRIGDPTH.

TRIGDPTH (100)
Indica il numero di messaggi richiesti per generare un evento trigger.

TRIGMPRI (5)
Indica la priorità dei messaggi che devono essere conteggiati dal gestore code nel decidere se generare un evento trigger. Vengono conteggiati solo i messaggi con priorità 5 o superiore.

- Definire una coda di iniziazione

  Quando si verifica un evento trigger, il gestore code inserisce un messaggio trigger nella coda di avvio specificata nella definizione della coda dell'applicazione. Le code di iniziazione non dispongono di impostazioni speciali, ma è possibile utilizzare la definizione seguente della coda locale MOTOR.INS.INIT.QUEUE per istruzioni:

  ```
  DEFINE QLOCAL(MOTOR.INS.INIT.QUEUE) +
  GET (ENABLED) +
  NOSHARE +
  NOTRIGGER +
  MAXMSGL (2000) +
  MAXDEPTH (1000)
  ```

- Definire un processo

  Utilizzare il comando DEFINE PROCESS per creare un processo di definizione. Una definizione processo definisce l'applicazione da utilizzare per elaborare i messaggi dalla coda dell'applicazione. La definizione della coda dell'applicazione denomina il processo da utilizzare e quindi associa la coda dell'applicazione all'applicazione da utilizzare per elaborare i relativi messaggi. Questa operazione viene eseguita tramite l'attributo PROCESS sulla coda dell'applicazione MOTOR.INSURANCE.QUEUE. Il seguente comando MQSC definisce il processo richiesto, MOTOR.INSURANCE.QUOTE.PROCESS, identificato in questo esempio:

  ```
  DEFINE PROCESS (MOTOR.INSURANCE.QUOTE.PROCESS) +
  DESCR ('Insurance request message processing') +
  APPLTYPE (UNIX) +
  APPLICID ('/u/admin/test/IRMP01') +
  USERDATA ('open, close, 235')
  ```

  dove:

MOTOR.INSURANCE.QUOTE.PROCESS
È il nome della definizione del processo.

DESCR ('Insurance request message processing')
Descrive il programma applicativo a cui si riferisce questa definizione. Questo testo viene visualizzato quando si utilizza il comando DISPLAY PROCESS. Questo può aiutare a identificare cosa fa il processo. Se si utilizzano spazi nella stringa, è necessario racchiudere la stringa tra virgolette singole.

APPLTYPE (UNIX)
Indica il tipo di applicazione da avviare.

APPLICID ('/u/admin/test/IRMP01')
È il nome del file eseguibile dell'applicazione, specificato come nome file completo. Nei sistemi Windows , un tipico valore APPLICID è `c:\appl\test\irmp01.exe`.

USERDATA ('open, close, 235')
Sono dati definiti dall'utente, che possono essere utilizzati dall'applicazione.

- Visualizza gli attributi di un processo di definizione

  Utilizzare il comando DISPLAY PROCESS per esaminare i risultati della definizione. Ad esempio:

```
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)
```

  È inoltre possibile utilizzare il comando MQSC ALTER PROCESS per modificare una definizione di processo esistente e il comando DELETE PROCESS per eliminare una definizione di processo.

Utilizzo del programma di utilità dmpmqmsg tra due sistemi

Il programma di utilità **dmpmqmsg** (in precedenza *qload*) consente di copiare o spostare il contenuto di una coda o dei relativi messaggi in un file.

Panoramica

Il file creato con **dmpmqmsg** può essere salvato come richiesto e utilizzato in un secondo momento per ricaricare i messaggi nella coda.

Importante:

1. Il file ha un formato specifico compreso dal programma di utilità. Tuttavia, il file è leggibile, in modo da poterlo aggiornare in un editor prima di ricaricarlo. Se si modifica il file, non è necessario modificarne il formato.
2. Il programma di utilità **dmpmqmsg** viene fornito con il fileset di runtime per AIX, Linux, and Windows, quindi è disponibile sia sul server IBM MQ che sul client.

I possibili utilizzi sono:

- Salvataggio dei messaggi che si trovano in una coda in un file. Possibilmente per scopi di archiviazione e ricaricamento successivo in una coda.
- Ricaricamento di una coda con messaggi precedentemente salvati in un file.
- Rimozione di vecchi messaggi da una coda.
- 'Riproducendo' i messaggi di test da una posizione memorizzata, anche mantenendo l'ora corretta tra i messaggi, se necessario.

Attenzione: SupportPac MO03 ha utilizzato il parametro **-l** per specificare il bind locale o client. **-l** è stato sostituito dal parametro **-c** .

-P viene ora utilizzato per le informazioni della codepage invece di **-c**.

Consultare dmpmqmsg per ulteriori informazioni sul comando e sui parametri disponibili.

Esempio di utilizzo del programma di utilità dmpmqmsg su Linux, utilizzando una macchina Windows

Si dispone di un gestore code su una macchina Linux che contiene messaggi su una coda (*Q1*) che si desidera spostare in un'altra coda (*Q2*) nello stesso gestore code. Si desidera avviare il programma di utilità **dmpmqmsg** da un computer Windows .

La coda (*Q1*) contiene quattro messaggi che sono stati aggiunti utilizzando l'applicazione di esempio **amqsput** (gestore code locale) o **amqsputc** (gestore code remoto).

Sulla macchina Linux vengono visualizzati:

```
display ql(Q1) CURDEPTH
        2 : display ql(Q1) CURDEPTH
AMQ8409: Display Queue details.
     QUEUE(Q1)
TYPE(QLOCAL)
     CURDEPTH(4)
```

Impostare la variabile di ambiente MQSERVER per puntare al gestore code in Linux. Ad esempio:

```
set MQSERVER=SYSTEM.DEF.SVRCONN/TCP/veracruz.x.com(1414)
```

dove *veracruz* è il nome della macchina.

Eseguire il programma di utilità **dmpmqmsg** per leggere dalla coda, *Q1*e memorizzare l'output in `c:\temp\mqqload.txt`.

Connettersi come client remoto al gestore code, *QM_VER*, in esecuzione sull'host Linux e sulla porta stabilita da MQSERVER. Si ottiene la connessione come un client remoto utilizzando l'attributo: `-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
```

Il file di output `c:\temp\mqqload.txt` contiene testo, utilizzando un formato che il programma di utilità **dmpmqmsg** comprende.

Sulla macchina Windows , emettere il comando **dmpmqmsg** (utilizzando l'opzione `-o` invece dell'opzione `-i`) per caricare la coda (*Q2*) sulla macchina Linux da un file sulla macchina 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
```

Sulla macchina Linux , notare che ora sono presenti quattro messaggi nella coda che sono stati ripristinati dal file.

```
display ql(Q2) CURDEPTH
        6 : display ql(Q2) CURDEPTH
AMQ8409: Display Queue details.
     QUEUE(Q2)
TYPE(QLOCAL)
     CURDEPTH(4)
```

Sulla macchina Linux ,

Eliminare i messaggi dalla coda originale.

```
clear qlocal(Q1)
        4 : clear qlocal(Q1)
AMQ8022: IBM MQ queue cleared.
```

Confermare che non ci sono più messaggi sulla coda originale:

```
display ql(Q1) CURDEPTH
        5 : display ql(Q1) CURDEPTH
AMQ8409: Display Queue details.
     QUEUE(Q1)
TYPE(QLOCAL)
     CURDEPTH(0)
```

Consultare dmpmqmsg per una descrizione del comando e dei relativi parametri.

Concetti correlati

“Esempi di utilizzo del programma di utilità dmpmqmsg” a pagina 196
Semplici modi in cui è possibile utilizzare il programma di utilità **dmpmqmsg** (in precedenza **qload**).

Esempi di utilizzo del programma di utilità dmpmqmsg

Semplici modi in cui è possibile utilizzare il programma di utilità **dmpmqmsg** (in precedenza **qload**).

Scarica una coda in un file

Utilizzare le seguenti opzioni sulla riga comandi per salvare i messaggi che si trovano in una coda in un file:

```
dmpmqmsg -m QM1 -i Q1 -f c:\myfile
```

Questo comando acquisisce una copia dei messaggi dalla coda e li salva nel file specificato.

Scarica una coda in una serie di file

È possibile scaricare una coda in una serie di file utilizzando un carattere `insert` nel nome file. In questa modalità ogni messaggio viene scritto in un nuovo file:

```
dmpmqmsg -m QM1 -i Q1 -f c:\myfile%n
```

Questo comando scarica la coda in file, `myfile1`, `myfile2`, `myfile3`e così via.

Carica una coda da un file

Per ricaricare una coda con i messaggi salvati in “Scarica una coda in un file” a pagina 196, utilizzare le seguenti opzioni sulla riga comandi:

```
dmpmqmsg -m QM1 -o Q1 -f c:\myfile%n
```

Questo comando scarica la coda in file, `myfile1`, `myfile2`, `myfile3`e così via.

Carica una coda da una serie di file

È possibile caricare una coda da una serie di file utilizzando un carattere `insert` nel nome file. In questa modalità ogni messaggio viene scritto in un nuovo file:

```
dmpmqmsg -m QM1 -o Q1 -f c:\myfile%n
```

Questo comando carica la coda in file, `myfile1`, `myfile2`, `myfile3`e così via.

Copiare i messaggi da una coda ad un'altra coda

Sostituire il parametro del file in “Scarica una coda in un file” a pagina 196con un altro nome coda e utilizzare le seguenti opzioni:

```
dmpmqmsg -m QM1 -i Q1 -o Q2
```

Questo comando consente di copiare i messaggi di una coda in un'altra coda.

Copiare i primi 100 messaggi da una coda all'altra

Utilizzare il comando nell'esempio precedente e aggiungere l'opzione `-r#100` :

```
dmpmqmsg -m QM1 -i Q1 -o Q2 -r#100
```

Spostare i messaggi da una coda all'altra

Una variazione su “Carica una coda da un file” a pagina 196. Notare la distinzione tra l'utilizzo di **-i** (minuscolo) che sfoglia solo una coda e **-I** (maiuscolo) che deriva in modo distruttivo da una coda:

```
dmpmqmsg -m QM1 -I Q1 -o Q2
```

Spostare i messaggi più vecchi di un giorno da una coda all'altra

Questo esempio mostra l'uso della selezione dell'età. È possibile selezionare messaggi più vecchi, più giovani o con un intervallo di età.

```
dmpmqmsg -m QM1 -I Q1 -o Q2 -T1440
```

Visualizza le età dei messaggi attualmente in coda

Utilizzare le seguenti opzioni sulla riga comandi:

```
dmpmqmsg -m QM1 -i Q1 -f stdout -dT
```

Gestione file messaggi

Dopo aver scaricato il messaggio dalla coda, come in “Scarica una coda in un file” a pagina 196, è possibile modificare il file.

Si potrebbe anche voler modificare il formato del file per utilizzare una delle opzioni di visualizzazione non specificate al momento dello scaricamento della coda.

È possibile utilizzare il programma di utilità **dmpmqmsg** per rielaborare il file nel formato richiesto anche dopo lo scaricamento della coda. Utilizzare le opzioni riportate di seguito sulla riga comandi.

```
dmpmqmsg -f c:\oldfile -f c:\newfile -dA
```

Consultare dmpmqmsg per una descrizione del comando e dei relativi parametri.

Utilizzo di oggetti IBM MQ remoti

È possibile gestire gli oggetti IBM MQ su gestori code remoti utilizzando i comandi MQSC, PCF o administrative REST API. Prima di poter utilizzare uno qualsiasi di questi metodi, è necessario definire le code di trasmissione e i canali tra il gestore code locale e il gestore code remoto, in modo che i comandi possano essere inviati al gestore code remoto e le risposte ricevute dal gestore code locale. In alternativa, è possibile configurare un cluster di gestore code e utilizzare gli stessi metodi di amministrazione remota.

Informazioni su questa attività

Per preparare i gestori code per la gestione remota, è necessario configurare i seguenti oggetti sul gestore code locale:

- Un listener.
- Una coda di trasmissione che ha il nome del gestore code remoto.
- Un canale mittente con i dettagli di connessione per il gestore code remoto.
- Un canale ricevente con lo stesso nome del canale mittente sul gestore code remoto.

È inoltre necessario configurare i seguenti oggetti sul gestore code remoto:

- Un listener.
- Una coda di trasmissione con il nome del gestore code locale.
- Un canale mittente che dispone dei dettagli di connessione per il gestore code locale.
- Un canale ricevente con lo stesso nome del canale mittente sul gestore code locale.

Per ulteriori informazioni sulla configurazione di questi oggetti, consultare “Configurazione dei gestori code per la gestione remota” a pagina 199.

In alternativa, è possibile configurare un cluster di gestore code. Un *cluster* è un gruppo di gestori code configurato in modo che i gestori code possano comunicare direttamente tra loro su una singola rete senza complesse definizioni di coda di trasmissione, canale e coda. I cluster possono essere configurati facilmente e in genere contengono gestori code che sono logicamente correlati in qualche modo e che devono condividere dati o applicazioni. Anche il cluster più piccolo riduce i costi di amministrazione del sistema.

La creazione di una rete di gestori code in un cluster implica meno definizioni rispetto alla creazione di un ambiente di accodamento distribuito tradizionale. Con un numero inferiore di definizioni da creare, è possibile impostare o modificare la rete in modo più rapido e semplice e ridurre il rischio di errori nelle definizioni.

Per impostare un cluster, è necessaria una definizione del mittente del cluster (CLUSSDR) e un destinatario del cluster (CLUSRCVR) per ogni gestore code. Non è necessaria alcuna definizione di coda di trasmissione o di coda remota. I principi dell'amministrazione remota sono gli stessi quando vengono utilizzati all'interno di un cluster, ma le definizioni stesse sono notevolmente semplificate.

Per ulteriori informazioni sulla configurazione di un cluster, consultare Configurazione di un cluster di gestori code.

Procedura

- Per informazioni su come gestire gli oggetti IBM MQ remoti, consultare i seguenti argomenti secondari:
  - “Configurazione dei gestori code per la gestione remota” a pagina 199
  - “Gestione del server dei comandi per la gestione remota” a pagina 203
  - “Immissione di comandi MQSC su un gestore code remoto” a pagina 203
  - “Conversione dati tra serie di caratteri codificati” a pagina 205

Configurazione dei gestori code per la gestione remota

È possibile gestire un gestore code remoto da un gestore code locale utilizzando i comandi administrative REST API, MQSC o PCF. Il gestore code remoto potrebbe trovarsi sullo stesso sistema, in un'installazione differente o su un altro sistema con lo stesso ambiente o su un altro ambiente IBM MQ . Prima di poter gestire in remoto un gestore code da un gestore code locale, è necessario creare un canale mittente e destinatario, un listener e una coda di trasmissione su ciascun gestore code. Questi canali e code consentono di inviare i comandi al gestore code remoto e di ricevere le risposte sul gestore code locale. La procedura per creare queste code e canali è la stessa se si desidera utilizzare i comandi administrative REST API, MQSC o PCF.

Prima di iniziare

- La seguente procedura utilizza i gestori code di esempio `source.queue.manager`e `target.queue.manager`. È necessario creare e avviare questi gestori code sul sistema per seguire questi passi oppure sostituire i propri nomi di gestori code nei passi pertinenti.
- La seguente procedura utilizza TCP/IP come tipo di trasporto. È necessario conoscere l'indirizzo IP di entrambi i sistemi per completare questa attività.
- La seguente procedura crea listener che utilizzano le porte di rete 1818 sul sistema locale e 1819 sul sistema remoto. È possibile utilizzare altre porte, ma è necessario sostituire i valori di porta nei passaggi appropriati.
- È necessario eseguire i comandi nella procedura localmente o su una funzione di rete come Telnet.

Informazioni su questa attività

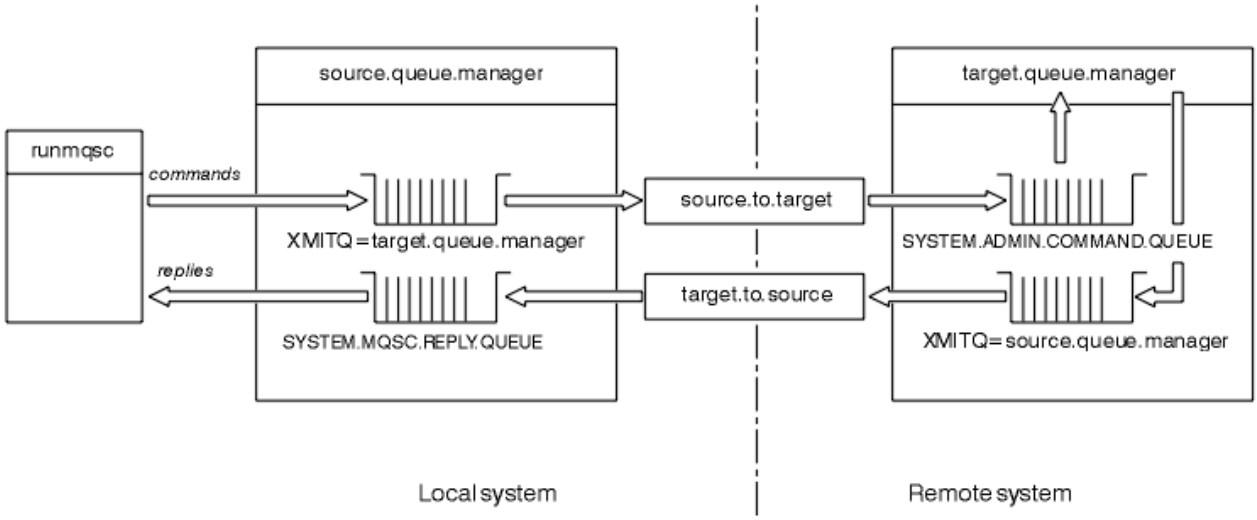


Figura 15. Impostazione di canali e code per l'amministrazione remota

Figura 15 a pagina 199 mostra la configurazione di gestori code, code e canali necessari per la gestione remota:

- L'oggetto `source.queue.manager` è il gestore code di origine da cui è possibile emettere comandi administrative REST API, MQSC o PCF e a cui vengono restituiti i risultati di questi comandi.
- L'oggetto `target.queue.manager` è il nome del gestore code di destinazione, che elabora i comandi e genera eventuali messaggi dell'operatore.
- I comandi vengono inseriti nella coda di trasmissione con lo stesso nome del gestore code remoto. In questo caso, `target.queue.manager`. Una coda di trasmissione è una coda locale specializzata che conserva temporaneamente i messaggi prima che MCA li raccolga e li invii al gestore code remoto.
- I comandi vengono inviati dal canale `source.to.target` a `SYSTEM.ADMIN.COMMAND.QUEUE` sul gestore code remoto. Ogni estremità del canale ha una definizione separata. Un'estremità è un mittente

e l'altra estremità è un destinatario. Le due definizioni devono avere lo stesso nome e insieme costituiscono un singolo canale di messaggi.

- L'output del comando viene inserito nella coda di trasmissione remota con lo stesso nome del gestore code locale da cui è stato inviato il comando. In questo caso, `source.queue.manager`.
- L'output viene inviato dal canale `target.to.source` ad una coda di risposta appropriata, dove viene acquisito ed emesso dal comando originale.

Procedura

1. Sul gestore code del sistema remoto, verificare la presenza della coda comandi SYSTEM.ADMIN.COMMAND.QUEUE . Questa coda viene creata per impostazione predefinita quando viene creato il gestore code.
2. Sul sistema remoto, controllare che il command server sia in esecuzione sul gestore code. Se il server dei comandi non è in esecuzione, la gestione remota non è possibile:
   a) Avviare **runmqsc** per il gestore code. Ad esempio, per il gestore code `target.queue.manager`, immettere il comando seguente:

   ```
   runmqsc target.queue.manager
   ```

   b) Visualizzare lo stato del server dei comandi immettendo il seguente comando:

   ```
   DISPLAY QMSTATUS CMDSERV
   ```

   c) Uscire dal prompt dei comandi di **runmqsc** immettendo il seguente comando:

   ```
   end
   ```

   d) Se il server dei comandi non è avviato, avviarlo. Ad esempio, per il gestore code `target.queue.manager`, immettere il comando seguente:

   ```
   strmqcsv target.queue.manager
   ```

3. Definire i canali, il listener e la coda di trasmissione sul gestore code locale:
   a) Avviare **runmqsc** per il gestore code. Ad esempio, per il gestore code `source.queue.manager`, immettere il comando seguente:

   ```
   runmqsc source.queue.manager
   ```

   b) Definire il canale mittente. Questo canale mittente deve avere lo stesso nome del canale ricevente sul gestore code remoto. Ad esempio, immettere il seguente comando MQSC, sostituendo il valore per **CONNAME** con l'indirizzo IP per il gestore code remoto e il numero porta del listener:

   ```
   DEFINE CHANNEL ('source.to.target') +
   CHLTYPE(SDR) +
   CONNAME (localhost:1819) +
   XMITQ ('target.queue.manager') +
   TRPTYPE(TCP)
   ```

   c) Definire il canale destinatario. Questo canale ricevente deve avere lo stesso nome del canale mittente sul gestore code remoto. Ad esempio, immetti il seguente comando:

   ```
   DEFINE CHANNEL ('target.to.source') +
   CHLTYPE(RCVR) +
   TRPTYPE(TCP)
   ```

   d) Definire il listener sul gestore code locale. Ad esempio, immetti il seguente comando:

   ```
   DEFINE LISTENER ('source.queue.manager') +
   ```

```
TRPTYPE (TCP) +
PORT (1818)
```

e) Definire la coda di trasmissione sul gestore code locale. Questa coda di trasmissione deve avere lo stesso nome del gestore code remoto. Ad esempio, immetti il seguente comando:

```
DEFINE QLOCAL ('target.queue.manager') +
USAGE (XMITQ)
```

f) Avviare il listener. Ad esempio, immetti il seguente comando:

```
START LISTENER ('source.queue.manager')
```

g) Uscire dal prompt dei comandi di **runmqsc** immettendo il seguente comando:

```
end
```

4. Definire i canali, il listener e la coda di trasmissione sul gestore code remoto:

a) Avviare **runmqsc** per il gestore code. Ad esempio, per il gestore code `target.queue.manager`, immettere il comando seguente:

```
runmqsc target.queue.manager
```

b) Definire il canale mittente. Questo canale mittente deve avere lo stesso nome del canale ricevente sul gestore code locale. Ad esempio, immettere il comando MQSC riportato di seguito, sostituendo il valore per **CONNAME** con l'indirizzo IP per il gestore code locale e il numero di porta per il listener:

```
DEFINE CHANNEL ('target.to.source') +
CHLTYPE(SDR) +
CONNAME (localhost:1818) +
XMITQ ('source.queue.manager') +
TRPTYPE(TCP)
```

c) Definire il canale destinatario. Questo canale ricevente deve avere lo stesso nome del canale mittente sul gestore code locale. Ad esempio, immettere il seguente comando:

```
DEFINE CHANNEL ('source.to.target') +
CHLTYPE(RCVR) +
TRPTYPE(TCP)
```

d) Definire il listener. Ad esempio, immetti il seguente comando:

```
DEFINE LISTENER ('target.queue.manager') +
TRPTYPE (TCP) +
PORT (1819)
```

e) Definire la coda di trasmissione. Questa coda di trasmissione deve avere lo stesso nome del gestore code locale. Ad esempio, immetti il seguente comando:

```
DEFINE QLOCAL ('source.queue.manager') +
USAGE (XMITQ)
```

f) Avviare il listener. Ad esempio, immetti il seguente comando:

```
START LISTENER ('target.queue.manager')
```

g) Uscire da **runmqsc** immettendo il comando seguente:

```
end
```

5. Avviare il canale mittente sul sistema locale:

a) Avviare **runmqsc** per il gestore code. Ad esempio, per il gestore code `source.queue.manager`, immettere il comando seguente:

```
runmqsc source.queue.manager
```

b) Avviare il canale mittente. Ad esempio, immetti il seguente comando:

```
START CHANNEL ('source.to.target')
```

c) Uscire da **runmqsc** immettendo il comando seguente:

```
end
```

6. Avviare il canale mittente sul sistema remoto:
   a) Avviare runmqsc per il gestore code. Ad esempio, per il gestore code `target.queue.manager`, immettere il comando seguente:

```
runmqsc target.queue.manager
```

   b) Avviare il canale mittente. Ad esempio, immetti il seguente comando:

```
START CHANNEL ('target.to.source')
```

   c) Uscire da **runmqsc** immettendo il comando seguente:

```
end
```

7. Verificare che la configurazione sia stata completata correttamente inviando un comando MQSC dal sistema locale al gestore code remoto:
   a) Avviare il prompt dei comandi **runmqsc** per il gestore code remoto dal sistema locale. Ad esempio, immetti il seguente comando:

```
runmqsc -w 30 -m source.queue.manager target.queue.manager
```

   b) Visualizzare le code sul gestore code remoto immettendo il seguente comando:

```
DISPLAY QUEUE (*)
```

   In caso di esito positivo, viene visualizzato un elenco di code dal gestore code remoto.

   c) Se questi passi non funzionano, verificare che i canali su entrambi i sistemi siano in uno stato di esecuzione. Se i canali non sono in esecuzione e non vengono avviati, verificare che i canali e le code di trasmissione siano configurati correttamente e che il server dei comandi sia in esecuzione. Ad esempio, verificare che sia specificato il CONNAME corretto per i canali mittente e che le code di trasmissione abbiano i nomi corretti. Inoltre, controllare i log del gestore code per le eccezioni di sicurezza che potrebbero aiutare a risolvere il problema.

Risultati

I gestori code sono configurati per gestire in remoto il gestore code remoto dal sistema locale.

Operazioni successive

- Ulteriori informazioni sulla gestione remota utilizzando i comandi MQSC: “Immissione di comandi MQSC su un gestore code remoto” a pagina 203
- Ulteriori informazioni sulla scrittura di programmi di gestione utilizzando i comandi PCF: “Utilizzo di IBM MQ Programmable Command Format” a pagina 26.
- Ulteriori informazioni sull'utilizzo di administrative REST API per la gestione remota: “Gestione remota mediante REST API” a pagina 80.

Gestione del server dei comandi per la gestione remota

A ciascun gestore code è associato un server dei comandi. Un server dei comandi elabora i comandi in entrata dai gestori code remoti o i comandi PCF dalle applicazioni. Presenta i comandi al gestore code per l'elaborazione e restituisce un codice di completamento o un messaggio operatore. È possibile avviare, arrestare e visualizzare lo stato del server dei comandi. Un server dei comandi è obbligatorio per tutte le operazioni di gestione che coinvolgono comandi PCF, MQAI e anche per la gestione remota.

Prima di iniziare

A seconda del valore dell'attributo del gestore code, **SCMDSERV**, il server dei comandi viene avviato automaticamente all'avvio del gestore code o deve essere avviato manualmente. Se il server dei comandi viene avviato automaticamente, non è possibile utilizzare i comandi strmqcsv o endmqcsv per avviare e arrestare il server dei comandi. È possibile modificare il valore dell'attributo **SCMDSERV** utilizzando il comando MQSC **ALTER QMGR**. Per impostazione predefinita, il server dei comandi viene avviato automaticamente.

L'arresto di un gestore code termina anche il server dei comandi ad esso associato.

Procedura

- Visualizzare lo stato del server dei comandi:
  a) Avviare il prompt dei comandi **runmqsc** per il gestore code appropriato immettendo il comando riportato di seguito:

     ```
     runmqsc target.queue.manager
     ```

     dove `target.queue.manager` è il gestore code per cui viene visualizzato il server dei comandi.
  b) Visualizzare lo stato del server dei comandi immettendo i seguenti comandi MQSC:

     ```
     DISPLAY QMSTATUS CMDSERV
     ```

  c) Uscire dal prompt dei comandi di **runmqsc** immettendo il seguente comando:

     ```
     end
     ```

- Se il server dei comandi non è impostato per l'avvio automatico, avviarlo immettendo il seguente comando:

  ```
  strmqcsv target.queue.manager
  ```

  dove `target.queue.manager` è il gestore code per cui viene avviato il server dei comandi.
- Se il server dei comandi non è impostato per l'avvio automatico, arrestare il server dei comandi immettendo il comando seguente:

  ```
  endmqcsv target.queue.manager
  ```

  dove `target.queue.manager` è il gestore code per cui viene arrestato il server dei comandi.

  Per impostazione predefinita, il server dei comandi viene arrestato in modo controllato. È possibile arrestare immediatamente il server dei comandi aggiungendo l'indicatore `-i` al comando.

Immissione di comandi MQSC su un gestore code remoto

Dopo aver configurato i gestori code per la gestione remota, è possibile utilizzare un particolare formato del comando **runmqsc** su un sistema locale per eseguire i comandi MQSC su un gestore code remoto.

Ogni comando viene inviato come PCF Escape alla coda comandi, SYSTEM.ADMIN.COMMAND.QUEUE, del gestore code remoto. Le risposte vengono ricevute sul SISTEMA SYSTEM.MQSC.REPLY.QUEUE .

Prima di iniziare

È necessario completare la procedura in "Configurazione dei gestori code per la gestione remota" a pagina 199 per configurare canali, code di trasmissione, listener e il server dei comandi prima di poter amministrare in remoto un gestore code utilizzando i comandi MQSC.

Procedura

1. Verificare che il server dei comandi sia in esecuzione sul gestore code remoto.

   Per informazioni su come avviare il server dei comandi su un gestore code, consultare "Gestione del server dei comandi per la gestione remota" a pagina 203.
2. Sul gestore code di origine, è possibile quindi eseguire i comandi MQSC in uno dei seguenti modi:
   - In modo interattivo, avviando **runmqsc** con i seguenti comandi:
     - Se il gestore code remoto si trova su z/OS, immettere il seguente comando:

       ```
       runmqsc -w 30 -x -m source.queue.manager target.queue.manager
       ```
     - Se il gestore code remoto si trova su Multiplatforms, immettere il seguente comando:

       ```
       runmqsc -w 30 -m source.queue.manager target.queue.manager
       ```
   - Da un file di comandi:
     a. Inserire i comandi MQSC da eseguire sul sistema remoto in un file di testo, un comando per riga.
     b. Verificare i comandi MQSC sul gestore code locale utilizzando l'indicatore -v sul comando **runmqsc** . L'indicatore -v verifica la validità dei comandi, ma non li esegue. Tenere presente che alcuni comandi potrebbero avere esito negativo se sono applicabili al gestore code remoto ma non al gestore code locale:

        ```
        runmqsc -v source.queue.manager < myCmdFile.in > results.out
        ```

        Il file `myCmdFile.in` contiene i comandi MQSC da controllare e il file `results.out` contiene i risultati della verifica per i comandi.
     c. Eseguire il file di comandi sul gestore code remoto immettendo uno dei seguenti comandi:
        - Se il gestore code remoto si trova su z/OS, immettere il seguente comando:

          ```
          runmqsc -w 30 -x -m source.queue.manager target.queue.manager < myCmdFile.in >
          results.out
          ```
        - Se il gestore code remoto si trova su Multiplatforms, immettere il seguente comando:

          ```
          runmqsc -w 30 -m source.queue.manager target.queue.manager < myCmdFile.in >
          results.out
          ```

   I parametri utilizzati sono i seguenti:

   -w *secondi*
   Specifica che i comandi MQSC vengono eseguiti in modalità indiretta, in cui i comandi vengono inseriti nella coda di input del server dei comandi ed eseguiti in ordine.

   La variabile *secondi* specifica il periodo di attesa, in secondi, per una risposta dal gestore code remoto. Tutte le risposte ricevute dopo questo orario vengono eliminate, ma i comandi MQSC

vengono ancora eseguiti sul gestore code remoto. Il seguente messaggio viene generato sul gestore code locale quando il comando scade:

```
AMQ8416: MQSC timed out waiting for a response from the command server.
```

Quando si smette di emettere comandi MQSC, il gestore code locale visualizza tutte le risposte in timeout che sono arrivate e scarta tutte le altre risposte.

-x
Specifica che il gestore code remoto è un gestore code z/OS .

-m *localQMgrNome*
Specifica il nome del gestore code locale che si desidera utilizzare per inoltrare i comandi al gestore code remoto

Operazioni successive

Se hai difficoltà ad eseguire i comandi MQSC in remoto:

- Verificare che il gestore code remoto sia in esecuzione.
- Controllare che il server dei comandi sia in esecuzione sul sistema remoto.
- Verificare che l'intervallo di disconnessione del canale non sia scaduto. Ad esempio, se un canale è stato avviato ma è stato chiuso dopo un certo periodo di tempo. Questo è particolarmente importante se si avviano i canali manualmente.
- Verificare che le richieste inviate dal gestore code locale abbiano senso al gestore code di destinazione. Ad esempio, le richieste che includono parametri non supportati sul gestore code remoto.
- Consultare anche Risoluzione dei problemi con i comandi MQSC.

Conversione dati tra serie di caratteri codificati

I dati dei messaggi nei formati definiti da IBM MQ (noti anche come formati integrati) possono essere convertiti dal gestore code da una serie di caratteri codificati a un'altra, a condizione che entrambe le serie di caratteri siano relative a una singola lingua o a un gruppo di lingue simili.

Ad esempio, è supportata la conversione tra serie di caratteri codificati con identificativi (CCSID) 850 e 500, poiché entrambe si applicano alle lingue dell'Europa occidentale.

Per le conversioni di caratteri NL (newline) EBCDIC in ASCII, consultare Tutte le stanze dei gestori code del file mqs.ini e la variabile di ambiente **AMQ_CONVEBCDICNEWLINE** .

Le conversioni supportate sono definite in Elaborazione conversione dati.

La conversione tra CCSID 37 e 500 è supportata su IBM MQ Appliance, Windows, Linuxe macOS.

Quando un gestore code non può convertire i messaggi in formati integrati

Il gestore code non può convertire automaticamente i messaggi in formati integrati se i relativi CCSID rappresentano gruppi di lingue nazionali differenti. Ad esempio, la conversione tra CCSID 850 e CCSID 1025 (che è una serie di caratteri codificati EBCDIC per le lingue che utilizzano lo script cirillico) non è supportata perché molti dei caratteri in una serie di caratteri codificati non possono essere rappresentati nell'altra. Se si dispone di una rete di gestori code che lavorano in lingue nazionali differenti e la conversione dei dati tra alcune serie di caratteri codificati non è supportata, è possibile abilitare una conversione predefinita.

Per le piattaforme a cui si applica `ccsid_part2.tbl` , consultare “Specifica della conversione dati predefinita” a pagina 209 utilizzo `ccsid_part2.tbl` per ulteriori informazioni. La conversione dati predefinita su piattaforme diverse da quelle a cui si applica il file `ccsid_part2.tbl` è descritta in “Conversione dati predefinita” a pagina 207.

Supporto di conversione dati Unicode avanzato

Il prodotto supporta tutti i caratteri Unicode definiti nello standard Unicode 8.0 nella conversione dati. Ciò include il supporto completo per UTF-16, incluse le coppie surrogate (una coppia di caratteri UTF-16 a 2 byte nell'intervallo compreso tra `X'D800'` e `X'DFFF'` che rappresentano un punto di codice Unicode sopra U+FFFF).

La combinazione delle sequenze di caratteri è supportata anche nei casi in cui un carattere precomposto in un CCSID viene messo in corrispondenza con una sequenza di caratteri di combinazione in un altro CCSID.

La conversione dati da e verso Unicode e CCSID 1388, 1390, 1399, 4933, 5488 e 16884 è stata estesa, su alcune piattaforme, per supportare tutti i punti di codice attualmente definiti per questi CCSID, inclusi quelli che si associano ai punti di codice in piani supplementari Unicode.

Nel caso dei CCSID 1390, 1399 e 16884, sono inclusi i caratteri definiti nello standard JIS X 0213 (JIS2004).

È stato inoltre aggiunto il supporto per la conversione da e verso Unicode e sei nuovi CCSID (da 1374 a 1379).

File ccsid_part2.tbl

Viene fornito un altro file, `ccsid_part2.tbl`.

Il file `ccsid_part2.tbl` ha la precedenza sul file `ccsid.tbl` e:

- Consente di aggiungere o modificare voci CCSID
- Specificare la conversione dati predefinita
- Specificare i dati per i diversi livelli di comando

`ccsid_part2.tbl` è applicabile solo alle seguenti piattaforme:

- Linux - tutte le versioni
- Windows

Windows Su IBM MQ for Windows, `ccsid_part2.tbl` si trova nella directory *MQDataRoot*`\conv\table` per impostazione predefinita. Inoltre, su IBM MQ for Windows registra tutti i codeset supportati.

Linux Su IBM MQ for Linux, `ccsid_part2.tbl` si trova nella directory *MQDataRoot*`/conv/tablee` i codeset supportati sono contenuti nelle tabelle di conversione fornite da IBM MQ.

Sebbene il file `ccsid_part2.tbl` sostituisca il file `ccsid.tbl` esistente utilizzato nelle versioni precedenti di IBM MQ per fornire ulteriori informazioni CCSID, il file `ccsid.tbl` continua ad essere analizzato da IBM MQ e pertanto non deve essere eliminato.

Per ulteriori informazioni, consultare “Il file ccsid_part2.tbl” a pagina 207.

`ccsid.tbl` file

Su piattaforme diverse da quelle a cui si applica `ccsid_part2.tbl` , il file `ccsid.tbl` viene utilizzato per i seguenti scopi:

- AIX Su AIX, i codeset supportati vengono mantenuti internamente dal sistema operativo.
- Specifica eventuali codeset aggiuntivi. Per specificare ulteriori codeset, è necessario modificare `ccsid.tbl` (le istruzioni su come eseguire questa operazione sono fornite nel file).
- Specifica qualsiasi conversione dati predefinita.

È possibile aggiornare le informazioni registrate in `ccsid.tbl`; è possibile eseguire questa operazione se, ad esempio, una release futura del sistema operativo supporta ulteriori serie di caratteri codificati.

Conversione dati predefinita

Se si impostano i canali tra due macchine su cui la conversione dati non è normalmente supportata, è necessario abilitare la conversione dati predefinita per il funzionamento dei canali.

Su piattaforme diverse da quelle a cui si applica `ccsid_part2.tbl` , per abilitare la conversione dati predefinita, modificare il file `ccsid.tbl` per specificare un CCSID EBCDIC predefinito e un CCSID ASCII predefinito. Le istruzioni su come eseguire questa operazione sono incluse nel file. È necessario effettuare questa operazione su tutte le macchine che verranno connesse utilizzando i canali. Riavviare il gestore code per rendere effettive le modifiche.

Il processo di conversione dati predefinito è il seguente:

- Se la conversione tra CCSID di origine e di destinazione non è supportata, ma i CCSID degli ambienti di origine e di destinazione sono entrambi EBCDIC o ASCII, i dati carattere vengono passati all'applicazione di destinazione senza conversione.
- Se un CCSID rappresenta una serie di caratteri codificati ASCII, e l'altro rappresenta una serie di caratteri codificati EBCDIC, IBM MQ converte i dati utilizzando i CCSID di conversione dati predefiniti definiti in ccsid.tbl.

Nota: Tentare di limitare i caratteri che vengono convertiti a quelli che hanno gli stessi valori di codice nella serie di caratteri codificati specificata per il messaggio e nella serie di caratteri codificati predefinita. Se si utilizza solo la serie di caratteri valida per i nomi oggetto IBM MQ (come definito in Naming IBM MQ objects) in generale, si soddisferà questo requisito. Le eccezioni si verificano con CCSID EBCDIC 290, 930, 1279 e 5026 utilizzati in Giappone, dove i caratteri minuscoli hanno codici diversi da quelli utilizzati in altri CCSID EBCDIC.

Conversione dei messaggi in formati definiti dall'utente

Il gestore code non è in grado di convertire i messaggi in formati definiti dall'utente da una serie di caratteri codificati ad un'altra. Se è necessario convertire i dati in un formato definito dall'utente, è necessario fornire un'uscita di conversione dati per ciascun formato. Non utilizzare CCSID predefiniti per convertire i dati carattere in formati definiti dall'utente. Per ulteriori informazioni sulla conversione dei dati in formati definiti dall'utente e sulla scrittura delle uscite di conversione dati, consultare Scrittura delle uscite di conversione dati.

Modifica del CCSID del gestore code

Quando è stato utilizzato l'attributo **CCSID** del comando **ALTER QMGR** per modificare il CCSID del gestore code, arrestare e riavviare il gestore code per garantire che tutte le applicazioni in esecuzione, inclusi il server dei comandi e i programmi del canale, vengano arrestati e riavviati.

Ciò è necessario poiché tutte le applicazioni in esecuzione quando il CCSID del gestore code viene modificato continuano ad utilizzare il CCSID esistente.

Il file `ccsid_part2.tbl`

Il file `ccsid_part2.tbl` viene utilizzato per fornire ulteriori informazioni CCSID. Il file `ccsid_part2.tbl` sostituisce il file `ccsid.tbl` utilizzato prima di IBM MQ 9.0.

Nota: Il file `ccsid.tbl` , utilizzato prima di IBM MQ 9.0 per fornire ulteriori informazioni CCSID, continua ad essere analizzato da IBM MQ e non deve essere eliminato. Tuttavia, le voci in `ccsid_part2.tbl` hanno la precedenza sulle altre voci in `ccsid.tbl`.

Si consiglia di utilizzare `ccsid_part2.tbl` invece di `ccsid.tbl` perché `ccsid_part2.tbl`:

- Contiene il supporto per i valori di codifica Unicode. Da IBM MQ 9.0, il prodotto supporta tutti i caratteri Unicode definiti nello standard Unicode 8.0 nella conversione dei dati, incluso il supporto completo per UTF-16. Per ulteriori informazioni, consultare “Conversione dati tra serie di caratteri codificati” a pagina 205.
- Consente di specificare la versione delle voci CCSID, in modo che le voci siano applicabili solo ai livelli di comandi selezionati.

È possibile utilizzare il file `ccsid_part2.tbl` per:

- Aggiungere o modificare voci CCSID
- Specificare la conversione dati predefinita
- Specificare i dati per i diversi livelli di comando

Il file `ccsid_part2.tbl` è applicabile solo alle piattaforme seguenti:

- Linux - tutte le versioni
- Windows

L'ubicazione del file `ccsid_part2.tbl` dipende dalla piattaforma:

- Linux La directory *MQDataRoot*`/conv/table` su tutte le versioni di Linux.
- Windows La directory *MQDataRoot*`\conv\table` su Windows.

Aggiunta o modifica delle voci CCSID

Una voce del file `ccsid_part2.tbl` ha il formato seguente:

```
<CCSID number> <Base CCSID> <DBCS CodePage> <SBCS CodePage>
<Type> <Encoding> <ACRI> <Name>
```

Una voce di esempio per CCSID 1200 (UTF-16) è:

```
  1200 1200 1200 1200 3 8 0 UTF-16
```

Nota: Per ulteriori dettagli sul valore per ACRI, vedere il commento nel file `ccsid_part2.tbl` .

Nel formato `ccsid_part2.tbl` :

Il tipo può essere:

1=SBCS

2=DBCS

3=MBCS

La codifica può essere uguale a:

1=EBCDIC

2 = ASCII

3 = ISO

4 = UCS-2

5 = UTF-8

6 = Euc

7 = GB18030

8 = UTF-16

9 = UTF-32

Quando si modifica il file:

- È possibile specificare un commento utilizzando il simbolo # all'inizio di una riga. Ciò impedisce a IBM MQ di tentare di analizzare la linea.
- Impossibile fornire commenti in linea.
- Assicurarsi di non creare righe vuote.
- Non aggiungere nuove voci alla fine del file.

  Le nuove voci CCSID devono essere aggiunte prima delle informazioni della tabella ACRI.

Specifica della conversione dati predefinita

È possibile definire i CCSID di conversione predefiniti, che vengono utilizzati per la conversione tra CCSID ASCII o simili e CCSID EBCDIC, se non è supportata alcuna conversione tra due CCSID.

Se si abilita questa funzione, la conversione predefinita viene utilizzata per la trasmissione e le intestazioni del messaggio e può essere utilizzata anche nella conversione dei dati utente.

Le conversioni predefinite vengono abilitate creando due linee simili alle seguenti:

```
default       0      500     1     1    0
default       0      850     1     2    0
```

La prima riga imposta il valore predefinito per i CCSID EBCDIC su 500 e la seconda riga imposta il valore predefinito per i CCSID ASCII e simili su 850.

Specifica di dati per livelli di comando diversi

Per specificare le voci CCSID per i diversi livelli di comandi di IBM MQ , utilizzare un simbolo di due punti seguito dal livello di comando (o livelli di comando) di IBM MQ a cui si desidera applicare la sezione successiva.

Il numero rappresenta il livello di comando minimo in cui deve essere eseguito il gestore code o il client. Ad esempio, se il gestore code corrente è il livello di comando 900 e rileva un indicatore di livello di comando 800 o 900, i CCSID vengono letti.

Tuttavia, un gestore code al livello 800 ignora i CCSID nella sezione 900.

Il livello di comando specificato è applicabile a tutte le voci CCSID incontrate dopo un indicatore del livello di comando, fino a quando non viene trovato un nuovo indicatore del livello di comando.

Se si richiede di impostare il livello di comando su tutti i livelli di comando, specificare il numero zero.

Durante la prima analisi di `ccsid_part2.tbl`, IBM MQ considera tutti i CCSID rilevati come validi per tutti i livelli di comando di IBM MQ.

Il controllo delle versioni inizia ad essere utilizzato solo quando IBM MQ rileva il primo indicatore di livello di comando.

Il seguente frammento di codice mostra un esempio di utilizzo della versione:

```
# 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
```

AmministrazioneManaged File Transfer

Utilizzare i comandi Managed File Transfer per gestire Managed File Transfer. È anche possibile utilizzare IBM MQ Explorer per alcune attività di amministrazione.

Avvia trasferimento inserendo un messaggio in una coda comandi dell'agent

È anche possibile avviare un trasferimento file inserendo un messaggio di trasferimento file nella coda comandi dell'agent origine. Un nome coda comandi di esempio è `SYSTEM.FTE.COMMAND.AGENT01`. È necessario assicurarsi che il messaggio raggiunga la coda comandi dell'agent di origine corretto; se il messaggio viene ricevuto da un agent che non corrisponde alle informazioni di origine nell'XML, il messaggio viene rifiutato.

L'XML della richiesta di trasferimento deve essere conforme allo schema `FileTransfer.xsd` e utilizzare l'elemento < request> come elemento root. Consultare Formato del messaggio di richiesta di trasferimento file per informazioni sulla struttura e sul contenuto di un messaggio di richiesta di trasferimento. Il modo in cui si inserisce il messaggio di richiesta di trasferimento su una coda comandi dell'agente è specifico dell'attività. Ad esempio, è possibile utilizzare l'API IBM MQ Java per inserire un messaggio nella coda in modo programmatico.

Concetti correlati

“Trasferimento dei dati dai file ai messaggi” a pagina 265
È possibile utilizzare la funzione file - to - message di Managed File Transfer per trasferire i dati da un file a un singolo messaggio o a più messaggi su una coda IBM MQ .

“Trasferimento dei dati dai messaggi ai file” a pagina 280
La funzione messaggio - a - file di Managed File Transfer consente di trasferire i dati da uno o più messaggi su una coda IBM MQ a un file, a un dataset (su z/OS) o a uno spazio file utente. Se si dispone di un'applicazione che crea o elabora messaggi IBM MQ , è possibile utilizzare la funzionalità messaggio - a - file di Managed File Transfer per trasferire questi messaggi a un file su qualsiasi sistema nella rete Managed File Transfer .

“Il bridge di protocollo” a pagina 291
Il bridge di protocollo consente alla rete Managed File Transfer (MFT) di accedere ai file memorizzati su un server di file esterno alla rete MFT , nel dominio locale o in un'ubicazione remota. Questo server di file può utilizzare i protocolli di rete FTP, FTPS o SFTP. Ogni server di file richiede almeno un agent dedicato. L'agent dedicato è noto come agent bridge di protocollo. Un agent bridge può interagire con più server di file.

“Utilizzo di MFT da IBM Integration Bus” a pagina 330
È possibile utilizzare Managed File Transfer da IBM Integration Bus utilizzando i nodi FTEOutput e FTEInput.

“Ripristino e riavvio di MFT” a pagina 330
Se l'agent o il gestore code non sono disponibili per qualsiasi motivo, ad esempio a causa di un errore di alimentazione o di rete, Managed File Transfer esegue il ripristino come riportato di seguito in questi scenari:

Attività correlate

“Avvio di un agent MFT” a pagina 211
Prima di poter utilizzare un agent Managed File Transfer per un trasferimento file, è necessario avviare l'agent.

“Avvio di un nuovo trasferimento file” a pagina 218
È possibile avviare un nuovo trasferimento file da IBM MQ Explorer o dalla riga comandi ed è possibile scegliere di trasferire un singolo file o più file in un gruppo.

“Monitoraggio dei trasferimenti file in corso” a pagina 225
È possibile monitorare un trasferimento file in corso utilizzando la scheda **Managed File Transfer - Avanzamento trasferimento corrente** in IBM MQ Explorer. Questo trasferimento file può essere avviato da IBM MQ Explorer o dalla riga comandi. La scheda visualizza anche l'avanzamento dei trasferimenti pianificati nel momento in cui iniziano i trasferimenti pianificati.

“Visualizzazione dello stato dei trasferimenti file nel log di trasferimento” a pagina 227
È possibile visualizzare i dettagli dei trasferimenti file utilizzando il **Log trasferimenti** in IBM MQ Explorer. Questi possono essere trasferimenti avviati dalla riga comandi o da IBM MQ Explorer. È anche possibile personalizzare quanto visualizzato nel **Log di trasferimento**.

“Monitoraggio delle risorse MFT” a pagina 229

È possibile monitorare le risorse Managed File Transfer ; ad esempio, una coda o una directory. Quando viene soddisfatta una condizione su questa risorsa, il monitoraggio risorse avvia un'attività, ad esempio un trasferimento file. È possibile creare un monitoraggio delle risorse utilizzando il comando **fteCreateMonitor** o la vista **Monitor** nel plug-in Managed File Transfer per IBM MQ Explorer.

"Utilizzo dei modelli di trasferimento file" a pagina 262
È possibile utilizzare i modelli di trasferimento file per memorizzare le impostazioni di trasferimento file comuni per trasferimenti ripetuti o complessi. Creare un modello di trasferimento dalla riga comandi utilizzando il comando **fteCreateTemplate** oppure utilizzare IBM MQ Explorer per creare un modello di trasferimento utilizzando la procedura guidata **Crea nuovo modello per il trasferimento file gestito** oppure salvare un modello mentre si sta creando un trasferimento file selezionando la casella di spunta **Salva impostazioni di trasferimento come modello** . La finestra **Template di trasferimento** visualizza tutti i template di trasferimento che sono stati creati nella rete Managed File Transfer .

"Elenco di agenti di MFT" a pagina 216
È possibile elencare gli agenti Managed File Transfer registrati con un particolare gestore code utilizzando la riga comandi o il IBM MQ Explorer.

"Arresto di un agent MFT" a pagina 217
È possibile arrestare un agent Managed File Transfer dalla riga comandi. Quando si arresta un agent, si sta disattivando l'agent e si consente all'agent di completare il trasferimento file corrente prima dell'arresto. È anche possibile specificare il parametro **-i** nella riga comandi per arrestare immediatamente un agente. Quando l'agent è stato arrestato, non è possibile utilizzare tale agent per trasferire i file fino a quando non viene riavviato.

Configurazione di un programma di registrazione MFT

Riferimenti correlati

Linee guida per il trasferimento di file

Avvio di un agent MFT

Prima di poter utilizzare un agent Managed File Transfer per un trasferimento file, è necessario avviare l'agent.

Informazioni su questa attività

È possibile avviare un Managed File Transfer Agent dalla riga comandi. In questo caso, il processo agent si arresta quando si scollega il sistema.

ALW Su AIX, Linux, and Windows, è possibile configurare un agent in modo che continui l'esecuzione quando ci si scollega dal sistema e si può continuare a ricevere trasferimenti file.

z/OS Su z/OS, è possibile configurare l'agent in modo che venga avviato come attività avviata da JCL senza la necessità di una sessione interattiva.

Tenere presente che, se un agent rileva un errore irreversibile durante l'esecuzione, viene generata una FDC (first failure data capture) e l'agent viene arrestato.

Procedura

- Per avviare un agente dalla riga comandi, utilizzare il comando **fteStartAgent** .
  Per ulteriori informazioni, consultare **fteStartAgent**.
- ALW
  Per configurare un agente in modo che continui l'esecuzione quando ci si scollega dal sistema:
  - Windows Su Windows, configurare l'agente da eseguire come servizio Windows . Per ulteriori informazioni, consultare "Avvio di un agent MFT come servizio Windows" a pagina 212.

– Linux AIX Su AIX and Linux, configurare l'agent in modo che si avvii automaticamente durante un riavvio utilizzando un file script. Per ulteriori informazioni, consultare “Avvio di un agent MFT all'avvio del sistema AIX and Linux” a pagina 214.

- z/OS
  Su z/OS, configurare l'agente in modo che venga avviato come attività avviata da JCL senza la necessità di una sessione interattiva.
  Per ulteriori informazioni, consultare “Starting an MFT agent on z/OS” a pagina 216.

Attività correlate

“Elenco di agenti di MFT” a pagina 216
È possibile elencare gli agenti Managed File Transfer registrati con un particolare gestore code utilizzando la riga comandi o il IBM MQ Explorer.

“Arresto di un agent MFT” a pagina 217
È possibile arrestare un agent Managed File Transfer dalla riga comandi. Quando si arresta un agent, si sta disattivando l'agent e si consente all'agent di completare il trasferimento file corrente prima dell'arresto. È anche possibile specificare il parametro **-i** nella riga comandi per arrestare immediatamente un agente. Quando l'agent è stato arrestato, non è possibile utilizzare tale agent per trasferire i file fino a quando non viene riavviato.

Riferimenti correlati

Valori di stato dell'agente MFT
fteStartAgent

Windows Avvio di un agent MFT come servizio Windows

È possibile avviare un agent come servizio Windows in modo che quando ci si scollega Windows, l'agent continua l'esecuzione e può ricevere trasferimenti file.

Informazioni su questa attività

Su Windows, quando si avvia un agent dalla riga comandi, il processo agent viene eseguito utilizzando il nome utente utilizzato per accedere a Windows. Quando si scollega il sistema, il processo agent si arresta. Per impedire l'arresto dell'agent, è possibile configurarne l'esecuzione come servizio Windows . L'esecuzione come servizio Windows ti consente anche di configurare gli agent in modo che vengano avviati automaticamente quando l'ambiente Windows viene avviato o riavviato.

Completare la seguente procedura per avviare un agent che viene eseguito come servizio Windows . Devi eseguire Managed File Transfer su una delle versioni Windows supportate per eseguire l'agent come servizio Windows . Per un elenco degli ambienti supportati, fare riferimento a Requisiti di sistema per IBM MQ.

I passi esatti dipendono dal fatto che si sia già creato un agente o che si stia creando un agente. Entrambe le opzioni sono descritte nei seguenti passi.

Procedura

1. Se si sta creando un agente Managed File Transfer , utilizzare il comando **fteCreateAgent**, **fteCreateCDAgent**o **fteCreateBridgeAgent** . Specificare il parametro **-s** per eseguire l'agente come servizio Windows . Nel seguente esempio, viene creato l'agent AGENT1 , che ha un gestore code agent QMGR1. Il servizio Windows viene eseguito utilizzando un nome utente `fteuser`, a cui è associata una password `ftepassword`.

   ```
   fteCreateAgent -agentName AGENT1 -agentQMgr QMGR1 -s -su fteuser -sp ftepassword
   ```

   Facoltativamente, è possibile specificare un nome per il servizio dopo il parametro **-s** . Se non si specifica un nome, il servizio viene denominato `mqmftAgent`*AGENTQMGR*, dove *AGENT* è il nome

dell'agent specificato e *QMGR* è il nome del gestore code dell'agent. In questo esempio, il nome predefinito per il servizio è `mqmftAgentAGENT1QMGR1`.

Nota: L'account utente Windows specificato utilizzando il parametro **-su** deve disporre dei diritti **Log on as a service** . Per informazioni su come configurarlo, vedi Troubleshooting a MFT agent o logger running as a Windows service.

Per ulteriori informazioni, consultare fteCreateAgent, fteCreateCDAgent: create a Connect:Direct bridge agentoppure fteCreateBridgeAgent (create and configure an MFT protocol bridge agent).

2. Se è stato seguito il passo precedente per creare un agent, eseguire i comandi MQSC generati dal comando **fteCreateAgent**, **fteCreateCDAgent**o **fteCreateBridgeAgent** . Questi comandi creano le code IBM MQ necessarie all'agente.

   Ad esempio, per un agent denominato *AGENT1*, un gestore code agent denominato *QMGR1* e un gestore code di coordinamento denominato *COORDQMGR1*, eseguire il seguente comando:

   ```
   runmqsc QMGR1 MQ_DATA_PATH\mqft\config\COORDQMGR1\agents\AGENT1\AGENT1_create.mqsc
   ```

3. Se non sono stati seguiti i passi precedenti per creare un agent e si desidera invece configurare un agent esistente per l'esecuzione come servizio Windows , arrestare prima l'agent, se è in esecuzione, quindi modificarne la configurazione.

   a) Il seguente esempio utilizza un agente denominato AGENT1. Esegui il seguente comando:

   ```
   fteStopAgent AGENT1
   ```

   b) Utilizzare il comando **fteModifyAgent** per configurare l'agente da eseguire come servizio Windows :

   ```
   fteModifyAgent -agentName AGENT1 -s -su fteuser -sp ftepassword
   ```

   Per ulteriori informazioni, consultare fteModifyAgent: eseguire un agent MFT come un Windows servizio.

4. Avviare l'agent utilizzando il comando **fteStartAgent** . In alternativa, è possibile utilizzare lo strumento Servizi Windows , disponibile da Strumenti di amministrazione nel pannello di controllo, selezionato dal menu di avvio del desktop Windows , per avviare il servizio.

   ```
   fteStartAgent AGENT1
   ```

   Il servizio continua ad essere eseguito anche se ci si scollega da Windows. Per garantire che il servizio venga riavviato anche quando Windows viene riavviato dopo un arresto, il campo **Tipo di avvio** nello strumento Servizi Windows è impostato su **Automatico** per impostazione predefinita. Modificare questa opzione in **Manuale** se non si desidera che il servizio venga riavviato al riavvio di Windows .

5. Opzionale: Per arrestare l'agent, utilizzare il comando fteStopAgent oppure utilizzare lo strumento Windows Services. Ad esempio, dalla riga comandi, immetti il seguente comando:

   ```
   fteStopAgent AGENT1
   ```

   - Quando esegui il comando **fteStopAgent** come un servizio, il comando viene sempre eseguito utilizzando il parametro **-i** indipendentemente dal fatto che tu abbia specificato o meno questo parametro. Il parametro **-i** arresta l'agent immediatamente senza completare i trasferimenti in corso. Ciò è causato da una limitazione del servizio Windows .

Operazioni successive

Se hai problemi ad avviare il tuo servizio Windows , vedi Risoluzione dei problemi di un agent o di un programma di registrazione MFT in esecuzione come un servizio Windows. Questo argomento descrive anche l'ubicazione dei file di log di servizio Windows .

Attività correlate

“Elenco di agenti di MFT” a pagina 216
È possibile elencare gli agenti Managed File Transfer registrati con un particolare gestore code utilizzando la riga comandi o il IBM MQ Explorer.

“Arresto di un agent MFT” a pagina 217
È possibile arrestare un agent Managed File Transfer dalla riga comandi. Quando si arresta un agent, si sta disattivando l'agent e si consente all'agent di completare il trasferimento file corrente prima dell'arresto. È anche possibile specificare il parametro **-i** nella riga comandi per arrestare immediatamente un agente. Quando l'agent è stato arrestato, non è possibile utilizzare tale agent per trasferire i file fino a quando non viene riavviato.

Riferimenti correlati

fteCreateAgent (crea un agent MFT)

fteCreateCDAgent (crea un agent bridge Connect:Direct)

fteCreateBridgeAgent (creazione e configurazione di un agent bridge di protocollo MFT)

fteModifyAgent (esegue un agent MFT come servizio Windows)

Informazioni correlate

Il file MFT `agent.properties`

Linux AIX Avvio di un agent MFT all'avvio del sistema AIX and Linux

Un Managed File Transfer Agent può essere configurato per essere avviato all'avvio del sistema su AIX and Linux. Quando ci si scollega, l'agent continua l'esecuzione e può ricevere trasferimenti file.

Una volta creato e configurato un agent utilizzando uno di tali comandi Managed File Transfer ; **fteCreateAgent**, **fteCreateCDAgent**o **fteCreateBridgeAgent**, è possibile configurarlo in modo che venga avviato automaticamente durante un riavvio su macchine AIX and Linux utilizzando un file di script che esegue semplicemente il seguente comando:

```
su -l mqmft_user -c mq_install_root/bin/fteStartAgent agent_name
```

Dove *mq_install_root* è la directory root dell'installazione Managed File Transfer richiesta, il valore predefinito è: `/opt/mqm` e *agent_name* è il nome del Managed File Transfer Agent da avviare. L'utilizzo di questo file script varia a seconda del sistema operativo specifico. Ad esempio, ci sono ulteriori opzioni disponibili in Linux.

Linux

Linux

Per sistemi Linux esistono diversi modi per avviare le applicazioni durante il processo di avvio del sistema. In generale, considerare le seguenti operazioni:

1. Creare un file denominato `/etc/rc.mqmft` con contenuto:

   ```
   #!/bin/sh
   su -l mqmft_user"-c mq_install_root/bin/fteStartAgent agent_name"
   ```

   Dove *mqmft_user* è l'ID utente con cui deve essere eseguito il processo dell'agent. Questo ID utente deve essere un membro del gruppo mqm.
2. Rendere il file eseguibile, ad esempio:

   ```
   chmod 755 /etc/rc.mqmft
   ```

3. Aggiungere quindi la riga seguente a /etc/inittab:

```
mqmft:5:boot:/etc/rc.mqmft
```

Altri modi per avviare un agent durante l'avvio su Linux includono l'aggiunta di righe di script al file /etc/rc.d/rc.local o su Linux SuSe, aggiungendo righe di script al file /etc/init.d/boot.local . È necessario selezionare il metodo che funziona meglio per il proprio ambiente. Di seguito sono riportate ulteriori informazioni su altri modi per avviare un agente durante l'avvio su specifiche distribuzioni Linux supportate:

SLES 10 e 11

Per sistemi SUSE Linux Enterprise Server (SLES) 10 e 11, attenersi alla seguente procedura:

1. Come ID utente root di sistema, creare il file /etc/init.d/rc.rclocal .
2. Aggiungere le seguenti righe al file 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. Eseguire i seguenti comandi:

```
chmod 755 rc.rclocal

chkconfig --add rc.rclocal
```

Avvio degli agent Managed File Transfer su Linux con systemd

Linux

Effettuare la seguente procedura:

1. Creare un file nella cartella di sistema /etc/systemd/ e denominarlo, ad esempio <agentname>.service. Aggiungere il seguente contenuto, dove <agentname> è 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. Per abilitare il servizio, eseguire i seguenti comandi:

```
# systemctl enable MFT_AGT_LNX_0
# systemctl daemon-reload
```

3. Per avviare l'agent e controllarne lo stato, eseguire i seguenti comandi:

```
# systemctl start MFT_AGT_LNX_0
# systemctl status MFT_AGT_LNX_0
```

Attività correlate

“Arresto di un agent MFT” a pagina 217
È possibile arrestare un agent Managed File Transfer dalla riga comandi. Quando si arresta un agent, si sta disattivando l'agent e si consente all'agent di completare il trasferimento file corrente prima dell'arresto. È anche possibile specificare il parametro **-i** nella riga comandi per arrestare immediatamente un agente. Quando l'agent è stato arrestato, non è possibile utilizzare tale agent per trasferire i file fino a quando non viene riavviato.

Riferimenti correlati

Agent fteCreate

fteCreateCDAgent: crea un agent bridge Connect:Direct

fteCreateBridgeAgent (creazione e configurazione di un agent bridge di protocollo 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 217
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

Elenco di agenti di MFT

È possibile elencare gli agenti Managed File Transfer registrati con un particolare gestore code utilizzando la riga comandi o il IBM MQ Explorer.

Informazioni su questa attività

Per elencare gli agent utilizzando la riga comandi, consultare fteListComando degli agent.

Per elencare gli agenti che utilizzano IBM MQ Explorer, nella vista Navigator fare clic su **Agenti** sotto il nome del gestore code di coordinamento.

Se un agent non è elencato dal comando **fteListAgents** o non viene visualizzato in IBM MQ Explorer, utilizzare il diagramma di flusso di diagnosi nel seguente argomento per individuare e correggere il problema: Cosa fare se l'agent MFT non viene elencato dal comando **fteListAgents**.

Riferimenti correlati

fteListAgent: elenca gli agent di MFT per un gestore code di coordinamento

Valori di stato dell'agent MFT

fteShowAgentDetails

Arresto di un agent MFT

È possibile arrestare un agent Managed File Transfer dalla riga comandi. Quando si arresta un agent, si sta disattivando l'agent e si consente all'agent di completare il trasferimento file corrente prima dell'arresto. È anche possibile specificare il parametro **-i** nella riga comandi per arrestare immediatamente un agente. Quando l'agent è stato arrestato, non è possibile utilizzare tale agent per trasferire i file fino a quando non viene riavviato.

Prima di iniziare

Se si desidera controllare i nomi degli agenti associati ad un gestore code, è possibile elencare gli agenti utilizzando IBM MQ Explorer o la riga comandi, consultare Comando degli agentifteList.

Informazioni su questa attività

Per arrestare un agent dalla riga comandi, consultare fteStopAgent.

Se un agent viene arrestato in modo controllato utilizzando **fteStopAgent**, l'agent non accetta alcuna nuova richiesta di trasferimento gestito e attende il completamento di eventuali trasferimenti in corso prima di arrestarsi. Da IBM MQ 9.3.0, per mostrare che l'agent è ancora in uno stato transitorio, e quindi non è stato ancora arrestato e non può essere riavviato, l'agent passa allo stato STOPPING fino al completamento di eventuali trasferimenti in corso. Questo stato viene visualizzato nell'emissione dei comandi di **fteListAgents** e **fteShowAgentDetails** , in Query MFT REST APIe nella vista **Agenti** di IBM MQ Explorerdel plugin MFT .

Windows Se il proprio agent è stato configurato per essere eseguito come servizio Windows , l'esecuzione del comando **fteStopAgent** arresta anche il servizio Windows . In alternativa, è possibile arrestare l'agent arrestandolo utilizzando lo strumento Servizi di Windows . Per ulteriori informazioni, consultare l'argomento “Avvio di un agent MFT come servizio Windows” a pagina 212.

Attività correlate

“Avvio di un agent MFT” a pagina 211
Prima di poter utilizzare un agent Managed File Transfer per un trasferimento file, è necessario avviare l'agent.

Riferimenti correlati

Valori di stato dell'agente MFT

Agent fteStop

“Stopping an MFT agent on z/OS” a pagina 217
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 216

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

Avvio di un nuovo trasferimento file

È possibile avviare un nuovo trasferimento file da IBM MQ Explorer o dalla riga comandi ed è possibile scegliere di trasferire un singolo file o più file in un gruppo.

Informazioni su questa attività

Per avviare un nuovo trasferimento file dalla riga comandi, consultare fteCreateComando di trasferimento.

Per iniziare un nuovo trasferimento file utilizzando il wizard **Crea nuovo trasferimento file gestito** in IBM MQ Explorer, effettuare le seguenti operazioni:

Procedura

1. Nella vista Navigator , fare clic su **Managed File Transfer**. **Managed File Transfer Central** viene visualizzato nella vista Contenuto.
2. Tutti i gestori code di coordinamento vengono visualizzati nella vista Navigator . Espandere il nome del gestore code di coordinamento per cui è registrato l'agent che si desidera utilizzare per il trasferimento. Se si è attualmente connessi a un gestore code di coordinamento diverso da quello che si desidera utilizzare per il trasferimento, fare clic con il tasto destro del mouse sul nome del gestore code di coordinamento nella vista Navigator e fare clic su **Disconnetti**. Quindi, selezionare con il tasto destro del mouse il nome del gestore code di coordinamento che si desidera utilizzare e fare clic su **Connetti**.
3. Avviare la procedura guidata **Crea nuovo trasferimento file gestito** utilizzando uno dei metodi seguenti:
   a) Fare clic con il tasto destro del mouse sul nome di uno dei seguenti nodi nella vista Navigator : il gestore code di coordinamento pertinente, **Modelli di trasferimento**, **Log di trasferimento**o **Trasferimenti in sospeso**. Quindi fare clic su **Nuovo trasferimento** per avviare la procedura guidata.
   b) Fare clic su **File** > **Nuovo** > **Altro** > **Managed File Transfer Wizard** > **Nuova procedura guidata di trasferimento**
4. Seguire le istruzioni sui pannelli della procedura guidata. È inoltre disponibile una guida sensibile al contesto per ciascun pannello. Per accedere alla guida sensibile al contesto su Windows, premere F1. Su Linux, premere `Ctrl+F1` o `Shift+F1`.

Concetti correlati

“Utilizzo dei file di definizione del trasferimento” a pagina 220
È possibile specificare un file di definizione trasferimento che può essere utilizzato per creare un trasferimento file. Il file di definizione del trasferimento è un file XML che definisce alcune o tutte le informazioni richieste per creare il trasferimento.

Attività correlate

“Creazione di un trasferimento file pianificato” a pagina 222
È possibile pianificare un nuovo trasferimento file da IBM MQ Explorero dalla riga comandi. Il trasferimento pianificato può contenere singoli file o più file in un gruppo. È possibile eseguire un trasferimento file pianificato una sola volta o ripetere il trasferimento più volte.

“Attivazione di un trasferimento file” a pagina 224
È possibile impostare determinate condizioni di trigger su un trasferimento file che devono essere true prima che tale trasferimento possa essere effettuato. Se le condizioni di attivazione non sono true, il trasferimento file non viene eseguito e un messaggio di log viene facoltativamente inoltrato per registrare il fatto che il trasferimento non si è verificato. La richiesta di trasferimento file viene quindi eliminata. Ad esempio, è possibile impostare un trasferimento file che si verifica solo se un file denominato sul sistema in cui si trova l'agent di origine è superiore a una dimensione specificata o se un particolare file denominato esiste sul sistema in cui si trova l'agent di origine. È possibile impostare un trasferimento file attivato da IBM MQ Explorer o dalla riga comandi.

“Impostazione di una scadenza per il recupero di trasferimenti in stallo” a pagina 331
È possibile impostare un timeout di ripristino del trasferimento per i trasferimenti di file in stallo che si applica a tutti i trasferimenti per un agent di origine. È anche possibile impostare un timeout di ripristino trasferimento per un singolo trasferimento. Se si imposta un periodo di tempo specifico, in secondi, durante il quale un agent di origine tenta di ripristinare un trasferimento file in stallo e il trasferimento non riesce quando l'agent raggiunge il timeout, il trasferimento non riesce.

Riferimenti correlati

`fteCreateTransfer`: avviare un nuovo trasferimento file

Formato del messaggio di richiesta di trasferimento file

Utilizzo dei file di definizione del trasferimento

È possibile specificare un file di definizione trasferimento che può essere utilizzato per creare un trasferimento file. Il file di definizione del trasferimento è un file XML che definisce alcune o tutte le informazioni richieste per creare il trasferimento.

I file di definizione trasferimento sono utili quando si desidera specificare più file di origine e più file di destinazione in una singola operazione di trasferimento. È possibile utilizzare un file di definizione trasferimento per inoltrare un trasferimento file complesso. È possibile riutilizzare e condividere il file di definizione trasferimento.

È possibile utilizzare due formati per un file di definizione del trasferimento e, mentre questi formati variano leggermente, entrambi sono conformi allo schema `FileTransfer.xsd` . È possibile trovare questo schema nella directory `samples\schema` dell'installazione di Managed File Transfer .

Sono supportati i seguenti due formati di file di definizione trasferimento:

- Una definizione dei file di origine e destinazione per un trasferimento. Questa definizione utilizza un elemento **`transferSpecifications`** come root.
- Una definizione dell'intero trasferimento, inclusi i file di origine e di destinazione e gli agenti di origine e di destinazione. Questa definizione utilizza un elemento **`request`** come root.
  - I file con questo formato possono essere generati dal comando **`fteCreateTransfer`** utilizzando il parametro **`-gt`** .

Il seguente esempio mostra un formato file di definizione trasferimento che specifica solo i file di origine e di destinazione per un trasferimento:

```
<?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>
```

Per inoltrare questo formato del file di definizione trasferimento, è necessario specificare gli agente di origine e di destinazione sulla riga comandi:

```
fteCreateTransfer -sa AGENT1 -sm agent1qm -da AGENT2 -dm agent2qm -td
  c:\definitions\example1.xml
```

Il seguente esempio è un formato file di definizione trasferimento che specifica tutte le informazioni richieste per un trasferimento:

```
<?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>
```

È possibile creare un file con questo formato utilizzando il parametro **-gt** sul comando **fteCreateTransfer** . Quando si inoltra un file di definizione trasferimento con questo formato, non è necessario specificare altro sulla riga comandi:

```
fteCreateTransfer -td c:\definitions\example2.xml
```

È possibile sovrascrivere le informazioni dell'agente di origine e di destinazione relative alla riga comandi passando i normali parametri in aggiunta al file di definizione trasferimento. Ad esempio:

```
fteCreateTransfer -da AGENT9 -dm agent9qm -td c:\definitions\example2.xml
```

Questo esempio utilizza le opzioni della riga comandi per sovrascrivere l'agent di destinazione definito all'interno del file di definizione del trasferimento con **AGENT9** e il gestore code di destinazione definito nel file di definizione del trasferimento come **agent9qm**.

Entrambi i formati descritti possono contenere uno o più elementi < item>. Per ulteriori informazioni sull'elemento < item>, consultare Formato del messaggio di richiesta di trasferimento file. Ognuno di questi elementi di trasferimento definisce una coppia di file di origine e di destinazione con attributi aggiuntivi per controllare il comportamento del trasferimento. Ad esempio, è possibile specificare il seguente comportamento:

- Se il trasferimento utilizza un checksum
- Se il trasferimento è di testo o binario
- Indica se eliminare il file di origine dopo il completamento del trasferimento
- Indica se sovrascrivere il file di destinazione se il file esiste

Un vantaggio dell'utilizzo dei file delle definizioni di trasferimento è che è possibile specificare opzioni aggiuntive che non sono disponibili dalla riga comandi. Ad esempio, quando si eseguono trasferimenti da messaggio a file, è possibile specificare l'attributo groupId utilizzando un file di definizione trasferimento. Questo attributo specifica l'ID gruppo IBM MQ dei messaggi letti dalla coda. Un altro vantaggio dei file di definizioni di trasferimento è che è possibile specificare opzioni differenti per ogni coppia di file. Ad esempio, è possibile specificare se viene utilizzato un checksum o se il file viene trasferito in modalità di testo o binaria, file per file. Se si utilizza la riga comandi, si applicano le stesse opzioni per ogni file in un trasferimento.

Ad esempio:

```
  <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 È possibile utilizzare gli elementi per trasferire un file da un sistema distribuito 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 Questo esempio trasferisce il file `textTransferTest.txt` dall'agent di origine al dataset `//TEXT.TRANS.TEST` sull'agent di destinazione in modalità testo. Questo trasferimento converte i dati di origine dalla codifica predefinita dell'agent di origine (non è specificato alcun attributo di codifica di origine) nella codepage: IBM-1047.

Creazione di un trasferimento file pianificato

È possibile pianificare un nuovo trasferimento file da IBM MQ Explorero dalla riga comandi. Il trasferimento pianificato può contenere singoli file o più file in un gruppo. È possibile eseguire un trasferimento file pianificato una sola volta o ripetere il trasferimento più volte.

Informazioni su questa attività

È possibile impostare una pianificazione di trasferimento file che si verifichi una sola volta o che si verifichi ai seguenti intervalli:

- Ogni minuto
- Orario
- Giornaliero
- Settimanale
- Mensile
- Ogni anno

È possibile quindi specificare le ricorrenze da arrestare nei seguenti punti:

- In una data e ora definite
- Dopo un numero definito di ricorrenze

In alternativa, è possibile specificare che le ricorrenze continuino per sempre.

Se un trasferimento pianificato viene eseguito ogni giorno alla stessa ora, utilizzare l'attributo **adjustScheduleTimeForDaylightSaving** nel file delle proprietà dell'agente per modificare l'ora in cui viene eseguita la pianificazione quando cambiano gli orologi. Per ulteriori informazioni, consultare Il file MFT `agent.properties` .

Per creare un nuovo trasferimento file pianificato utilizzando la riga comandi, utilizzare i parametri di pianificazione (**-tb**, **-ss**, **-oi**, **-of**, **-oc**e **-es**) per il comando fteCreateTransfer.

Per creare un nuovo trasferimento file pianificato utilizzando la procedura guidata **Crea nuovo trasferimento file gestito** in IBM MQ Explorer, attenersi alla seguente procedura:

Procedura

1. Nella vista Navigator , fare clic su **Managed File Transfer**. **Managed File Transfer Central** viene visualizzato nella vista Contenuto.
2. Tutti i gestori code di coordinamento vengono visualizzati nella vista Navigator . Espandere il nome del gestore code di coordinamento per cui è registrato l'agent che si desidera utilizzare per il trasferimento. Se si è attualmente connessi a un gestore code di coordinamento diverso da quello che si desidera utilizzare per il trasferimento, fare clic con il tasto destro del mouse sul nome del gestore code di coordinamento nella vista Navigator e fare clic su **Disconnetti**. Quindi, selezionare con il tasto destro del mouse il nome del gestore code di coordinamento che si desidera utilizzare e fare clic su **Connetti**.
3. Avviare la procedura guidata **Crea nuovo trasferimento file gestito** utilizzando uno dei seguenti metodi:
   a) Fare clic con il tasto destro del mouse sul nome di uno dei seguenti nodi nella vista Navigator : il gestore code di coordinamento pertinente, **Modelli di trasferimento**, **Log di trasferimento**o **Trasferimenti in sospeso**. Quindi fare clic su **Nuovo trasferimento** per avviare la procedura guidata.
   b) Fare clic su **File** > **Nuovo** > **Altro** > **Managed File Transfer Wizard** > **Nuova procedura guidata di trasferimento**
4. Seguire le istruzioni sui pannelli della procedura guidata. Accertarsi di selezionare la check box **Abilita trasferimento pianificazione** e immettere i dettagli della pianificazione nella scheda **Pianificazione** . I trasferimenti file pianificati iniziano entro un minuto dall'ora di inizio della pianificazione, se non vi sono problemi che potrebbero influire sul trasferimento. Ad esempio, potrebbero verificarsi problemi con la rete o l'agent che impediscono l'avvio del trasferimento pianificato. Per ogni pannello viene fornita una guida sensibile al contesto. Per accedere alla guida sensibile al contesto su Windows, premere F1. Su Linux, premere `Ctrl+F1` o `Shift+F1`.

Risultati

Per informazioni sui messaggi coinvolti nei trasferimenti file pianificati, vedere Formati dei messaggi di log dei trasferimenti file pianificati.

Gestione trasferimenti file in sospeso

È possibile visualizzare i trasferimenti file pianificati in sospeso da IBM MQ Explorer. La finestra **Trasferimenti in sospeso** visualizza tutti i trasferimenti in sospeso registrati con il gestore code di coordinamento a cui si è attualmente connessi.

Informazioni su questa attività

Per visualizzare lo stato di un trasferimento file pianificato non ancora avviato, utilizzare la seguente procedura:

Procedura

1. Espandere **Managed File Transfer** nella vista Navigator . **Managed File Transfer Central** viene visualizzato nella vista Contenuto.
2. Tutti i gestori code di coordinamento vengono visualizzati nella vista Navigator . Espandere il nome del gestore code di coordinamento utilizzato per il trasferimento pianificato. Se si desidera modificare il gestore code di coordinamento a cui si è connessi, fare clic con il tasto destro del mouse sul nome del gestore code di coordinamento che si desidera utilizzare nella vista Navigator e fare clic su **Connetti**.
3. Fare clic su **Trasferimenti in sospeso**. Viene visualizzata la finestra **Trasferimenti in sospeso** nella vista Contenuto.
4. La finestra **Trasferimenti in sospeso** visualizza i seguenti dettagli sui trasferimenti file pianificati:
   a) **Nome** Il numero del trasferimento file pianificato. Questo numero viene assegnato automaticamente.

b) **Origine** Il nome dell'agent di origine.

c) **File di origine** Il nome del file da trasferire sul sistema host.

d) **Destinazione** Il nome dell'agent di destinazione.

e) **File di destinazione** Il nome del file dopo che è stato trasferito al sistema di destinazione.

f) **Avvio pianificato (fuso orario selezionato)** L'ora e la data in cui è pianificato l'avvio del trasferimento file nel fuso orario selezionato dall'amministratore. Per modificare il fuso orario visualizzato, fare clic su **Finestra** > **Preferenze** > **IBM MQ Explorer** > **Managed File Transfer** e selezionare un fuso orario alternativo dall'elenco **Fuso orario:** . Fare clic su **OK**.

g) **Ripeti ogni** Se si è scelto di ripetere il trasferimento pianificato, l'intervallo specificato in cui si desidera ripetere il trasferimento, espresso come numero.

h) **Tipo di ripetizione** Se si è scelto di ripetere il trasferimento pianificato, il tipo di intervallo di ripetizione specificato per il trasferimento file. Il tipo può essere uno dei seguenti valori: `minuti`, `ore`, `giorni`, `settimane`, `mesi` o `anni`.

i) **Ripeti fino a** Se si è scelto di ripetere il trasferimento pianificato, i dettagli di quando si desidera arrestare il trasferimento file ripetuto. Ad esempio, una data e ora specificate o dopo un numero specificato di ricorrenze.

Risultati

Per aggiornare quanto visualizzato nella finestra **Trasferimenti in sospeso** , fare clic sul pulsante Aggiorna  sulla barra degli strumenti della vista Contenuto.

Per annullare un trasferimento file in sospeso, fare clic con il pulsante destro del mouse sul trasferimento particolare e fare clic su **Annulla**. L'annullamento di un trasferimento elimina completamente la richiesta di trasferimento file.

Attivazione di un trasferimento file

È possibile impostare determinate condizioni di trigger su un trasferimento file che devono essere true prima che tale trasferimento possa essere effettuato. Se le condizioni di attivazione non sono true, il trasferimento file non viene eseguito e un messaggio di log viene facoltativamente inoltrato per registrare il fatto che il trasferimento non si è verificato. La richiesta di trasferimento file viene quindi eliminata. Ad esempio, è possibile impostare un trasferimento file che si verifica solo se un file denominato sul sistema in cui si trova l'agent di origine è superiore a una dimensione specificata o se un particolare file denominato esiste sul sistema in cui si trova l'agent di origine. È possibile impostare un trasferimento file attivato da IBM MQ Explorer o dalla riga comandi.

Informazioni su questa attività

È possibile monitorare continuamente una risorsa per soddisfare una condizione di trigger. Per ulteriori informazioni sul monitoraggio delle risorse, consultare: “Monitoraggio delle risorse MFT” a pagina 229.

Esistono tre diverse condizioni di attivazione che è possibile impostare. Le condizioni sono le seguenti:

- Se un particolare file esiste sullo stesso sistema dell'agent di origine
- Se un particolare file non esiste sullo stesso sistema dell'agent di origine
- Se un particolare file supera una determinata dimensione sul sistema in cui si trova l'agent di origine (la dimensione può essere espressa in byte, KB, MB o GB). Queste unità di misura utilizzano la convenzione 2^{10} , ad esempio 1 KB equivale a 1024 byte e 1 MB equivale a 1024 KB.

I tipi di attivazione nell'elenco precedente possono essere combinati in due modi:

- Per una singola condizione, è possibile specificare più di un file sul sistema in cui si trova l'agente di origine. Ciò attiva il trasferimento se uno qualsiasi dei file specificati soddisfa la condizione (operatore booleano OR).
- È possibile specificare più condizioni. Ciò attiva il trasferimento solo se tutte le condizioni sono soddisfatte (operatore booleano AND).

Puoi anche combinare un trasferimento attivato con un trasferimento pianificato. Per ulteriori informazioni, consultare Creazione di un trasferimento file pianificato . In questo caso, le condizioni di trigger vengono valutate nel momento in cui la pianificazione deve essere avviata o per una pianificazione ripetuta ogni volta che la pianificazione deve essere avviata.

I trasferimenti attivati non sono supportati sugli agent bridge di protocollo.

Per creare un trasferimento file attivato utilizzando la riga comandi, utilizzare il parametro **-tr** sul comando fteCreateTransfer .

Per creare un trasferimento file pianificato utilizzando la procedura **Crea nuovo trasferimento file gestito** in IBM MQ Explorer, effettuare le seguenti operazioni:

Procedura

1. Nella vista Navigator , fare clic su **Managed File Transfer**. **Managed File Transfer Central** viene visualizzato nella vista Contenuto.
2. Tutti i gestori code di coordinamento vengono visualizzati nella vista Navigator . Espandere il nome del gestore code di coordinamento utilizzato per il trasferimento pianificato. Se si desidera modificare il gestore code di coordinamento a cui si è connessi, fare clic con il tasto destro del mouse sul nome del gestore code di coordinamento che si desidera utilizzare nella vista Navigator e fare clic su **Connetti**.
3. Avviare la procedura guidata **Crea nuovo trasferimento file gestito** utilizzando uno dei metodi seguenti:
   a) Fare clic con il tasto destro del mouse sul nome di uno dei seguenti nodi nella vista Navigator : il gestore code di coordinamento pertinente, **Modelli di trasferimento**, **Log di trasferimento**o **Trasferimenti in sospeso**. Quindi, fare clic su **Nuovo trasferimento** per aprire la procedura guidata.
   b) Fare clic su **File** > **Nuovo** > **Altro** > **Managed File Transfer Wizard** > **Nuova procedura guidata di trasferimento**
4. Seguire le istruzioni sui pannelli della procedura guidata. Assicurarsi di selezionare la casella di spunta **Abilita trasferimento attivato** nella scheda **Trigger** e completare i campi in tale scheda per impostare il trigger. Per ogni pannello viene fornita una guida sensibile al contesto. Per accedere alla guida sensibile al contesto su Windows, premere F1. Su Linux, premere **Ctrl+F1** o `Shift+F1`.

Monitoraggio dei trasferimenti file in corso

È possibile monitorare un trasferimento file in corso utilizzando la scheda **Managed File Transfer - Avanzamento trasferimento corrente** in IBM MQ Explorer. Questo trasferimento file può essere avviato da IBM MQ Explorer o dalla riga comandi. La scheda visualizza anche l'avanzamento dei trasferimenti pianificati nel momento in cui iniziano i trasferimenti pianificati.

Informazioni su questa attività

Se si desidera utilizzare IBM MQ Explorer per monitorare i trasferimenti associati a un gestore code di coordinamento su un sistema remoto, seguire le istruzioni nell'argomento “Configurazione di IBM MQ Explorer per monitorare un gestore code di coordinamento remoto” a pagina 227 .

Le informazioni sul trasferimento file precedenti non vengono conservate dopo l'arresto e il riavvio di IBM MQ Explorer. Al riavvio, le informazioni sui trasferimenti precedenti vengono cancellate dalla scheda **Avanzamento trasferimento corrente** . È possibile cancellare i trasferimenti completati utilizzando l'icona **Rimuovi trasferimenti completati** in qualsiasi momento quando IBM MQ Explorer è aperto.

Procedura

Dopo aver avviato un nuovo trasferimento file utilizzando IBM MQ Explorer o la riga comandi, è possibile monitorare l'avanzamento del trasferimento nella scheda **Avanzamento trasferimento corrente** . Per ogni trasferimento in corso vengono visualizzate le seguenti informazioni:

a) **Origine**. Il nome dell'agente utilizzato per trasferire il file dal sistema di origine.

b) **Destinazione:**. Il nome dell'agente utilizzato per ricevere il file sul sistema di destinazione.

c) **File corrente**. Il nome del file attualmente trasferito. La parte del singolo file che è già stato trasferito viene visualizzata in B, KiB, MiB. GiBo TiB insieme alla dimensione totale del file tra parentesi. L'unità di misura visualizzata dipende dalla dimensione del file.

   B è byte al secondo. KiB/s è kibibyte al secondo, dove 1 kibibyte è uguale a 1024 byte. MiB/s è mebibyte al secondo, dove 1 mebibyte equivale a 1 048 576 byte. GiB/s è gibibyte al secondo dove 1 gibibyte è uguale a 1 073 741 824 byte. TiB/s è tebibyte al secondo dove 1 tebibyte equivale a 1 099 511 627 776 byte.

d) **Numero file**. Se si sta trasferendo più di un file, questo numero rappresenta la distanza del gruppo totale di file da trasferire.

e) **Avanzamento**. La barra di avanzamento mostra il completamento del trasferimento file corrente come percentuale.

f) **Tasso**. La frequenza con cui il file viene trasferito in KiB/s (kibibyte al secondo, dove 1 kibibyte equivale a 1024 byte).

g) **Avviato (fuso orario selezionato)**. L'ora in cui è stato avviato il trasferimento file, presentata nel fuso orario selezionato dell'amministratore. Per modificare il fuso orario visualizzato, fare clic su **Finestra** > **Preferenze** > **IBM MQ Explorer** > **Managed File Transfer** e selezionare un fuso orario alternativo dall'elenco **Fuso orario:** . Fare clic su **OK**.

   Se il trasferimento entra in uno stato di ripristino durante il trasferimento del file, l'ora di inizio viene aggiornata in modo da riflettere l'ora in cui è stato ripreso il trasferimento del file.

Risultati

Questa scheda aggiorna regolarmente le proprie informazioni automaticamente, ma per forzare una vista aggiornata di ciò che viene visualizzato nella scheda **Avanzamento trasferimento corrente** , fare clic su **Aggiorna** nella barra degli strumenti della vista Contenuto.

Per eliminare i trasferimenti file dalla scheda **Avanzamento trasferimento corrente** , fare clic su **Rimuovi trasferimenti completati** nella barra degli strumenti della vista Contenuto. Facendo clic su questo pulsante si rimuovono i dettagli del trasferimento file solo dalla scheda; non si arresta o si annulla un trasferimento corrente o pianificato.

Se si desidera tornare alla scheda **Avanzamento trasferimento corrente** dopo averla chiusa, è possibile visualizzare la scheda facendo clic su **Finestra** > **Mostra vista** > **Altro** > **Altro** > **Trasferimento file gestito - Avanzamento trasferimento corrente**. Fare clic su **OK**.

Operazioni successive

Inoltre, è possibile sviluppare applicazioni per il monitoraggio del trasferimento file personalizzato. Ciò può essere realizzato creando una sottoscrizione all'argomento di gestione Managed File Transfer appropriato (in modo programmatico o amministrativo) e l'applicazione di monitoraggio può quindi ricevere le pubblicazioni dell'attività di trasferimento file Managed File Transfer sull'argomento. Per ulteriori informazioni sull'argomento della sottoscrizione e sul formato del messaggio di pubblicazione, consultare Esempi di messaggi di avanzamento del trasferimento file.

Attività correlate

“Configurazione di IBM MQ Explorer per monitorare un gestore code di coordinamento remoto” a pagina 227
Utilizzare IBM MQ Explorer per monitorare i trasferimenti di file associati a un gestore code di coordinamento in esecuzione su un sistema remoto. È necessario un sistema in grado di eseguire IBM MQ Explorer. Il componente IBM MQ Explorer deve essere installato per essere in grado di connettersi al gestore code di coordinamento remoto.

“Visualizzazione dello stato dei trasferimenti file nel log di trasferimento” a pagina 227
È possibile visualizzare i dettagli dei trasferimenti file utilizzando il **Log trasferimenti** in IBM MQ Explorer. Questi possono essere trasferimenti avviati dalla riga comandi o da IBM MQ Explorer. È anche possibile personalizzare quanto visualizzato nel **Log di trasferimento**.

Configurazione di IBM MQ Explorer per monitorare un gestore code di coordinamento remoto

Utilizzare IBM MQ Explorer per monitorare i trasferimenti di file associati a un gestore code di coordinamento in esecuzione su un sistema remoto. È necessario un sistema in grado di eseguire IBM MQ Explorer. Il componente IBM MQ Explorer deve essere installato per essere in grado di connettersi al gestore code di coordinamento remoto.

Informazioni su questa attività

Presupposti: autorizzazione a connettersi al gestore code di coordinamento remoto configurando il gestore code per consentire le connessioni remote.

Per ulteriori informazioni su come configurare questa configurazione, fare riferimento a Connessione a un gestore code in modalità client con autenticazione di canale e Gestione delle autorizzazioni per le risorse specifiche di MFT.

Per monitorare i gestori code e i trasferimenti file tra agent su un sistema su cui non è in esecuzione Windows o Linux, configurare IBM MQ Explorer per la connessione al sistema remoto utilizzando la seguente procedura:

Procedura

1. Avviare il IBM MQ Explorerlocale.
2. Quando IBM MQ Explorer viene caricato, fare clic con il pulsante destro del mouse sulla cartella **Managed File Transfer** e selezionare **Nuova configurazione**.
3. Procedere con la procedura guidata, selezionando il gestore code Coordinamento e Comandi, quindi definire un nome per la configurazione.
4. Fare clic su **Fine** per completare la definizione.
5. Una volta terminata la definizione, fare clic con il tasto destro del mouse sulla definizione e selezionare **Connetti**.

Risultati

Ora avviare IBM MQ Explorer e utilizzarlo per monitorare l'attività di trasferimento per la rete Managed File Transfer associata con il gestore code di coordinamento.

Attività correlate

“Monitoraggio dei trasferimenti file in corso” a pagina 225
È possibile monitorare un trasferimento file in corso utilizzando la scheda **Managed File Transfer - Avanzamento trasferimento corrente** in IBM MQ Explorer. Questo trasferimento file può essere avviato da IBM MQ Explorer o dalla riga comandi. La scheda visualizza anche l'avanzamento dei trasferimenti pianificati nel momento in cui iniziano i trasferimenti pianificati.

“Visualizzazione dello stato dei trasferimenti file nel log di trasferimento” a pagina 227
È possibile visualizzare i dettagli dei trasferimenti file utilizzando il **Log trasferimenti** in IBM MQ Explorer. Questi possono essere trasferimenti avviati dalla riga comandi o da IBM MQ Explorer. È anche possibile personalizzare quanto visualizzato nel **Log di trasferimento**.

Visualizzazione dello stato dei trasferimenti file nel log di trasferimento

È possibile visualizzare i dettagli dei trasferimenti file utilizzando il **Log trasferimenti** in IBM MQ Explorer. Questi possono essere trasferimenti avviati dalla riga comandi o da IBM MQ Explorer. È anche possibile personalizzare quanto visualizzato nel **Log di trasferimento**.

Procedura

1. Espandere **Managed File Transfer** nella vista Navigator ed espandere il nome del gestore code di coordinamento per cui si desidera visualizzare il log di trasferimento.

2. Fare clic su **Log trasferimento** nella vista Navigator . Il **Log di trasferimento** viene visualizzato nella vista Contenuto.
3. La finestra **Log trasferimenti** visualizza i seguenti dettagli sui trasferimenti file:
   a) **Origine** Il nome dell'agent sul sistema in cui si trova il file di origine.
   b) **Destinazione** il nome dell'agent sul sistema a cui si desidera trasferire il file.
   c) **Stato di completamento** Lo stato del trasferimento file. Lo stato può avere uno dei seguenti valori: "Avviato", "In corso", "Riuscito", "Parzialmente riuscito", "Annullato" o "Non riuscito".
   d) **Proprietario** L'ID utente sull'host che ha inoltrato la richiesta di trasferimento.
   e) **Avviato (fuso orario selezionato)** La data/ora in cui la richiesta di trasferimento file è stata accettata dall'agent Managed File Transfer , presentata nel fuso orario selezionato dell'amministratore. Per modificare il fuso orario visualizzato, fare clic su **Finestra** > **Preferenze** > **IBM MQ Explorer** > **Managed File Transfer** e selezionare un fuso orario alternativo dall'elenco **Fuso orario:** . Fare clic su **OK**.
   f) **Stato registrato (fuso orario selezionato)** (Questa colonna non è visualizzata per default. È possibile scegliere di visualizzare la colonna utilizzando la finestra **Configura colonne log di trasferimento**  . La data e l'ora in cui è stato registrato lo stato di completamento, nel fuso orario selezionato dall'amministratore.
   g) **Nome lavoro** Un identificativo specificato da un utente utilizzando il parametro **`-jn`** di **`fteCreateTransfer`** o in uno script Ant
   h) **ID trasferimento** L'identificativo univoco per il trasferimento file.
   i) **Connessione: diretta** Sono elencati i dettagli su **Numero processo**, **Nome processo**, **Nodo primario**, **Nodo secondario**, **Tipo di origine** e **Tipo di destinazione** .

Risultati

Nota: Il formato interno del log di trasferimento è stato modificato in IBM MQ 8.0.0 Fix Pack 1 per APAR IC99545. Di conseguenza, se un IBM MQ Explorer viene aggiornato a V8.0.0.1 o versioni successive e quindi ripristinato a V8.0.0.0, non viene visualizzato alcun XML di verifica per i trasferimenti che hanno avuto luogo mentre IBM MQ Explorer si trovava in V8.0.0.1. Il pannello XML nella finestra **Proprietà** per questi trasferimenti conterrà una finestra di testo vuota.

Per visualizzare ulteriori dettagli su un trasferimento completato, espandere il trasferimento a cui si è interessati facendo clic sul segno più (+). È quindi possibile visualizzare tutti i nomi file di origine e di destinazione inclusi in tale trasferimento. Tuttavia, se il trasferimento è attualmente in corso ed è composto da molti file, è possibile visualizzare solo i file che sono già stati trasferiti fino ad ora.

Per aggiornare quanto visualizzato nel **Log di trasferimento**, fare clic sul pulsante **Aggiorna**  nella barra degli strumenti della vista Contenuto. Le informazioni sul trasferimento file nel log di trasferimento rimangono nel log dopo l'arresto e il riavvio di IBM MQ Explorer. Se si desidera eliminare tutti i trasferimenti file completati dal log, fare clic su **Rimuovi trasferimenti completati** nella barra degli strumenti della vista Contenuto.

Per eliminare un singolo trasferimento file completato dal log, fare clic con il tasto destro del mouse sul trasferimento e fare clic su **Elimina**. Se si elimina un trasferimento, non si arresta o si annulla un trasferimento in corso o che è stato pianificato; si stanno eliminando solo i dati cronologici memorizzati.

Per copiare l'identificativo univoco di un trasferimento negli appunti, fare clic con il pulsante destro del mouse su tale trasferimento e fare clic su **Copia ID**.

I metadati e l'XML di controllo completo per il trasferimento sono disponibili dal menu a comparsa, nell'azione **Proprietà** .

Attività correlate

“Monitoraggio dei trasferimenti file in corso” a pagina 225
È possibile monitorare un trasferimento file in corso utilizzando la scheda **Managed File Transfer - Avanzamento trasferimento corrente** in IBM MQ Explorer. Questo trasferimento file può essere avviato

da IBM MQ Explorer o dalla riga comandi. La scheda visualizza anche l'avanzamento dei trasferimenti pianificati nel momento in cui iniziano i trasferimenti pianificati.

"Configurazione del log di trasferimento" a pagina 229
È possibile configurare quali informazioni vengono visualizzate e come vengono visualizzate nel **Log di trasferimento** in IBM MQ Explorer.

"Impostazione di una scadenza per il recupero di trasferimenti in stallo" a pagina 331
È possibile impostare un timeout di ripristino del trasferimento per i trasferimenti di file in stallo che si applica a tutti i trasferimenti per un agent di origine. È anche possibile impostare un timeout di ripristino trasferimento per un singolo trasferimento. Se si imposta un periodo di tempo specifico, in secondi, durante il quale un agent di origine tenta di ripristinare un trasferimento file in stallo e il trasferimento non riesce quando l'agent raggiunge il timeout, il trasferimento non riesce.

Configurazione del log di trasferimento

È possibile configurare quali informazioni vengono visualizzate e come vengono visualizzate nel **Log di trasferimento** in IBM MQ Explorer.

Informazioni su questa attività

Per riorganizzare l'ordine delle colonne nel **Log di trasferimento**, fare clic sul titolo della colonna che si desidera spostare e trascinare la colonna nella nuova posizione. Il nuovo ordine delle colonne viene conservato solo fino al successivo arresto e riavvio di IBM MQ Explorer.

Per filtrare le voci nel **Log trasferimenti**, immettere una stringa nel campo **Filtra le voci di log visualizzate** . Per ripristinare tutte le voci nel log, eliminare la stringa immessa dal campo. È possibile utilizzare qualsiasi espressione regolare Java valida in questo campo. Per ulteriori informazioni, consultare Espressioni regolari utilizzate da MFT.

Per personalizzare le colonne visualizzate nel log di trasferimento, utilizzare **Configura colonne log di trasferimento**  . Utilizzare la seguente procedura per avviare e utilizzare la finestra di dialogo **Configura colonne log di trasferimento** .

Procedura

1. Assicurarsi di avere il **Log di trasferimento** aperto nella vista Contenuto. Fare clic su **Configura colonne log di trasferimenti**  sulla barra degli strumenti della vista Contenuto. Viene visualizzata la finestra **Configura colonne log di trasferimento** .
2. Per personalizzare la vista del **Log di trasferimento**, selezionare o deselezionare le singole caselle di spunta per le colonne che si desidera visualizzare o nascondere. È possibile fare clic su **Seleziona tutto**, quindi su **OK** per selezionare tutte le caselle di spunta oppure su **Deseleziona tutto**, quindi su **OK** per deselezionare tutte le caselle di spunta.

Attività correlate

"Monitoraggio dei trasferimenti file in corso" a pagina 225
È possibile monitorare un trasferimento file in corso utilizzando la scheda **Managed File Transfer - Avanzamento trasferimento corrente** in IBM MQ Explorer. Questo trasferimento file può essere avviato da IBM MQ Explorer o dalla riga comandi. La scheda visualizza anche l'avanzamento dei trasferimenti pianificati nel momento in cui iniziano i trasferimenti pianificati.

"Visualizzazione dello stato dei trasferimenti file nel log di trasferimento" a pagina 227
È possibile visualizzare i dettagli dei trasferimenti file utilizzando il **Log trasferimenti** in IBM MQ Explorer. Questi possono essere trasferimenti avviati dalla riga comandi o da IBM MQ Explorer. È anche possibile personalizzare quanto visualizzato nel **Log di trasferimento**.

Monitoraggio delle risorse MFT

È possibile monitorare le risorse Managed File Transfer ; ad esempio, una coda o una directory. Quando viene soddisfatta una condizione su questa risorsa, il monitoraggio risorse avvia un'attività, ad

esempio un trasferimento file. È possibile creare un monitoraggio delle risorse utilizzando il comando **fteCreateMonitor** o la vista **Monitor** nel plug-in Managed File Transfer per IBM MQ Explorer.

Informazioni su questa attività

Il monitoraggio delle risorse Managed File Transfer usa la terminologia seguente:

Controllo risorse
Un monitoraggio risorse è un processo che esegue il polling di una risorsa (come un indirizzario o una coda) a un intervallo regolare predefinito per verificare se il contenuto della risorsa è stato modificato. Se lo sono, il contenuto viene confrontato con la serie di condizioni per questo monitor. Se esiste una corrispondenza, l'attività per questo monitoraggio viene avviata.

risorsa
La risorsa di sistema che il monitoraggio risorse esamina ad ogni intervallo di polling da confrontare con le condizioni di trigger. Le code, le directory o le strutture di directory nidificate possono essere la risorsa monitorata.

Condizione e condizione trigger
Una condizione è un'espressione che viene valutata (generalmente rispetto al contenuto della risorsa monitorata). Se l'espressione viene valutata true, la condizione contribuisce alla condizione di trigger generale.

La condizione trigger è la condizione generale, che viene soddisfatta quando tutte le condizioni vengono soddisfatte. Quando la condizione di trigger viene soddisfatta, l'attività può procedere.

Attività
Un'attività è l'operazione che viene avviata quando viene soddisfatta la condizione di trigger o la serie di condizioni. Le attività supportate sono il trasferimento file e la chiamata del comando.

File trigger
Un file trigger è un file collocato in una directory monitorata per indicare che un'attività (in genere un trasferimento) può iniziare. Ad esempio, potrebbe indicare che tutti i file da elaborare sono arrivati in una posizione nota e possono essere trasferiti o altrimenti gestiti. Il nome del file trigger può essere utilizzato per specificare i file da trasferire utilizzando la sostituzione variabile. Per ulteriori informazioni, consultare “Personalizzazione delle attività di monitoraggio delle risorse MFT con la sostituzione della variabile” a pagina 242.

Il file trigger è noto anche come file pronto o file go. Tuttavia, in questa documentazione viene generalmente indicato come file trigger.

Il monitoraggio delle risorse non è supportato sugli agent bridge di protocollo o sugli agent bridge Connect:Direct .

Concetti correlati
Guida per configurare un monitoraggio risorse MFT per evitare il sovraccarico di un agente
Riferimenti correlati
fteCreateMonitor: crea un monitoraggio risorse MFT
fteListMonitoraggi: elenco MFT monitoraggi risorse
Controllo fteDelete: elimina un controllo risorse MFT
Formati del messaggio di richieste di controllo MFT

Concetti di monitoraggio delle risorse MFT

Una panoramica dei concetti chiave della funzione di monitoraggio delle risorse Managed File Transfer .

Monitoraggi risorse

Si crea un controllo risorse utilizzando il comando **fteCreateMonitor** , che crea e avvia un nuovo controllo risorse dalla riga comandi. Il monitoraggio delle risorse è associato a un agent Managed File Transfer ed è attivo solo quando tale agent è in esecuzione. Quando l'agent di monitoraggio viene arrestato, lo stesso vale per il monitoraggio delle risorse. Se l'agent è già in esecuzione quando viene

creato il monitoraggio risorse, il monitoraggio risorse viene avviato immediatamente. L'agente di controllo deve essere anche l'agente di origine dell'attività avviata dal controllo risorse.

I nomi del monitoraggio risorse devono essere univoci all'interno del relativo agent. Il nome del controllo risorse deve avere una lunghezza minima di un carattere e non deve contenere caratteri asterisco (*), percentuale (%) o punto interrogativo (?). Il caso in cui viene fornito un nome di monitoraggio risorse viene ignorato e il nome di monitoraggio risorse viene convertito in maiuscolo. Se si tenta di creare un monitoraggio risorse con un nome già presente, la richiesta viene ignorata e il tentativo viene registrato nell'argomento del log del monitoraggio risorse.

Nota: Non è possibile creare un monitoraggio risorse con una definizione attività che contenga trasferimenti pianificati.

Prima di IBM MQ 9.3.0, l'unico modo per arrestare un monitoraggio risorse consiste nell'arrestare l'agent che sta eseguendo l'operazione di monitoraggio. Per riavviare un monitoraggio risorse, è necessario riavviare l'agente. Da IBM MQ 9.3.0, è possibile avviare e arrestare i monitoraggi delle risorse senza dover arrestare o riavviare un agent. Per ulteriori informazioni, consultare “Avvio e arresto dei monitoraggi delle risorse” a pagina 233.

Non esiste alcuna limitazione sul numero di monitoraggi delle risorse che possono essere creati su un agent e tutti eseguiti con la stessa priorità. Considerare le implicazioni della sovrapposizione delle risorse monitorate, le condizioni di trigger in conflitto e la frequenza con cui viene eseguito il polling delle risorse.

La sovrapposizione dei monitoraggi delle risorse può causare:

- Possibile conflitto sull'ubicazione/elementi di origine.
- Possibili richieste di trasferimento duplicate per gli stessi elementi di origine.
- Errori o errori imprevisti per i trasferimenti dovuti a conflitti di elementi di origine.

Se più monitoraggi eseguono la scansione della stessa ubicazione e possono essere attivati sugli stessi elementi, è possibile che si verifichi il problema di due monitoraggi differenti che inoltrano le richieste di trasferimento gestito per lo stesso elemento.

I monitoraggi risorse esaminano il contenuto delle risorse dopo ogni periodo di intervallo di polling. Il contenuto della risorsa viene confrontato con le condizioni di trigger e se tali condizioni vengono soddisfatte, viene richiamata l'attività associata al monitoraggio risorse.

L'attività viene avviata in maniera asincrona. Se esiste una corrispondenza di condizioni e l'attività viene avviata, il monitoraggio delle risorse continua a eseguire il polling per ulteriori modifiche al contenuto della risorsa. Quindi, ad esempio, se si verifica una corrispondenza perché un file denominato `reports.go` è arrivato in una directory monitorata, l'attività verrà avviata una sola volta. Al successivo intervallo di polling, anche se il file esiste ancora, l'attività non viene riavviata. Tuttavia, se il file viene eliminato e quindi riposizionato nella directory o se il file viene aggiornato (in modo che l'attributo della data dell'ultima modifica venga modificato), il successivo controllo della condizione di trigger fa sì che l'attività venga richiamata di nuovo.

Prima di IBM MQ 9.1.5, se un monitoraggio delle risorse esegue un polling che impiega più tempo dell'intervallo di polling, ciò significa che il successivo polling inizia non appena quello corrente termina senza interlinea, il che potrebbe avere un effetto sulla velocità con cui i monitoraggi delle risorse inoltrano il lavoro a un agente. Ciò potrebbe causare problemi di prestazioni se gli elementi rilevati durante il primo polling sono ancora presenti quando si verifica il secondo.

Il monitoraggio risorse utilizza il Servizio ScheduledExecutore avvia il successivo polling solo dopo il completamento del polling precedente più il tempo di intervallo di polling configurato. Ciò significa che ci sarà sempre un divario tra gli intervalli di polling, piuttosto che avere un altro polling che inizia subito dopo il polling precedente se il tempo di polling era più lungo dell'intervallo di polling.

Se il trasferimento di un file non è riuscito, è possibile cancellare la cronologia del monitoraggio delle risorse, che consente l'inoltro di un'altra richiesta di trasferimento senza la necessità di eliminare il file e inserirlo nuovamente nell'indirizzario o aggiornare il file per modificare l'attributo della data dell'ultima modifica. La cancellazione della cronologia è utile, ad esempio, in situazioni in cui è necessario trasferire il file ma non è possibile modificarlo. Per ulteriori informazioni, consultare “Cancellazione della cronologia del controllo risorse” a pagina 260.

Risorse

I monitoraggi delle risorse in Managed File Transfer possono eseguire il polling del contenuto dei due seguenti tipi di risorse:

Directory o strutture di directory nidificate
Uno scenario comune consiste nel monitorare una directory per la presenza di un file trigger. Un'applicazione esterna potrebbe elaborare più file e collocarli in una directory di origine nota. Quando l'applicazione ha completato l'elaborazione, indica che i file sono pronti per essere trasferiti o su cui si agisce in altro modo, posizionando un file trigger in un'ubicazione monitorata. Il file trigger può essere rilevato da un controllo risorse Managed File Transfer e il trasferimento di tali file dalla directory di origine a un altro Managed File Transfer Agent viene avviato.

Per impostazione predefinita, la directory specificata viene monitorata. Per esaminare anche le sottodirectory, impostare il livello di ricorsione nel comando **`fteCreateTransfer`** .

Di seguito sono riportati due esempi di monitoraggio di una directory:

- Monitorare un file trigger (ad esempio, `trigger.file`) e quindi trasferire un carattere jolly (ad esempio, `*.zip`).
- Monitorare `*.zip` e quindi trasferire `${FilePath}` (ad esempio, il file che ha attivato il trasferimento). Per ulteriori informazioni sulla sostituzione delle variabili, consultare “Personalizzazione delle attività di monitoraggio delle risorse MFT con la sostituzione della variabile” a pagina 242.

Nota: Non creare un monitoraggio che monitori `*.zip`e quindi trasferisca `*.zip`. Il monitoraggio tenta di avviare un trasferimento di `*.zip` per ogni file .zip sul sistema. Vale a dire, il monitor genera * numero di trasferimenti per `*.zip`.

Per un esempio di creazione di un monitoraggio risorse per monitorare una directory, consultare “Monitoraggio di una directory e utilizzo della sostituzione della variabile” a pagina 239.

Code IBM MQ
Un esempio di monitoraggio di una coda è che un'applicazione esterna potrebbe generare messaggi e posizionarli su una coda nota con lo stesso ID gruppo. Quando l'applicazione ha completato l'inserimento dei messaggi nella coda, indica che il gruppo è completo. Il gruppo completo di messaggi può essere rilevato da un monitoraggio risorse Managed File Transfer e viene avviato il trasferimento del gruppo di messaggi dalla coda di origine a un file. Per un esempio di creazione di un controllo risorse per monitorare una coda, consultare “Esempio: configurazione di una risorsa MFT” a pagina 241.

Nota: È possibile specificare un solo monitor per coda. Se si specifica più di un monitor per eseguire il polling di una coda IBM MQ , si verifica un comportamento imprevedibile.

Il monitoraggio dei dataset non è supportato.

Condizioni e condizioni di attivazione

La condizione viene soddisfatta quando la risorsa contiene un valore che corrisponde a qualche altra stringa o modello. Le condizioni possono essere una delle seguenti:

- Corrispondenza sul nome file (modello)
- Nessuna corrispondenza sul nome file (modello)
- Dimensione file
- Corrisponde se la dimensione del file rimane la stessa per un certo numero di polling

La corrispondenza del nome file può essere espressa come:

- Corrispondenza stringa esatta
- Corrispondenza di caratteri jolly semplici come descritto in Utilizzo di caratteri jolly con MFT
- corrispondenza espressione regolare

I nomi file possono anche essere esclusi dalla corrispondenza dei nomi file utilizzando un carattere jolly o un'espressione regolare Java che identifica i nomi file che non corrispondono mai.

Quando viene rilevato un file corrispondente, viene conservata la data / ora dell'ultima modifica. Se i polling successivi rilevano che il file è stato modificato, la condizione di trigger viene soddisfatta di nuovo e l'attività viene avviata. Se la condizione è rilevare quando un file non esiste, se nessun file nella directory monitorata corrisponde al modello del nome file, l'attività viene avviata. Se un file viene aggiunto alla directory che non corrisponde al pattern del nome file, l'attività viene avviata solo se il file viene eliminato.

Attività

Managed File Transfer supporta i seguenti due tipi di attività che è possibile configurare per essere avviati dai monitoraggi delle risorse:

Attività di trasferimento file
Le attività di trasferimento file sono definite nello stesso modo di qualsiasi altro trasferimento file. Un modo utile per generare l'attività XML richiesta da un controllo consiste nell'eseguire il comando fteCreateTransfer con il parametro **-gt** . Questo comando genera una definizione di attività come documento XML, inclusa la specifica di trasferimento. Si passa quindi il nome del documento XML dell'attività come valore per il parametro **-mt** nel comando fteCreateMonitor . Quando **fteCreateMonitor** viene eseguito, legge il documento XML di attività. Dopo l'esecuzione di **fteCreateMonitor** , tutte le modifiche apportate al file XML dell'attività non vengono utilizzate dal monitoraggio.

Quando si utilizza un'attività di trasferimento file, è possibile selezionare quante condizioni di trigger vengono raggruppate in un'attività. Il valore predefinito è per una condizione trigger per avviare un'attività. È possibile eseguire il comando fteCreateMonitor con l'opzione **-bs** per selezionare il numero di condizioni di trigger raggruppate in un'unica attività.

Attività di comando
Le attività di comando possono eseguire script Ant , richiamare programmi eseguibili o eseguire lavori JCL. Per ulteriori informazioni, consultare “Configurazione delle attività di monitoraggio MFT per avviare comandi e script” a pagina 235.

File trigger

È possibile utilizzare il contenuto di un file trigger in un controllo risorse per definire una serie di file da trasferire in una richiesta di trasferimento singola. Ogni volta che viene rilevato un file trigger corrispondente, il suo contenuto viene analizzato per i percorsi file di origine e facoltativamente per i percorsi file di destinazione. Questi percorsi file vengono quindi utilizzati per definire gli elementi file nel file XML di trasferimento attività specificato, inoltrato come singola richiesta di trasferimento all'agente. La definizione del monitoraggio risorse determina se il contenuto del trigger è abilitato.

Il formato di ciascun file trigger è un singolo percorso file da trasferire su ciascuna riga di testo. Il formato predefinito per la riga è un singolo percorso file di origine o un percorso file di origine e di destinazione separati da una virgola.

Per ulteriori informazioni ed esempi, consultare “Utilizzo di un file trigger” a pagina 251.

Avvio e arresto dei monitoraggi delle risorse

Prima di IBM MQ 9.3.0, l'unico modo per arrestare un monitoraggio risorse consiste nell'arrestare l'agent che sta eseguendo l'operazione di monitoraggio. Per riavviare un monitoraggio risorse, è necessario riavviare l'agente. Per ulteriori informazioni, consultare “Avvio di un agent MFT” a pagina 211 e “Arresto di un agent MFT” a pagina 217.

Da IBM MQ 9.3.0, è possibile avviare e arrestare i monitoraggi risorse senza dover arrestare o riavviare un agent utilizzando i comandi **fteStartMonitor** e **fteStopMonitor** . Ciò è utile, ad esempio, nelle situazioni seguenti:

- Se un agent dispone di più monitoraggi delle risorse e solo alcuni di essi hanno rilevato errori, ma i restanti monitoraggi delle risorse stanno ancora funzionando correttamente, si desidera solo riavviare i monitoraggi delle risorse non riusciti.
- Se si desidera arrestare un controllo risorse per eseguire del lavoro di manutenzione o se il controllo risorse non è richiesto per un determinato periodo di tempo e non si desidera che venga eseguito inutilmente, consumando risorse di sistema preziose.

Per ulteriori informazioni, consultare "Avvio di un monitoraggio risorse MFT" a pagina 256 e "Arresto di un monitoraggio risorse MFT" a pagina 257.

Tabella 9. Comportamento di un monitoraggio risorse a seconda del comando eseguito

Comando	Comportamento del monitoraggio risorse
fteStartMonitor	Se l'agent è in esecuzione, il monitoraggio delle risorse viene avviato se è attualmente arrestato.
fteStopMonitor	Se l'agent è in esecuzione, il monitoraggio delle risorse viene arrestato se è attualmente avviato.
fteStartAgent	Il monitoraggio delle risorse viene avviato come parte dell'avvio dell'agente, indipendentemente dalle chiamate precedenti a **fteStopMonitor**.
fteStopAgent	Tutti i monitoraggi delle risorse in esecuzione vengono arrestati.

Backup e ripristino dei monitoraggi risorse

È possibile eseguire il backup dei monitoraggi risorse già definiti in modo da poterli riutilizzare in futuro. Esistono diverse opzioni che è possibile utilizzare come segue:

- Utilizzare il comando **fteCreateMonitor** con il parametro **-ox** per esportare una configurazione del controllo risorse in un file XML e con il parametro **-ix** per ripristinare un controllo risorse importando la configurazione del controllo risorse da un file XML.
- Utilizzare il comando **fteListMonitors** con **-ox** per esportare la definizione per un singolo monitoraggio risorse in un file XML.
- Utilizzare il comando **fteListMonitors** con il **-od** per esportare più definizioni di monitoraggio risorse in una directory specificata. Ogni definizione di monitoraggio risorse viene salvata in un file XML separato. È anche possibile utilizzare l'opzione **-od** per esportare una singola definizione di monitoraggio risorse in una directory specificata.

Per ulteriori informazioni, consultare "Backup e ripristino dei monitoraggi delle risorse MFT" a pagina 258.

Registrazione monitoraggio risorse

Managed File Transfer include la registrazione del monitoraggio risorse. Per ulteriori informazioni, consultare "Registrazione dei monitoraggi delle risorse MFT" a pagina 254.

Concetti correlati

"Personalizzazione delle attività di monitoraggio delle risorse MFT con la sostituzione della variabile" a pagina 242
Quando le condizioni del trigger di un controllo risorse attivo vengono soddisfatte, viene richiamata l'attività definita. Oltre a richiamare l'attività di trasferimento o di comando con lo stesso agent di destinazione o con lo stesso nome file di destinazione ogni volta, è anche possibile modificare la definizione dell'attività in fase di runtime. A tale scopo, inserire i nomi delle variabili nel file XML di definizione attività. Quando il controllo determina che le condizioni di trigger sono soddisfatte e che la definizione dell'attività contiene nomi di variabili, sostituisce i nomi di variabile con valori di variabile e richiama l'attività.

Attività correlate

“Configurazione delle attività di monitoraggio MFT per avviare comandi e script” a pagina 235
I monitoraggi risorse non sono limitati all'esecuzione di trasferimenti file come attività associata. È anche possibile configurare il monitor per richiamare altri comandi dall'agent di monitoraggio, inclusi i programmi eseguibili, gli script Ant o i lavori JCL. Per richiamare i comandi, modificare l'XML di definizione dell'attività di monitoraggio in modo da includere uno o più elementi di comando con i parametri di chiamata del comando corrispondenti, ad esempio argomenti e proprietà.

“Esempio: configurazione di una risorsa MFT” a pagina 241
È possibile specificare una coda IBM MQ come risorsa che deve essere monitorata da un monitoraggio risorse utilizzando il parametro **-mq** con il comando **fteCreateMonitor** .

“Monitoraggio di una coda e utilizzo della sostituzione di variabili” a pagina 247
È possibile monitorare una coda e trasferire messaggi dalla coda monitorata a un file utilizzando il comando **fteCreateMonitor** . Il valore di qualsiasi proprietà del messaggio IBM MQ nel primo messaggio da leggere dalla coda monitorata può essere sostituito nella definizione XML dell'attività e utilizzato per definire il comportamento del trasferimento.

Riferimenti correlati

fteCreateMonitor: crea un monitoraggio risorse MFT

fteListMonitoraggi: elenco MFT monitoraggi risorse

Controllo fteDelete: elimina un controllo risorse MFT

Configurazione delle attività di monitoraggio MFT per avviare comandi e script

I monitoraggi risorse non sono limitati all'esecuzione di trasferimenti file come attività associata. È anche possibile configurare il monitor per richiamare altri comandi dall'agent di monitoraggio, inclusi i programmi eseguibili, gli script Ant o i lavori JCL. Per richiamare i comandi, modificare l'XML di definizione dell'attività di monitoraggio in modo da includere uno o più elementi di comando con i parametri di chiamata del comando corrispondenti, ad esempio argomenti e proprietà.

Informazioni su questa attività

Il percorso file del programma eseguibile, dello script Ant o del lavoro JCL che si desidera venga richiamato dall'agent di monitoraggio deve essere incluso nel `commandPath` dell'agent di monitoraggio. Per informazioni sulla proprietà del percorso del comando, consultare commandPath MFT property.

È possibile creare il documento XML di definizione attività in uno dei seguenti modi:

- Creare manualmente il documento XML di definizione attività in base allo schema `FileTransfer.xsd` .
- Utilizzare un documento XML generato come base per la propria definizione di attività.

Se si desidera un'attività di trasferimento o un'attività di comando, la definizione dell'attività deve iniziare con un elemento root `<request>` . L'elemento child di `<request>` deve essere `<managedTransfer>` o `<managedCall>`. Generalmente, è possibile scegliere `<managedCall>` quando è presente un singolo comando o script da eseguire e `<managedTransfer>` se si desidera che l'attività includa un trasferimento file e, facoltativamente, fino a quattro chiamate di comando.

Procedura

- Per creare manualmente il documento XML di definizione attività in base allo schema `FileTransfer.xsd` , consultare “Creazione manuale di un XML di definizione attività in base allo schema” a pagina 236.
- Per creare una definizione di attività modificando un documento generato, modificare il documento XML generato dal parametro **fteCreateTransfer -gt** . Per ulteriori informazioni, consultare “Creazione di un documento di definizione attività modificando un documento generato” a pagina 238.

Creazione manuale di un XML di definizione attività in base allo schema

È possibile creare manualmente un file XML di definizione attività in base allo schema `FileTransfer.xsd`.

Informazioni su questa attività

Lo schema `FileTransfer.xsd` è disponibile in *MQ_INSTALLATION_PATH*`/mqft/samples/schema`. Per ulteriori informazioni su questo schema, consultare Formato del messaggio di richieste di trasferimento file.

Esempio

Il seguente esempio mostra un documento XML di definizione attività di esempio salvato come `cleanuptask.xml`, che utilizza l'elemento `<managedCall>` per richiamare uno script Ant denominato `RunCleanup.xml`. Lo script `RunCleanup.xml` Ant deve trovarsi sul commandPath dell'agent di monitoraggio.

```
<?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>
```

L'elemento `<agent>` specifica il Managed File Transfer Agent configurato con lo script Ant denominato sul relativo commandPath.

La struttura `<call><command>...` definisce l'eseguibile o lo script che si desidera eseguire. Il comando utilizza un attributo `type` facoltativo che può avere uno dei seguenti valori:

antscript
Eseguire uno script Ant in una JVM separata.

eseguibile
Richiamare un programma eseguibile.

JCL
Richiamare un lavoro JCL.

Se si omette l'attributo `type` , viene utilizzato il valore predefinito eseguibile .

L'attributo `name` specifica il nome dello script Ant , dell'eseguibile o del lavoro JCL che si desidera eseguire, senza alcuna informazione sul percorso. L'agent ricerca lo script o il programma nelle ubicazioni specificate dalla proprietà commandPath nel file `agent.properties` dell'agent.

L'attributo `retrycount` specifica il numero di volte in cui provare a richiamare di nuovo il programma se il programma non restituisce un codice di ritorno di esito positivo. Il valore assegnato a questo attributo non deve essere negativo. Se non si specifica l'attributo `retrycount` , viene utilizzato il valore predefinito zero.

L'attributo `retrywait` specifica il tempo di attesa, in secondi, prima di ritentare il richiamo del programma. Il valore assegnato a questo attributo non deve essere negativo. Se non si specifica l'attributo `retrywait` , viene utilizzato il valore predefinito zero.

L'attributo `successrc` è un'espressione utilizzata per stabilire quando il richiamo del programma viene eseguito correttamente. Il codice di ritorno del processo per il comando viene valutato utilizzando questa espressione. Il valore può essere composto da una o più espressioni combinate con un carattere barra verticale (|) per indicare OR booleano o una e commerciale (&) per indicare il valore booleano AND. Ogni espressione può essere uno dei seguenti tipi di espressione:

- Un numero che indica un test di uguaglianza tra il codice di ritorno del processo e il numero.
- Un numero preceduto da un carattere maggiore di (>) per indicare un test maggiore di tra il numero e il codice di ritorno del processo.
- Un numero preceduto da un carattere minore di (<) per indicare un test minore di quello tra il numero e il codice di ritorno del processo.
- Un numero preceduto da un carattere punto esclamativo (!) per indicare un test non uguale tra il numero e il codice di ritorno del processo. Ad esempio:> 2 & < 7 &! 5 | 0 | 14 viene interpretato correttamente come i seguenti codici di ritorno: 0, 3, 4, 6, 14. Tutti gli altri codici di ritorno vengono interpretati come non riusciti.

Se non si specifica l'attributo `successrc` , viene utilizzato il valore predefinito zero. Ciò significa che si ritiene che il comando sia stato eseguito correttamente se, e solo se, restituisce un codice di zero.

Per uno script Ant , generalmente si specificano gli elementi `<target>` e `<property>` . I valori dell'elemento `<target>` devono corrispondere ai nomi di destinazione nello script Ant .

Per i programmi eseguibile, è possibile specificare elementi `<argument>` . Gli elementi argomento nidificati specificano gli argomenti da passare al programma richiamato come parte del richiamo del programma. Gli argomenti del programma vengono creati dai valori specificati dagli elementi dell'argomento nell'ordine in cui vengono rilevati gli elementi dell'argomento. È possibile specificare zero o più elementi argomento come elementi nidificati di un richiamo del programma.

L'amministratore definisce e avvia il monitor normalmente utilizzando il documento XML di definizione attività che include l'elemento `<managedCall>` . Ad esempio:

```
fteCreateMonitor -ma AGENT1 -mm QM1 -md /monitored -mn MONITOR01 -mt
   /tasks/cleanuptask.xml -pi 30 -pu seconds -tr match,*.go
```

Il percorso del documento XML di definizione del trasferimento deve essere sul filesystem locale da cui si esegue il comando **fteCreateMonitor** (in questo esempio `/tasks/cleanuptask.xml`). Il documento `cleanuptask.xml` viene utilizzato solo per creare il monitoraggio risorse. Tutte le attività a cui fa riferimento il documento `cleanuptask.xml` (script Ant o lavori JCL) devono essere nel percorso del comando dell'agent di monitoraggio. Quando la condizione del trigger di monitoraggio viene soddisfatta, tutte le variabili nell'XML di definizione dell'attività vengono sostituite con i valori effettivi dal monitoraggio. Quindi, ad esempio, ${FilePath} viene sostituito nel messaggio di richiesta inviato all'agente con `/monitored/cleanup.go`. Il messaggio di richiesta viene inserito sulla coda comandi dell'agente. Il processore comandi rileva che la richiesta è per una chiamata di programma e avvia il programma specificato. Se viene richiamato un comando di tipo `antscript` , viene avviata una nuova JVM e l'attività Ant viene eseguita nella nuova JVM. Per ulteriori informazioni sull'utilizzo della sostituzione di variabili, consultare Personalizzazione delle attività con la sostituzione di variabili.

Concetti correlati

“Personalizzazione delle attività di monitoraggio delle risorse MFT con la sostituzione della variabile” a pagina 242

Quando le condizioni del trigger di un controllo risorse attivo vengono soddisfatte, viene richiamata l'attività definita. Oltre a richiamare l'attività di trasferimento o di comando con lo stesso agent di destinazione o con lo stesso nome file di destinazione ogni volta, è anche possibile modificare la definizione dell'attività in fase di runtime. A tale scopo, inserire i nomi delle variabili nel file XML di definizione attività. Quando il controllo determina che le condizioni di trigger sono soddisfatte e che la definizione dell'attività contiene nomi di variabili, sostituisce i nomi di variabile con valori di variabile e richiama l'attività.

Riferimenti correlati

Formato del messaggio di richiesta di trasferimento file

proprietà commandPath MFT

Creazione di un documento di definizione attività modificando un documento generato

È possibile creare il documento di definizioni di attività di monitoraggio modificando il documento XML generato dall'opzione **-gt** di **fteCreateTransfer**.

Informazioni su questa attività

Il documento generato dispone di un elemento <request> seguito da <managedTransfer> . Per convertire questa definizione attività in una struttura <managedCall> valida, attenersi alla seguente procedura:

Procedura

1. Sostituire le tag di inizio e fine <managedTransfer> con le tag <managedCall> .
2. Rimuovere tutti gli elementi <schedule> e i nodi secondari.
3. Sostituire le tag di inizio e di fine <sourceAgent> con <agent> per corrispondere ai dettagli di configurazione dell'agent di monitoraggio.
4. Rimuovere elementi <destinationAgent> e <trigger> .
5. Rimuovere <item> elementi.
6. Rimuovere gli elementi preSourceCall, postSourceCall, preDestinationCallo postDestinationCall .
7. Inserire una nuova struttura <call>...</call> all'interno dell'elemento <transferSet> . Questa struttura contiene la definizione del comando come mostrato nel seguente esempio:

```
<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>
```

Esempio

È inoltre possibile conservare l'elemento <managedTransfer> includendo tutti i dettagli di trasferimento file e inserire fino a quattro chiamate di comando. In questo caso, inserire una selezione dei seguenti elementi di chiamata tra gli elementi <metaDataSet> e <item> :

Chiamata preSource
Richiamare un programma sull'agent di origine prima di avviare il trasferimento.

Chiamata postSource
Richiamare un programma sull'agent di origine dopo aver completato il trasferimento.

Chiamata preDestination
Richiamare un programma nell'agent di destinazione prima di avviare il trasferimento.

Chiamata postDestination
Richiamare un programma nell'agent di destinazione dopo aver completato il trasferimento.

Ciascuno di questi elementi prende la struttura dell'elemento <command> come descritto nell'esempio precedente. Lo schema FileTransfer.xsd definisce i tipi utilizzati dai vari elementi di chiamata.

Il seguente esempio mostra preSourceCall, postSourceCall, preDestinationCall e postDestinationCall in un documento di definizione attività:

```
⋮
    <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">
⋮
```

È possibile combinare diversi tipi di comando nel trasferimento. Gli elementi argomento, destinazione e proprietà sono facoltativi.

Monitoraggio di una directory e utilizzo della sostituzione della variabile

È possibile monitorare una directory utilizzando il comando **fteCreateMonitor** . Il valore di una variabile di sostituzione può essere sostituito nella definizione XML dell'attività e utilizzato per definire il funzionamento del trasferimento.

Informazioni su questa attività

In questo esempio, l'agent di origine è denominato AGENT_HOP. La directory monitorata da AGENT_HOP è denominata `/test/monitored`. L'agent esegue il polling della directory ogni 5 minuti.

Dopo che un file `.zip` è stato scritto nella directory, l'applicazione che lo scrive nella directory scrive un file trigger nella stessa directory. Il nome del file trigger è uguale al nome del file `.zip` , ma ha un'estensione file diversa. Ad esempio, dopo aver scritto il file `file1.zip` nella directory, il file `file1.go` viene scritto nella directory. Il monitoraggio risorse monitora l'indirizzario per file che corrispondono al modello `*.go` , quindi utilizza la sostituzione della variabile per richiedere un trasferimento del file `.zip` associato.

Procedura

1. Creare l'attività XML che definisce l'attività che il monitoraggio esegue quando viene attivato.

```
<?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>
```

Le variabili sostituite con i valori associati al file trigger sono evidenziate in **grassetto**. Questa attività XML viene salvata nel file `/home/USER1/task.xml`

2. Creare un monitoraggio risorse per monitorare l'indirizzario `/test/monitored`.

   Immettere il seguente comando:

```
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 utente o un programma scrive il file `jump.zip` nella directory `/test/monitored`, quindi scrive il file `jump.go` nella directory.
4. Il monitoraggio viene attivato dall'esistenza del file `jump.go`. L'agent sostituisce le informazioni relative al file trigger nell'XML dell'attività.

   Ciò determina la trasformazione dell'attività XML in:

```
<?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>
```

Risultati

Viene eseguito il trasferimento definito dall'XML dell'attività. Il file `jump.zip` viene letto dalla directory `/test/monitored` da AGENT_HOP e viene trasferito in un file denominato `/out/jump.zip` che si trova sul sistema su cui è in esecuzione AGENT_SKIP.

Concetti correlati

“Personalizzazione delle attività di monitoraggio delle risorse MFT con la sostituzione della variabile” a pagina 242

Quando le condizioni del trigger di un controllo risorse attivo vengono soddisfatte, viene richiamata l'attività definita. Oltre a richiamare l'attività di trasferimento o di comando con lo stesso agent di

destinazione o con lo stesso nome file di destinazione ogni volta, è anche possibile modificare la definizione dell'attività in fase di runtime. A tale scopo, inserire i nomi delle variabili nel file XML di definizione attività. Quando il controllo determina che le condizioni di trigger sono soddisfatte e che la definizione dell'attività contiene nomi di variabili, sostituisce i nomi di variabile con valori di variabile e richiama l'attività.

Attività correlate

“Configurazione delle attività di monitoraggio MFT per avviare comandi e script” a pagina 235
I monitoraggi risorse non sono limitati all'esecuzione di trasferimenti file come attività associata. È anche possibile configurare il monitor per richiamare altri comandi dall'agent di monitoraggio, inclusi i programmi eseguibili, gli script Ant o i lavori JCL. Per richiamare i comandi, modificare l'XML di definizione dell'attività di monitoraggio in modo da includere uno o più elementi di comando con i parametri di chiamata del comando corrispondenti, ad esempio argomenti e proprietà.

Riferimenti correlati

`fteCreateMonitor`: crea un monitoraggio risorse MFT

Esempio: configurazione di una risorsa MFT

È possibile specificare una coda IBM MQ come risorsa che deve essere monitorata da un monitoraggio risorse utilizzando il parametro **`-mq`** con il comando **`fteCreateMonitor`** .

Informazioni su questa attività

In questo esempio, la risorsa da monitorare è la coda *MONITORED_QUEUE*. Questa coda deve trovarsi sul gestore code dell'agent di monitoraggio, *QM_NEPTUNE*. La condizione per cui la coda è monitorata è la presenza di un gruppo completo di messaggi. L'attività da eseguire se la condizione è soddisfatta è definita nel file `task.xml`.

Nota: Non creare più di un controllo risorse per monitorare una singola coda. Se lo si fa, si verifica un comportamento imprevedibile.

Procedura

Immettere il seguente comando:

```
fteCreateMonitor -ma AGENT_NEPTUNE -mn myMonitor -mm QM_NEPTUNE -mq MONITORED_QUEUE
  -mt task.xml -tr completeGroups -pi 5 -pu minutes
```

Il controllo controlla la coda ogni cinque minuti per vedere se la condizione `completeGroups` è true. Se ci sono uno o più gruppi completi sulla coda, il monitoraggio esegue l'attività definita nel file `task.xml` una volta per ciascun gruppo completo.

Concetti correlati

“Personalizzazione delle attività di monitoraggio delle risorse MFT con la sostituzione della variabile” a pagina 242
Quando le condizioni del trigger di un controllo risorse attivo vengono soddisfatte, viene richiamata l'attività definita. Oltre a richiamare l'attività di trasferimento o di comando con lo stesso agent di destinazione o con lo stesso nome file di destinazione ogni volta, è anche possibile modificare la definizione dell'attività in fase di runtime. A tale scopo, inserire i nomi delle variabili nel file XML di definizione attività. Quando il controllo determina che le condizioni di trigger sono soddisfatte e che la definizione dell'attività contiene nomi di variabili, sostituisce i nomi di variabile con valori di variabile e richiama l'attività.

Attività correlate

“Configurazione delle attività di monitoraggio MFT per avviare comandi e script” a pagina 235
I monitoraggi risorse non sono limitati all'esecuzione di trasferimenti file come attività associata. È anche possibile configurare il monitor per richiamare altri comandi dall'agent di monitoraggio, inclusi i programmi eseguibili, gli script Ant o i lavori JCL. Per richiamare i comandi, modificare l'XML di definizione dell'attività di monitoraggio in modo da includere uno o più elementi di comando con i parametri di chiamata del comando corrispondenti, ad esempio argomenti e proprietà.

“Monitoraggio di una coda e utilizzo della sostituzione di variabili” a pagina 247
È possibile monitorare una coda e trasferire messaggi dalla coda monitorata a un file utilizzando il comando **fteCreateMonitor** . Il valore di qualsiasi proprietà del messaggio IBM MQ nel primo messaggio da leggere dalla coda monitorata può essere sostituito nella definizione XML dell'attività e utilizzato per definire il comportamento del trasferimento.

Riferimenti correlati

fteCreateMonitor: crea un monitoraggio risorse MFT

Personalizzazione delle attività di monitoraggio delle risorse MFT con la sostituzione della variabile

Quando le condizioni del trigger di un controllo risorse attivo vengono soddisfatte, viene richiamata l'attività definita. Oltre a richiamare l'attività di trasferimento o di comando con lo stesso agent di destinazione o con lo stesso nome file di destinazione ogni volta, è anche possibile modificare la definizione dell'attività in fase di runtime. A tale scopo, inserire i nomi delle variabili nel file XML di definizione attività. Quando il controllo determina che le condizioni di trigger sono soddisfatte e che la definizione dell'attività contiene nomi di variabili, sostituisce i nomi di variabile con valori di variabile e richiama l'attività.

Attenzione: I nomi delle variabili non sono sensibili al maiuscolo / minuscolo.

Le variabili utilizzate per la sostituzione sono disponibili solo per le condizioni trigger positive. Solo le condizioni trigger `match` e `fileSize` causano la sostituzione delle variabili. Se viene utilizzata una condizione `noMatch` e sono presenti nomi di variabili di sostituzione nella definizione dell'attività, l'attività non viene richiamata e il controllo genera un codice di ritorno di 110 e un messaggio di errore `BFGDM0060E`.

Se la risorsa monitorata è una coda

Il valore di qualsiasi proprietà del messaggio IBM MQ nel primo messaggio da leggere dalla coda monitorata può essere sostituito nella definizione XML dell'attività.

Le proprietà del messaggio definite dall'utente hanno come prefisso `usr.` ma non includono questo prefisso nel nome della variabile. I nomi delle variabili devono essere preceduti dal simbolo del dollaro ($) e racchiusi tra parentesi graffe {}.

Ad esempio, `${destFileName}` viene sostituito con il valore della proprietà del messaggio `usr.destFileName` del primo messaggio da leggere dalla coda di origine. Per ulteriori informazioni, consultare MQ message properties read by MFT from messages on source queues e “Monitoraggio di una coda e utilizzo della sostituzione di variabili” a pagina 247.

Se una variabile non è definita come una proprietà del messaggio, il monitoraggio riporta un errore BFGDM0060E e restituisce il codice di ritorno 110 (la sostituzione della variabile dell'attività di controllo non è riuscita). Inoltre, l'agent scrive il seguente messaggio di errore nel relativo log eventi (`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.
```

Se la registrazione del monitoraggio delle risorse moderata o verbose è abilitata per il monitoraggio, il monitoraggio scrive il seguente messaggio nel log degli eventi del monitoraggio delle risorse dell'agent (`resmoneventN.log`):

```
BFGDM0060E: A monitor task could not complete as a variable substitution
<variable name> was not present.
```

Consultare “Registrazione dei monitoraggi delle risorse MFT” a pagina 254 per ulteriori informazioni sulla registrazione del monitoraggio delle risorse.

La seguente tabella mostra quali variabili di sostituzione sono fornite per impostazione predefinita. Ad esempio, `${AGENTNAME}` viene sostituito con il nome dell'agent di monitoraggio risorse.

Tabella 10. Variabili di sostituzione fornite per impostazione predefinita	
Variabile	**Descrizione**
NomeAgent	Il nome dell'agent del controllo risorse.
QUEUENAME	Il nome della coda monitorata.
Codifica	La codifica dei caratteri del primo messaggio della coda o del primo messaggio di un gruppo.
MessageId	L'ID del messaggio IBM MQ del primo messaggio sulla coda o del primo messaggio nel gruppo.
GroupID	L'ID gruppo IBM MQ del gruppo o l'ID messaggio se viene trovato solo un singolo messaggio. Questa variabile è impostata solo se si stanno controllando gruppi completi.
CurrentTime	Una data/ora basata sull'ora locale in cui è stato attivato il monitoraggio. Il valore data / ora è univoco per l'agente.
CurrentTimeStampUTC	Una data/ora basata sull'ora, nel fuso orario UTC, in cui è stato attivato il monitoraggio. Il valore data / ora è univoco per l'agente.

Se la risorsa monitorata è una directory

La seguente tabella mostra la serie di nomi di variabili che possono essere sostituiti nella definizione XML dell'attività.

Tabella 11. Variabili che possono essere sostituite	
Variabile	**Descrizione**
FilePath	Il percorso completo del file di trigger.
FileName	La parte nome file del trigger.
LastModified	L'ora di ultima modifica del file di trigger. Questa ora è espressa come l'ora locale del fuso orario in cui è in esecuzione l'agent ed è formattata come ora ISO 8601.
Data LastModified	La data di ultima modifica del file di trigger. Questa data è espressa come la data locale del fuso orario in cui è in esecuzione l'agent ed è formattata come data ISO 8601.
LastModifiedTimeUTC	L'ora di ultima modifica del file di trigger. Questa ora è espressa come l'ora locale convertita nel fuso orario UTC e viene formattata come ora ISO 8601
LastModifiedDateUTC	La data di ultima modifica del file di trigger. Tale data viene espressa come data locale convertita nel fuso orario UTC e ha il formato data ISO 8601.
AgentName	Il nome dell'agent del controllo risorse.
CurrentTime	Una data/ora che si basa sull'ora locale in cui è stato attivato il monitoraggio. Il valore data / ora è univoco per l'agente.
CurrentTimeStampUTC	Una data / ora basata sull'ora nel fuso orario UTC in cui è stato attivato il monitoraggio. Il valore data / ora è univoco per l'agente.

Se la risorsa monitorata è un file trigger

La seguente tabella mostra la serie di nomi di variabili che è possibile sostituire quando un controllo risorse utilizza il contenuto di un file trigger per determinare i file che devono essere trasferiti.

Tabella 12. Variabili che possono essere sostituite quando si utilizza un file trigger	
Variabile	**Descrizione**
contentSource	Il nome percorso completo del file di origine.
contentDestination	Il nome percorso completo del file di destinazione.

I nomi delle variabili devono essere preceduti dal simbolo del dollaro ($) e racchiusi tra parentesi graffe, {}. Ad esempio, `${FilePath}` viene sostituito con il percorso file completo del file trigger corrispondente.

Ci sono due parole chiave speciali che possono essere applicate ai nomi delle variabili per fornire un ulteriore perfezionamento. Essi sono:

token
L'indice token da sostituire (a partire da 1 da sinistra e a partire da -1 da destra)

separatore
Un singolo carattere per suddivide in token il valore della variabile. Il valore predefinito è il carattere barra (/) su piattaforme AIX and Linux o il carattere barra rovesciata (\) su piattaforme Windows , ma il separatore può essere qualsiasi carattere valido che può essere presente nel valore della variabile.

Se la parola chiave separatore viene specificata in un nome di variabile, il valore della variabile viene suddiviso in token in base al carattere separatore.

Il valore assegnato alla parola chiave token viene utilizzato come indice per selezionare quale token utilizzare per sostituire il nome della variabile. L'indice del token è relativo al primo carattere nella variabile e inizia da 1. Se la parola chiave token non viene specificata, viene inserita l'intera variabile.

Tutti i valori sostituiti in un nome agent nell'XML del messaggio vengono trattati in modo non sensibile al maiuscolo / minuscolo. Tutti i nomi Managed File Transfer Agent sono in maiuscolo. Se il valore `Paris` viene sostituito in un attributo agent nell'XML del messaggio, questo valore viene interpretato come un riferimento all'agent `PARIS`.

Concetti correlati

“Esempi: sostituzione di variabili per le definizioni di monitoraggio risorse” a pagina 244
Esempi di sostituzione di variabili per le definizioni di monitoraggio risorse utilizzando XML e IBM MQ Explorer.

Attività correlate

Cosa fare se la sostituzione della variabile fa sì che più file passino a un singolo nome file

Esempi: sostituzione di variabili per le definizioni di monitoraggio risorse

Esempi di sostituzione di variabili per le definizioni di monitoraggio risorse utilizzando XML e IBM MQ Explorer.

Esempi che mostrano il modo in cui funziona la sostituzione delle variabili

Supponendo che il percorso del file del trigger corrispondente sia `c:\MONITOR\REPORTS\Paris\Report2009.doc` su Windows e `/MONITOR/REPORTS/Paris/Report2009.doc` su piattaforme AIX and Linux , le variabili vengono sostituite come mostrato nella seguente tabella.

Tabella 13. Modalità di sostituzione delle variabili	
Specifica della variabile	**Dopo la sostituzione della variabile**
${FilePath}	Windows :`c:\MONITOR\REPORTS\Paris\Report2009.doc` AIX and Linux : `/MONITOR/REPORTS/Paris/Report2009.doc`

Tabella 13. Modalità di sostituzione delle variabili (Continua)	
Specifica della variabile	**Dopo la sostituzione della variabile**
${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 : PROSPETTI AIX and Linux : Parigi

È anche possibile specificare un indice di token negativo per selezionare i token relativi all'ultimo carattere della variabile, come mostrato nella tabella seguente. Gli esempi nella tabella utilizzano lo stesso valore di variabile, `c:\MONITOR\REPORTS\Paris\Report2009.doc` su Windows e `/MONITOR/REPORTS/Paris/Report2009.doc` su AIX and Linux.

Tabella 14. Esempi di utilizzo di un indice token negativo	
Specifica della variabile	**Dopo la sostituzione della variabile**
${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 : Parigi AIX and Linux : Parigi
${FilePath{token=-4}}	Windows : MONITOR AIX and Linux : MONITOR

Le variabili utilizzate per la sostituzione sono disponibili solo per le seguenti condizioni di trigger positivo e l'opzione `noSizeChange` , che è un'eccezione alla regola di condizione di trigger positivo:

- `corrispondenza`
- `fileSize`
- `noSizeModifica`

Se viene utilizzata una condizione `noMatch` e sono presenti nomi di variabili di sostituzione nella definizione dell'attività, l'attività non viene richiamata e il controllo genera un codice di ritorno di 110 e un messaggio di errore `BFGDM0060E`.

Esempio di utilizzo di XML

La seguente definizione di attività di esempio XML utilizza il nome dell'agent di monitoraggio come agent di origine per il trasferimento (`Paris`), utilizza il penultimo nome di directory nel percorso file come nome

dell'agent di destinazione per il trasferimento (Report2009) e rinomina il file trasferito in modo che sia la root del nome file trigger con estensione .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>
```

Ciò determina la trasformazione dell'attività XML in:

```
<?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 variabile ${FilePath{token=-2}} nell'attributo agent dell'elemento <destinationAgent> viene sostituita con il valore Paris. Questo valore viene trattato in modo non sensibile al maiuscolo / minuscolo e interpretato come un riferimento all'agent PARIS.

Esempi che utilizzano IBM MQ Explorer

Quando si crea un monitoraggio risorse tramite IBM MQ Explorere una volta specificate le proprietà del monitoraggio e le condizioni di trigger, viene fornita l'opzione per aggiungere elementi di trasferimento al monitoraggio. I seguenti esempi dimostrano come è possibile utilizzare le variabili ${FilePath} e ${FileName} nel "**pannello Aggiungi un elemento di trasferimento**" per personalizzare i trasferimenti risultanti da una corrispondenza del monitoraggio risorse.

Esempio 1

Per trasferire semplicemente il file di origine in un'altra posizione quando viene soddisfatta una condizione di trigger, è possibile utilizzare la variabile ${FilePath} :

- Impostare l'origine **Nome file** su ${FilePath}.
- Dal menu a discesa di **Tipo** per la destinazione, selezionare **Directory**.

- Impostare la destinazione **Nome file** in modo che sia l'ubicazione in cui si desidera trasferire il file di origine, ad esempio, `C:\MFT\out\`.

Esempio 2

Per trasferire il file di origine in un'altra ubicazione e modificare l'estensione del file, la variabile `${FileName}` può essere utilizzata insieme alla variabile `${FilePath}` :

Nel seguente esempio si presume che il percorso del file di origine sia uguale a `C:\MONITOR\REPORTS\Paris\Report2009.doc`:

- Impostare l'origine **Nome file** su `${FilePath}`.
- Impostare il **Nome file** di destinazione come ubicazione in cui si desidera trasferire il file di origine, seguito da `${FileName{token=1}{separator=.}}`, seguito dalla nuova estensione del file. Ad esempio, potrebbe essere `C:\MFT\out\${FileName{token=1}{separator=.}}.rpt`, che equivale a `C:\MFT\out\Report2009.rpt` con il nome del file di origine.

Esempio 3

Per utilizzare parte del percorso file del file di origine per determinare la destinazione del trasferimento, la variabile `${FilePath}` può essere utilizzata insieme alle specifiche del token e del separatore.

Nel seguente esempio si assume che il percorso file del file di origine sia uguale a `C:\MONITOR\REPORTS\Paris\Report2009.doc`.

È possibile utilizzare parte del percorso del file di origine per determinare la destinazione del file. Utilizzando l'esempio di percorso file di `C:\MONITOR\REPORTS\Paris\Report2009.doc`, se il file deve essere trasferito in una cartella in base all'ubicazione del file di origine, ovvero `Paris` in questo esempio, è possibile effettuare le seguenti operazioni:

- Impostare l'origine **Nome file** su `${FilePath}`.
- Impostare la destinazione **Nome file** in modo che sia la destinazione in cui si trovano le cartelle per ciascuna ubicazione, quindi aggiungere la parte di destinazione del percorso file e il nome file. Ad esempio, potrebbe essere `C:\MFT\out\${FilePath{token=-2}{separator=\}}\${FileName}`, che equivale a `C:\MFT\out\Paris\Report2009.doc` con il nome del file di origine.

Concetti correlati

“Personalizzazione delle attività di monitoraggio delle risorse MFT con la sostituzione della variabile” a pagina 242

Quando le condizioni del trigger di un controllo risorse attivo vengono soddisfatte, viene richiamata l'attività definita. Oltre a richiamare l'attività di trasferimento o di comando con lo stesso agent di destinazione o con lo stesso nome file di destinazione ogni volta, è anche possibile modificare la definizione dell'attività in fase di runtime. A tale scopo, inserire i nomi delle variabili nel file XML di definizione attività. Quando il controllo determina che le condizioni di trigger sono soddisfatte e che la definizione dell'attività contiene nomi di variabili, sostituisce i nomi di variabile con valori di variabile e richiama l'attività.

Attività correlate

Cosa fare se la sostituzione della variabile fa sì che più file passino a un singolo nome file

Monitoraggio di una coda e utilizzo della sostituzione di variabili

È possibile monitorare una coda e trasferire messaggi dalla coda monitorata a un file utilizzando il comando **`fteCreateMonitor`** . Il valore di qualsiasi proprietà del messaggio IBM MQ nel primo messaggio da leggere dalla coda monitorata può essere sostituito nella definizione XML dell'attività e utilizzato per definire il comportamento del trasferimento.

Informazioni su questa attività

In questo esempio, l'agent di origine è denominato AGENT_VENUS, che si connette a QM_VENUS. La coda monitorata da AGENT_VENUS è denominata START_QUEUE e si trova su QM_VENUS. L'agent esegue il polling della coda ogni 30 minuti.

Quando un gruppo completo di messaggi viene scritto nella coda, l'attività di monitoraggio invia il gruppo di messaggi a uno dei diversi agent di destinazione, tutti connessi al gestore code QM_MARS. Il nome del file a cui viene trasferito il gruppo di messaggi viene definito dalla proprietà IBM MQ message `usr.fileName` sul primo messaggio del gruppo. Il nome dell'agent a cui viene inviato il gruppo di messaggi è definito dalla IBM MQ proprietà del messaggio `usr.toAgent` nel primo messaggio del gruppo. Se l'intestazione `usr.toAgent` non è impostata, il valore predefinito da utilizzare per l'agent di destinazione è AGENT_MAGENTA.

Quando si specifica `useGroups="true"`, se non si specifica anche `groupId="${GROUPID}"`, il trasferimento acquisisce solo il primo messaggio sulla coda. Ad esempio, se si sta utilizzando la sostituzione della variabile per generare il fileName, è possibile che il contenuto di a `.txt` non sia corretto. Ciò è dovuto al fatto che fileName viene generato dal monitoraggio, ma il trasferimento in realtà riceve un messaggio che non è quello che dovrebbe generare il file denominato fileName.

Procedura

1. Creare l'attività XML che definisce l'attività che il monitoraggio esegue quando viene attivato.

```
<?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>
```

   Le variabili sostituite con i valori delle intestazioni dei messaggi IBM MQ vengono evidenziati in **grassetto**. Questa attività XML viene salvata nel file `/home/USER1/task.xml`

2. Creare un monitoraggio risorse per monitorare la coda START_QUEUE.

   Immettere il seguente comando:

```
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 utente o un programma scrive un gruppo di messaggi nella coda START_QUEUE.

   Il primo messaggio in questo gruppo ha le seguenti proprietà del messaggio IBM MQ :

```
usr.fileName=larmer
usr.toAgent=AGENT_VIOLET
```

4. Il controllo viene attivato quando viene scritto il gruppo completo. L'agent sostituisce le proprietà del messaggio IBM MQ nell'XML dell'attività.

Ciò determina la trasformazione dell'attività XML in:

```
<?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>
```

Risultati

Viene effettuato il trasferimento definito dall'attività XML. Il gruppo completo di messaggi letti da START_QUEUE da AGENT_VENUS viene scritto in un file denominato `/reports/larmer.rpt` sul sistema su cui è in esecuzione AGENT_VIOLET.

Operazioni successive

Trasferimento di ogni messaggio in un file separato

Se si desidera monitorare una coda e trasferire ogni messaggio in un file separato, è possibile utilizzare una tecnica simile a quella descritta in precedenza in questo argomento.

1. Creare il controllo come descritto in precedenza, specificando il parametro **`-tr completeGroups`** nel comando **`fteCreateMonitor`** .
2. Nell'XML dell'attività specificare quanto segue:

```
<queue useGroups="true" groupId="${GROUPID}">START_QUEUE</queue>
```

Tuttavia, quando si inserono i messaggi nella coda di origine, non inserirli in un gruppo IBM MQ . Aggiungere le proprietà del messaggio IBM MQ a ciascun messaggio. Ad esempio, specificare la proprietà `usr.filename` con un valore di nome file univoco per ogni messaggio. Ciò fa sì che Managed File Transfer Agent consideri ciascun messaggio sulla coda di origine come un gruppo separato.

Concetti correlati

“Trasferimento dei dati dai messaggi ai file” a pagina 280
La funzione messaggio - a - file di Managed File Transfer consente di trasferire i dati da uno o più messaggi su una coda IBM MQ a un file, a un dataset (su z/OS) o a uno spazio file utente. Se si dispone di un'applicazione che crea o elabora messaggi IBM MQ , è possibile utilizzare la funzionalità messaggio - a - file di Managed File Transfer per trasferire questi messaggi a un file su qualsiasi sistema nella rete Managed File Transfer .

“Personalizzazione delle attività di monitoraggio delle risorse MFT con la sostituzione della variabile” a pagina 242
Quando le condizioni del trigger di un controllo risorse attivo vengono soddisfatte, viene richiamata l'attività definita. Oltre a richiamare l'attività di trasferimento o di comando con lo stesso agent di destinazione o con lo stesso nome file di destinazione ogni volta, è anche possibile modificare la definizione dell'attività in fase di runtime. A tale scopo, inserire i nomi delle variabili nel file XML di definizione attività. Quando il controllo determina che le condizioni di trigger sono soddisfatte e che la

definizione dell'attività contiene nomi di variabili, sostituisce i nomi di variabile con valori di variabile e richiama l'attività.

Cosa fare se i file di destinazione creati da un trasferimento avviato da un controllo risorse della coda contengono dati errati

Attività correlate

"Configurazione delle attività di monitoraggio MFT per avviare comandi e script" a pagina 235
I monitoraggi risorse non sono limitati all'esecuzione di trasferimenti file come attività associata. È anche possibile configurare il monitor per richiamare altri comandi dall'agent di monitoraggio, inclusi i programmi eseguibili, gli script Ant o i lavori JCL. Per richiamare i comandi, modificare l'XML di definizione dell'attività di monitoraggio in modo da includere uno o più elementi di comando con i parametri di chiamata del comando corrispondenti, ad esempio argomenti e proprietà.

"Esempio: configurazione di una risorsa MFT" a pagina 241
È possibile specificare una coda IBM MQ come risorsa che deve essere monitorata da un monitoraggio risorse utilizzando il parametro **-mq** con il comando **fteCreateMonitor** .

Riferimenti correlati

fteCreateMonitor: crea un monitoraggio risorse MFT

Proprietà dei messaggi MQ lette da MFT dai messaggi sulle code origine

Configurazione del comportamento dei tentativi di monitoraggio per i trasferimenti da messaggio a file

Se un trasferimento da messaggio a file attivato da un controllo risorse non riesce e lascia il gruppo di messaggi che ha attivato il controllo sulla coda, tale trasferimento viene reinoltrato a intervalli di polling successivi. Il numero di volte in cui il trasferimento viene inoltrato nuovamente è limitato dalla proprietà **monitorGroupRetryLimit** dell'agent di monitoraggio.

Informazioni su questa attività

Ogni volta che viene attivato un nuovo trasferimento da messaggio a file, viene generato un nuovo ID trasferimento per l'attività di trasferimento.

Se l'agent viene riavviato, il monitoraggio attiva nuovamente un trasferimento anche se il numero di volte in cui il trasferimento è stato attivato ha superato il valore di **monitorGroupRetryLimit** nel file `agent.properties` . Il valore della proprietà **monitorGroupRetryLimit** è il numero massimo di volte in cui un monitoraggio attiva nuovamente un trasferimento da messaggio a file se il gruppo di messaggi è ancora presente nella coda. Il valore predefinito di questa proprietà è 10. Il valore di questa proprietà può essere impostato su qualsiasi valore intero positivo o -1. Se il valore -1 viene specificato per questa proprietà, il monitor attiva di nuovo il trasferimento un numero illimitato di volte, fino a quando la condizione trigger non viene soddisfatta.

Se un tentativo di trasferimento fa sì che il numero di volte in cui il trasferimento è stato attivato superi il valore di **monitorGroupRetryLimit**, l'agent scrive un errore nel relativo log eventi.

Un singolo messaggio viene considerato come se fosse un singolo gruppo e il trasferimento viene attivato nuovamente ad ogni intervallo di polling, mentre il messaggio rimane nella coda e mentre il numero di volte in cui il trasferimento è stato attivato è inferiore al valore di **monitorGroupRetryLimit**.

Per impostare la proprietà **monitorGroupRetryLimit** sull'agent di monitoraggio, effettuare le seguenti operazioni:

Procedura

1. Arrestare l'agent di monitoraggio, utilizzando il comando **fteStopAgent** .
2. Modificare il file `agent.properties` per l'agent di monitoraggio per includere la seguente riga:

```
monitorGroupRetryLimit=number_of_retries
```

Il file `agent.properties` si trova nella directory *MQ_DATA_PATH*`/mqft/config/`*coordination_qmgr_name*`/agents/`*monitoring_agent_name*.

3. Avviare l'agent di monitoraggio utilizzando il comando **fteStartAgent** .

Attività correlate

“Esempio: configurazione di una risorsa MFT” a pagina 241
È possibile specificare una coda IBM MQ come risorsa che deve essere monitorata da un monitoraggio risorse utilizzando il parametro **-mq** con il comando **fteCreateMonitor** .

Utilizzo di un file trigger

È possibile utilizzare il contenuto di un file trigger in un controllo risorse per definire una serie di file da trasferire in una richiesta di trasferimento singola. Ogni volta che viene rilevato un file trigger corrispondente, il suo contenuto viene analizzato per i percorsi file di origine e facoltativamente per i percorsi file di destinazione. Questi percorsi file vengono quindi utilizzati per definire gli elementi file nel file XML di trasferimento attività specificato, inoltrato come singola richiesta di trasferimento all'agente. La definizione del monitoraggio risorse determina se il contenuto del trigger è abilitato.

È possibile abilitare il trigger del contenuto file quando si crea un controllo specificando il parametro **-tc** (contenuto trigger). Questo parametro **-tc** si applica solo alle opzioni trigger del file `match` e `noSizeChange`. Per ulteriori informazioni sulla creazione di un monitoraggio, consultare **fteCreateMonitor**: create an MFT resource monitor.

Quando si utilizza un file di contenuto trigger, il formato predefinito di ciascuna riga è:

- Un percorso file di origine singolo o
- Un percorso del file di origine e un percorso del file di destinazione, separati da una virgola

dove i caratteri spazio sono gestiti come parte dei percorsi file. È possibile modificare il formato di riga predefinito specificando i parametri **-tcr** e **-tcc** nel comando **fteCreateMonitor** . Per ulteriori informazioni, consultare “Opzioni avanzate” a pagina 252.

Una volta analizzato un file trigger, viene generato un elenco di percorsi file che vengono applicati all'XML dell'attività di trasferimento specificato. Come con tutti i monitor, il formato dell'XML dell'attività di trasferimento è un XML dell'attività di trasferimento completo generato dal comando **fteCreateTransfer** con un singolo elemento o file definito. Il singolo elemento deve utilizzare le variabili di sostituzione ${contentSource}e facoltativamente ${contentDestination}, come sostituzioni per i percorsi dei file di origine e di destinazione. Il monitoraggio espande l'XML dell'attività di trasferimento per includere un elemento file per ogni riga (percorso file) nel file trigger.

Non è possibile utilizzare il trigger del contenuto file con il parametro **-bs** perché il parametro **-tc** implica una richiesta di trasferimento per ogni file trigger.

Esempio

Il seguente esempio definisce un controllo da attivare su un file che termina con `trig` e legge i percorsi file in tale file.

```
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
```

Il comando **fteCreateTransfer** crea un file denominato `task.xml` per un file singolo con un percorso file origine ${contentSource}. Ad esempio:

```
<item checksumMethod="MD5" mode="binary">
    <source disposition="leave" recursive="false">
          <file>${contentSource}</file>
    </source>
</item>
```

Il comando **fteCreateMonitor** esegue una scansione dei file che terminano in `trig` nella directory `/home/trigdir` e utilizza i contenuti per creare una singola richiesta di trasferimento che si basa su `task.xml` per tutti i percorsi in tale file trigger. Il formato del file trigger deve essere un percorso file (solo origine) su ogni riga senza separatore virgola. Ad esempio:

```
/home/file/first.txt
/home/file/second.txt
/home/different/third.txt
⋮
```

Tutti i file vengono consegnati alla directory `/file/destdir` con il relativo nome file e non con il percorso file, ovvero `/home/file/first.txt` viene consegnato a `/file/destdir/first.txt`.

In alternativa, se si modifica il parametro **-dd /file/destdir** nel comando **fteCreateTransfer** in `-df ${contentDestination}` e il formato del contenuto di un file di trigger in *percorso file di origine*,*percorso file di destinazione*, è possibile definire percorsi di destinazione differenti per lo stesso agent di destinazione. Ad esempio:

```
/home/file/first.txt,/home/other/sixth.txt
```

L'ubicazione di destinazione diventa `/home/other/sixth.txt`.

Le variabili di sostituzione possono essere suddivise in token. Ad esempio, è possibile separare la parte del nome file dal percorso fornito utilizzando ${contentDestination{token=-1}}. Pertanto, se la destinazione **fteCreateTransfer** è definita come `-df /file/destdir/${contentDestination{token=-1}}`, la nuova destinazione per `/home/file/first.txt` è `/file/destdir/sixth.txt`.

Opzioni avanzate

È possibile modificare il formato riga predefinito per il contenuto del file trigger utilizzando il parametro **-tcr** *regex* . Fornire un'espressione regolare che corrisponda al formato riga richiesto e fornire uno o due gruppi di cattura. Il primo gruppo di cattura è l'origine e il secondo, facoltativo, gruppo di cattura è la destinazione. Ad esempio:

- Il percorso di origine e di destinazione sono separati da un trattino:

  ```
  ((?:[^-])+)-((?:[^-])+)
  ```

  In questo esempio, il separatore è definito in tre posizioni e tutte e tre le istanze del trattino, -, possono essere modificate in qualsiasi carattere. Assicurarsi di eseguire l'escape dei caratteri speciali.
- I percorsi di origine e di destinazione sono separati da una virgola con spazi finali. I commenti indicati da un cancelletto (#) vengono ignorati.

  ```
  ((?:[^,])+),((?:[^,])+) *(?:#.*)+
  ```

  I percorsi file non possono contenere il simbolo del numero (#). Generalmente, una voce è la seguente: `/home/source/from.txt,/home/destination/to.txt # some comment`.

Se si utilizza il parametro **-tcr** , assicurarsi che l'espressione regolare sia ben progettata e testata in modo che l'espressione possa rilevare gli errori e analizzare correttamente i file trigger.

È possibile invertire l'ordine della cattura utilizzando il parametro **-tcc destSrc** . Se si specifica questo parametro, il primo gruppo di cattura è il percorso del file di destinazione e il secondo gruppo è il percorso del file di origine.

Come vengono gestiti gli errori

File trigger vuoto
Se il file trigger è vuoto, il risultato non è un trasferimento file. Ossia, il controllo crea una richiesta di trasferimento, ma non viene specificato alcun elemento file.

File trigger con errori
Se una voce in un file trigger non riesce ad analizzare rispetto al formato previsto, non viene generata alcuna richiesta di trasferimento. Viene pubblicato un log degli errori di monitoraggio e l'errore viene registrato anche nel log eventi. Il file trigger viene contrassegnato come elaborato e il monitoraggio non tenta di elaborare nuovamente il file fino a quando il file non viene aggiornato.

XML attività di trasferimento non corrispondente
L'XML dell'attività di trasferimento deve corrispondere al file trigger, ossia se l'XML dell'attività di trasferimento ha sia ${contentSource} che ${contentDestination}, tutti i file trigger per tale monitoraggio devono avere percorsi file di origine e di destinazione e allo stesso modo per l'inverso. Nel primo caso, il monitoraggio riporta un errore di sostituzione di ${contentDestination} se il file trigger fornisce solo il percorso del file di origine.

Esempi

Il seguente esempio è un trigger di contenuto di base in cui il contenuto di un file trigger ha solo un percorso file di origine:

```
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
```

Il parametro **-tcr** definisce due gruppi di cattura di una sequenza di caratteri separati da un carattere spazio. Il parametro e l'opzione **-tcc destSrc** indicano che i gruppi di cattura devono essere elaborati come destinazione e come origine.

```
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
```

Concetti correlati

“Personalizzazione delle attività di monitoraggio delle risorse MFT con la sostituzione della variabile” a pagina 242
Quando le condizioni del trigger di un controllo risorse attivo vengono soddisfatte, viene richiamata l'attività definita. Oltre a richiamare l'attività di trasferimento o di comando con lo stesso agent di destinazione o con lo stesso nome file di destinazione ogni volta, è anche possibile modificare la definizione dell'attività in fase di runtime. A tale scopo, inserire i nomi delle variabili nel file XML di definizione attività. Quando il controllo determina che le condizioni di trigger sono soddisfatte e che la definizione dell'attività contiene nomi di variabili, sostituisce i nomi di variabile con valori di variabile e richiama l'attività.

Attività correlate

“Monitoraggio di una coda e utilizzo della sostituzione di variabili” a pagina 247
È possibile monitorare una coda e trasferire messaggi dalla coda monitorata a un file utilizzando il comando **fteCreateMonitor** . Il valore di qualsiasi proprietà del messaggio IBM MQ nel primo messaggio da leggere dalla coda monitorata può essere sostituito nella definizione XML dell'attività e utilizzato per definire il comportamento del trasferimento.

Riferimenti correlati

fteCreateMonitor: crea un monitoraggio risorse MFT

fteCreateTransfer: avviare un nuovo trasferimento file

Registrazione dei monitoraggi delle risorse MFT

È possibile ottenere informazioni diagnostiche sui monitoraggi risorse utilizzando la registrazione.

Informazioni su questa attività

È possibile utilizzare la registrazione per i monitoraggi delle risorse utilizzando il comando **`fteSetAgentLoglevel`** o il file `agent.properties` per controllare la registrazione del monitoraggio delle risorse.

Si noti che i punti di traccia esistenti sono ancora utilizzati per la cattura delle informazioni.

I log di monitoraggio delle risorse vengono scritti in un file denominato `resmonevent`*N*`.log`, dove *N* indica un numero; ad esempio, `resmonevent0.log`. I file di log eventi registrano diverse azioni che si verificano quando un monitor esegue il polling di una risorsa, ad esempio una directory o una coda.

Attenzione: Tutti i monitoraggi risorse di un agent scrivono nello stesso file di log.

Per un esempio di output di un file `resmonevent`*N*`.log` , consultare Cosa fare se il controllo risorse dell'indirizzario MFT non sta attivando i file.

La seguente tabella elenca il tipo di eventi che il controllo risorse scrive nel file di log. La terza colonna descrive il livello di log necessario per catturare ogni evento dove il livello più basso è INFO e il più alto è VERBOSE.

Tenere presente che l'impostazione di un livello di log superiore, scrive anche eventi di livello inferiore. Ad esempio, l'impostazione del livello di registrazione su MODERATO scrive anche eventi di livello INFO, ma non eventi di livello VERBOSE.

Numero	Evento	Livello di log	Descrizione
1	Monitor creato	INFORMAZIONI	È stato creato un monitoraggio risorse.
2	Monitoraggio eliminato	INFORMAZIONI	È stato eliminato un controllo risorse.
3	Monitoraggio arrestato	INFORMAZIONI	Un controllo risorse è stato arrestato.
4	Monitoraggio avviato	INFORMAZIONI	È stato avviato un controllo risorse.
5	Avvia sondaggio	INFORMAZIONI	Un monitoraggio risorse ha avviato un nuovo ciclo di polling.
6	Fine polling	INFORMAZIONI	Un ciclo di polling del controllo risorse è terminato.
7	Corrispondenza modello	VERBOSE	È stato trovato un file nell'indirizzario di controllo dei trigger o un messaggio in una coda che corrisponde al modello specificato.
8	Mancata corrispondenza modello	VERBOSE	È stato trovato un file non corrispondente nell'indirizzario di controllo dei trigger o un messaggio in una coda che non corrisponde al modello specificato.
9	Richiesta di trasferimento	INFORMAZIONI	Un trasferimento è stato avviato dal controllo risorse.
10	Directory troppo profonda	VERBOSE	L'indirizzario monitorato dal controllo risorse contiene più sottoindirizzari di cui eseguire il

Numero	Evento	Livello di log	Descrizione
			polling, rispetto al numero specificato nella configurazione del controllo risorse.
11	File bloccato	MODERATO	Il file trigger monitorato dal monitoraggio risorse è bloccato da un'altra elaborazione.
12	Dimensione file piccola	MODERATO	Il file trigger è più piccolo della dimensione specificata nella configurazione del controllo risorse.
13	Dimensione file instabile	MODERATO	Il file trigger viene modificato più frequentemente del previsto dalla configurazione del controllo risorse.
14	Troppi polling	MODERATO	Un controllo risorse ha eseguito il polling di un file trigger instabile troppe volte.
15	Elementi corrispondenti	INFORMAZIONI	Numero totale di file trigger trovati nell'indirizzario sottoposto a polling da un controllo risorse.
16	Elementi trasferimento	INFORMAZIONI	Numero totale di elementi nella richiesta di trasferimento.
17	FDC generato	MODERATO	Un controllo risorse ha generato un'eccezione.
18	Richiesta di trasferimento	INFORMAZIONI	Richiesta di trasferimento inoltrata dal controllo risorse.
19	Avvio monitoraggio non riuscito	MODERATO	Non è stato possibile avviare un controllo risorse.
20	Cronologia cancellata	INFORMAZIONI	Le informazioni sulla cronologia del controllo sono state eliminate.
21	Cancellazione cronologia monitor non riuscita	INFORMAZIONI	Il tentativo di cancellare le informazioni cronologiche del controllo ha avuto esito negativo.
22	ID trasferimento	INFORMAZIONI	L'ID della richiesta di trasferimento è stato inoltrato dal monitoraggio.
23	Batch	INFORMAZIONI	Numero totale di richieste di trasferimento per gli elementi corrispondenti: *N*, dove *N* è un numero.
V 9.4.0 24	Connesso	VERBOSE	Un monitoraggio risorse si è collegato al gestore code dell'agent.
V 9.4.0 25	Disconnesso	VERBOSE	Un monitoraggio risorse si è disconnesso dal gestore code dell'agent.
V 9.4.0 26	Errore durante la disconnessione	VERBOSE	Un monitoraggio risorse ha rilevato un problema durante la disconnessione dal gestore code dell'agent.

Procedura

- Per utilizzare il comando **fteSetAgentLoglevel** per attivare e disattivare la registrazione di monitoraggio risorse, consultare fteSetAgentLogLevel per una descrizione del parametro **logMonitor** e per esempi su come utilizzare le diverse opzioni.

- Per utilizzare il file `agent.properties` per controllare la registrazione del monitoraggio risorse, consultare Il file MFT `agent.properties` per una descrizione delle ulteriori proprietà che consentono di eseguire le seguenti attività di registrazione:
  - Attiva o disattiva la registrazione
  - Limita la dimensione di ciascun file di log
  - Limita il numero di log che i monitoraggi risorse possono generare

Esempio

Le seguenti serie di messaggi di esempio `verbose` livello di registrazione per l'agent HA2, sul gestore code 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>
```

Riferimenti correlati

Comando di livello fteSetAgentLog
Il file MFT `agent.properties`

Avvio di un monitoraggio risorse MFT

Da IBM MQ 9.3.0, è possibile avviare i monitoraggi delle risorse senza dover arrestare o riavviare un agent utilizzando il comando **`fteStartMonitor`** .

Prima di iniziare

Se la gestione dell'autorizzazione utente è stata abilitata impostando l'attributo **`authorityChecking`** su `vero` nel file `agent.properties` , è necessario disporre dell'autorizzazione `Monitora` o `Operazioni di monitoraggio` per avviare un monitoraggio risorse. Per ulteriori informazioni sulla gestione delle autorizzazioni utente, consultare Limitazione delle autorizzazioni utente sulle MFT azioni agent.

Informazioni su questa attività

È possibile eseguire il comando **`fteStartMonitor`** da qualsiasi sistema in cui è installato il componente dei comandi Managed File Transfer , il che significa che è possibile avviare un monitoraggio delle risorse da qualsiasi punto e non sono limitati al sistema in cui è in esecuzione l'agent che possiede il monitoraggio delle risorse. Per informazioni sui parametri obbligatori e facoltativi per questo comando, consultare fteStartMonitor (avvio di un MFT monitoraggio risorse).

Procedura

- Per individuare lo stato di un agente prima o dopo l'esecuzione del comando **`fteStartMonitor`** , utilizzare il comando **`fteListMonitors`** con il parametro **`-v`** come mostrato nel seguente esempio:

  ```
  fteListMonitors -ma monitoring_agent_name -v
  ```

- Per avviare un monitoraggio risorse in un agente in esecuzione sulla stessa macchina, immettere il seguente comando **`fteStartMonitor`** :

```
fteStartMonitor -mn monitor_name -ma agent_name
```

- Per avviare un monitoraggio risorse in un agente in esecuzione su una macchina differente, immettere il seguente comando **fteStartMonitor** :

  ```
  fteStartMonitor -mn monitor_name -ma agent_name -mm AgentQueueManager
  ```

  Se il gestore code comandi è anche il gestore code agent per l'agent di monitoraggio, il parametro **-mm** è facoltativo, altrimenti è necessario specificare il gestore code agent con il parametro **-mm** .

Risultati

Se l'agent è in esecuzione, il monitoraggio delle risorse viene avviato se è attualmente arrestato. Il comando emette i seguenti messaggi e registra un evento nel `output0.log`dell'agent.

```
BFGCL0816I: È stata emessa una richiesta di avvio del monitoraggio risorse 'nome_monitoraggio'
dell'agent 'nome_agent'.
BFGCL0251I: La richiesta è stata completata correttamente.
```

Per informazioni sui messaggi emessi dal comando se non è in grado di avviare il monitoraggio delle risorse, consultare fteStartMonitor (avvio di un MFT monitoraggio delle risorse).

Concetti correlati

“Concetti di monitoraggio delle risorse MFT” a pagina 230
Una panoramica dei concetti chiave della funzione di monitoraggio delle risorse Managed File Transfer .

Attività correlate

“Arresto di un monitoraggio risorse MFT” a pagina 257
Da IBM MQ 9.3.0, è possibile arrestare i monitoraggi delle risorse senza dover arrestare o riavviare un agent utilizzando il comando **fteStopMonitor** .

Riferimenti correlati

fteStartMonitor (avvia un controllo risorse MFT)

Arresto di un monitoraggio risorse MFT

Da IBM MQ 9.3.0, è possibile arrestare i monitoraggi delle risorse senza dover arrestare o riavviare un agent utilizzando il comando **fteStopMonitor** .

Prima di iniziare

Se la gestione delle autorizzazioni utente è stata abilitata impostando l'attributo **authorityChecking** su `vero` nel file `agent.properties` , è necessario disporre dell'autorizzazione `Monitora` o `Operazioni di monitoraggio` per arrestare un monitoraggio delle risorse. Per ulteriori informazioni sulla gestione delle autorizzazioni utente, consultare Limitazione delle autorizzazioni utente sulle MFT azioni agent.

Informazioni su questa attività

È possibile eseguire il comando **fteStopMonitor** da qualsiasi altro sistema in cui è installato il componente dei comandi Managed File Transfer , il che significa che è possibile arrestare un monitoraggio risorse da qualsiasi punto e non sono limitati al sistema in cui è in esecuzione l'agent proprietario del monitoraggio risorse. Per informazioni sui parametri richiesti e facoltativi per questo comando, consultare fteStopMonitor (stop an MFT resource monitor).

Quando un controllo risorse viene arrestato, scrive un messaggio nel log eventi del monitoraggio risorse dell'agent, `resmoneventnumber.log`. Se il monitoraggio risorse viene arrestato con il comando **fteStopMonitor** , il messaggio include il nome dell'utente che ha emesso la richiesta di arresto:

```
Monitoraggio risorse arrestato dall'utente '< mquser_id>'
```

Un monitoraggio delle risorse viene avviato automaticamente se il relativo agent viene riavviato, anche se il monitoraggio delle risorse è stato precedentemente arrestato utilizzando il comando **fteStopMonitor** .

Gli agent elaborano le richieste di arresto del monitoraggio in modo seriale anziché in parallelo, quindi, ad esempio, se un agent riceve una richiesta di arresto del monitoraggio M1 e quindi un'altra richiesta di arresto del monitoraggio M2 in rapida successione, arresta M1 prima di tentare di arrestare M2.

Procedura

- Per individuare lo stato di un agente prima o dopo l'esecuzione del comando **fteStopMonitor** , utilizzare il comando **fteListMonitors** con il parametro **-v** come mostrato nel seguente esempio:

  ```
  fteListMonitors -ma monitoring_agent_name -v
  ```

- Per arrestare un monitoraggio risorse in un agente in esecuzione sulla stessa macchina, immettere il seguente comando **fteStopMonitor** :

  ```
  fteStopMonitor -mn monitor_name -ma agent_name
  ```

- Per arrestare un monitoraggio risorse in un agente in esecuzione su una macchina differente, immettere il comando **fteStopMonitor** nel modo seguente:

  ```
  fteStopMonitor -mn monitor_name -ma agent_name -mm AgentQueueManager
  ```

  Se il gestore code comandi è anche il gestore code agent per l'agent di monitoraggio, il parametro **-mm** è facoltativo, altrimenti è necessario specificare il gestore code agent con il parametro **-mm** .

Risultati

Se l'agent è in esecuzione, il monitoraggio delle risorse viene arrestato se è attualmente avviato. Il comando emette i seguenti messaggi e registra un evento nel `output0.log`dell'agent.

```
BFGCL0813I: È stata inoltrata una richiesta di arresto del monitoraggio risorse 'MNTR' dell'agent
'SOURCE'.
BFGCL0251I: La richiesta è stata completata correttamente.
```

Per informazioni sui messaggi emessi dal comando se non è in grado di arrestare il monitoraggio risorse, consultare fteStopMonitor (stop an MFT resource monitor)

Concetti correlati

“Concetti di monitoraggio delle risorse MFT” a pagina 230
Una panoramica dei concetti chiave della funzione di monitoraggio delle risorse Managed File Transfer .

Attività correlate

“Avvio di un monitoraggio risorse MFT” a pagina 256
Da IBM MQ 9.3.0, è possibile avviare i monitoraggi delle risorse senza dover arrestare o riavviare un agent utilizzando il comando **fteStartMonitor** .

Riferimenti correlati

fteStopMonitor (arresta un controllo risorse MFT)

Backup e ripristino dei monitoraggi delle risorse MFT

È possibile eseguire il backup dei monitoraggi risorse che si desidera rendere disponibili per un utilizzo futuro esportando le relative definizioni in un file XML che è possibile quindi importare per creare un nuovo monitoraggio risorse dal backup.

Informazioni su questa attività

Potrebbe essere necessario eseguire il backup dei monitoraggi risorse precedentemente definiti in modo da poterne riutilizzare le definizioni in futuro, ad esempio per ricreare i monitoraggi risorse in un'infrastruttura differente o se un monitoraggio risorse deve essere ricreato a causa di problemi del gestore code.

È possibile eseguire il backup di una singola definizione del gestore risorse utilizzando il comando **fteCreateMonitor** o il comando **fteListMonitors** con il parametro **-ox** . In entrambi i casi, viene

eseguito il backup della definizione del gestore risorse esportandola in un file XML. È possibile utilizzare il parametro **-ix** del comando **fteCreateMonitor** per creare un nuovo gestore risorse importando la definizione dal file XML.

Con il parametro **-ox** , è possibile eseguire il backup di una sola definizione di monitoraggio risorse alla volta.

Il parametro **-od** viene aggiunto al comando **fteListMonitors** . Specificando questo parametro, è possibile eseguire il backup di più di un controllo risorse alla volta esportando le relative definizioni in massa in una directory specificata. Ogni definizione di monitoraggio delle risorse viene salvata in un file XML separato con un nome nel formato *agent name*.*monitor name*.xml.

Il parametro **-od** è particolarmente utile se si dispone di un numero elevato di monitoraggi delle risorse di cui si desidera eseguire il backup perché è necessario eseguire il comando **fteListMonitors -od** una sola volta, invece di dover eseguire il comando **fteListMonitors -ox** separatamente per ciascuna definizione di risorsa o utilizzare uno script separato per eseguire il comando **fteListMonitors -ox** per ciascun monitoraggio delle risorse.

Procedura

- Per eseguire il back up della definizione di un controllo risorse esportandolo in un file XML, utilizzare uno dei seguenti comandi:
  - Il comando **fteCreateMonitor** con il parametro **-ox** .
  - Il comando **fteListMonitors** con il parametro **-ox** .

  Quando si utilizza il parametro **-ox** , è necessario anche specificare i parametri **-ma** e **-mn** , come mostrato nel seguente esempio:

  ```
  fteListMonitors -ma AGENT1 -mn MONITOR1 -ox filename1.xml
  ```

- Per eseguire il backup di più definizioni di monitoraggio risorse esportandole in file XML in una directory specificata, utilizzare il comando **fteListMonitors** con il parametro **-od** come mostrato nel seguente esempio:

  ```
  fteListMonitors -od /usr/mft/resmonbackup
  ```

  È necessario specificare una directory di destinazione valida quando si esegue il backup dei monitoraggi risorse in massa. Non specificando un percorso di destinazione si genera un messaggio di errore come mostrato nel seguente esempio:

  ```
  BFGCL0762E: Directory output non specificata. Eseguire di nuovo il comando specificando un
  percorso valido.
  ```

  Il parametro **-od** non deve essere combinato con il parametro **-ox** , altrimenti viene visualizzato il seguente messaggio di errore:

  ```
  BFGCL0761E: Non è valido specificare insieme i parametri '- od' e '- ox'.
  ```

  È possibile definire una particolare serie di monitoraggi risorse da includere nel backup. Ad esempio, utilizzando il parametro **-ma** per specificare il nome di un agente, è possibile eseguire il backup di tutti i controlli risorse per tale agente, come mostrato nel seguente esempio:

  ```
  fteListMonitors -ma AGENT1 -od /usr/mft/resmonbackup
  ```

  È anche possibile utilizzare la corrispondenza con caratteri jolly includendo un carattere asterisco (*) quando si definisce un pattern da utilizzare per la corrispondenza dei nomi agent e / o dei nomi di monitoraggio. Il seguente esempio esegue il backup di tutti i monitoraggi delle risorse che hanno nomi che corrispondono a un pattern specificato e che si trovano in un agent con un nome che corrisponde a un pattern specificato:

  ```
  fteListMonitors -ma AGENT* -mn MON* -od /usr/mft/resmonbackup
  ```

  Mentre il comando è in esecuzione, vengono visualizzati i seguenti messaggi di report di avanzamento:

```
È stato rilevato un totale di numero definizioni di monitoraggio risorse corrispondenti.
indice di numero definizioni di monitoraggio risorse salvate nel filesystem.
```

Se si sta utilizzando l'opzione verbose, il totale parziale è ancora visualizzato, ma invece di visualizzarlo

```
indice di numero definizioni di monitoraggio risorse salvate nel filesystem
```

il comando visualizza il nome della definizione di monitor che si sta salvando, ad esempio:

```
BFGCL0762I: Definizione del monitoraggio 'FILEMON' dell'agent 'XFERAGENT' salvata come
FILEMON.XFERAGENT.XML nel filesystem.
```

- Per eseguire il backup di un controllo risorse per un determinato agente esportandolo in un file XML in una directory specificata, utilizzare il comando **fteListMonitors** con il parametro **-od** :

```
fteListMonitors -ma AGENT1 -mn MONITOR1 -od /usr/mft/resmonbackup
```

L'uso del parametro **-od** per eseguire il backup di un singolo monitoraggio delle risorse è simile all'utilizzo del parametro **-ox** , tranne che il nome del file di output è nel formato *agent name*.*monitor name*.xml.

- Per ripristinare le definizioni di monitoraggio delle risorse da un backup, utilizzare il comando **fteCreateMonitor** con il parametro **-ix** come mostrato nel seguente esempio:

```
fteCreateMonitor -ix file name
```

Per ulteriori esempi su come utilizzare il parametro **-od** , consultare fteListMonitors: list MFT resource monitors.

Riferimenti correlati

fteCreateMonitor: crea un monitoraggio risorse MFT

fteListMonitoraggi: elenco MFT monitoraggi risorse

Cancellazione della cronologia del controllo risorse

È possibile cancellare la cronologia di un controllo risorse in modo che un'altra richiesta di trasferimento file possa essere inoltrata per un file che non è stato trasferito in precedenza a causa di un errore. Per cancellare la cronologia del monitoraggio risorse, è possibile utilizzare il comando **fteClearMonitorHistory** o IBM MQ Explorer.

Prima di iniziare

Se la gestione delle autorizzazioni utente è stata abilitata impostando l'attributo **authorityChecking** su `true` nel file `agent.properties` , l'utente che cancella la cronologia di controllo deve disporre dell'autorizzazione appropriata, come mostrato nella seguente tabella.

*Tabella 15. Autorizzazione utente richiesta per eseguire il comando **fteClearMonitorHistory***

Utente che cancella la cronologia del monitoraggio	Autorizzazione di accesso MFT	Autorizzazione richiesta
Lo stesso utente di quello che ha creato il monitoraggio risorse.	Monitor	SFOGLIA su SYSTEM.FTE.AUTHMON1.<*nome_agent_monitoraggio*> Si tratta della stessa autorizzazione richiesta per creare o cancellare il controllo risorse.
Qualsiasi utente diverso dall'utente che ha creato il monitoraggio risorse.	Operazioni di monitoraggio	SET su SYSTEM.FTE.AUTHPS1.<*nome_agent*> Si tratta della stessa autorizzazione richiesta per cancellare il controllo risorse.

Per ulteriori informazioni sulla gestione delle autorizzazioni utente, consultare Limitazione delle autorizzazioni utente sulle MFT azioni agent.

Se un utente senza l'autorizzazione richiesta tenta di cancellare la cronologia del monitoraggio delle risorse, il comando **fteClearMonitorHistory** emette un messaggio di errore e registra l'errore nel file `output0.log` dell'agent. Per ulteriori informazioni, consultare fteClearMonitorHistory: clear resource monitor history.

Informazioni su questa attività

Se è stato avviato un trasferimento file e un file non può essere trasferito per qualsiasi motivo, il controllo risorse non seleziona nuovamente questo file per il trasferimento nel successivo polling poiché la cronologia del controllo indica che il file è stato visualizzato in un polling precedente e non è stato modificato da allora (consultare "Concetti di monitoraggio delle risorse MFT" a pagina 230).

Prima di IBM MQ 9.1.3, se un file non riesce a trasferire, il trasferimento file può essere avviato di nuovo solo se il file viene eliminato e quindi nuovamente inserito nella directory, se il file viene aggiornato in modo che l'attributo della data dell'ultima modifica venga modificato o se il monitoraggio risorse stesso viene ricreato.

Tuttavia, è possibile cancellare la cronologia del controllo risorse utilizzando il comando **fteClearMonitorHistory** o utilizzando IBM MQ Explorer. La cancellazione della cronologia consente ad un'altra richiesta di trasferimento per un file che non è riuscito a trasferire di essere inoltrato senza la necessità di eliminare il file e quindi riposizionarlo nella directory oppure aggiornare il file per modificare l'attributo della data dell'ultima modifica, che è utile, ad esempio, in situazioni in cui è necessario trasferire il file ma non è possibile modificare il file. Essere in grado di cancellare la cronologia di un monitoraggio risorse significa anche che non è necessario ricreare il monitoraggio risorse per inoltrare un'altra richiesta di trasferimento per un file che non è riuscito a trasferire.

z/OS Il membro SCSQFCMD di esempio fornito con Managed File Transfer su z/OS include uno script JCL per cancellare la cronologia di un controllo.

Procedura

- Per utilizzare il comandi **fteClearMonitorHistory** per cancellare la cronologia del controllo risorse, immettere il comando nel formato seguente:

  ```
  fteClearMonitorHistory -p <configuration> -ma <agent name> -mn <monitor name> -w 1000
  ```

  Sono richiesti solo i parametri **-ma** e **-mn** . Tutti gli altri parametri sono facoltativi. Per ulteriori informazioni su come utilizzare il comando **fteClearMonitorHistory** , inclusi gli esempi, consultare fteClearMonitorHistory: clear resource monitor history.

  Se la cronologia viene cancellata correttamente, il comando visualizza il seguente messaggio:

  ```
  BFGCL0780I: È stata emessa una richiesta di cancellazione della cronologia del monitoraggio
  delle risorse 'nome monitoraggio' dell'agent 'nome agent'.
  BFGCL0251I: La richiesta è stata completata correttamente.
  ```

  e registra la riuscita nel file `output0.log` dell'agent.

  Se il tentativo di cancellare la cronologia del monitoraggio risorse ha esito negativo, **fteClearMonitorHistory** emette un messaggio di errore e registra l'errore nel file `output0.log` dell'agent.
- Per utilizzare la vista Monitoraggio risorse nel plug-in IBM MQ Explorer MFT per cancellare la cronologia del monitoraggio risorse, fare clic con il tasto destro del mouse sul monitoraggio risorse e selezionare **Cancella cronologia** dal menu a discesa.

  Se la cronologia viene cancellata correttamente, viene visualizzato il seguente messaggio:

  ```
  BFGUI00171: La cronologia del monitoraggio risorse è stata cancellata correttamente.
  ```

  Se il tentativo di cancellare la cronologia ha esito negativo, viene visualizzato un messaggio di errore. Ad esempio:

BFGUI0016E Impossibile cancellare la cronologia del controllo risorse specificato - motivo 2059

Utilizzo dei modelli di trasferimento file

È possibile utilizzare i modelli di trasferimento file per memorizzare le impostazioni di trasferimento file comuni per trasferimenti ripetuti o complessi. Creare un modello di trasferimento dalla riga comandi utilizzando il comando **`fteCreateTemplate`** oppure utilizzare IBM MQ Explorer per creare un modello di trasferimento utilizzando la procedura guidata **Crea nuovo modello per il trasferimento file gestito** oppure salvare un modello mentre si sta creando un trasferimento file selezionando la casella di spunta **Salva impostazioni di trasferimento come modello** . La finestra **Template di trasferimento** visualizza tutti i template di trasferimento che sono stati creati nella rete Managed File Transfer .

Informazioni su questa attività

Per creare un modello di trasferimento dalla riga comandi, utilizzare il comando fteCreateTemplate . Quindi, quando si desidera inoltrare un modello di trasferimento creato sulla riga di comando, fare clic su **Inoltra** in IBM MQ Explorer.

Per visualizzare i modelli di trasferimento in IBM MQ Explorer, utilizzare la seguente procedura:

Procedura

1. Espandere **Managed File Transfer** nella vista Navigator . **Managed File Transfer Central** viene visualizzato nella vista Contenuto.
2. Tutti i gestori code di coordinamento sono elencati nella vista Navigator . Espandere il nome del gestore code di coordinamento utilizzato per il trasferimento pianificato. Se si desidera modificare il gestore code di coordinamento a cui si è connessi, fare clic con il tasto destro del mouse sul nome del gestore code di coordinamento che si desidera utilizzare nella vista Navigator e fare clic su **Connetti**.
3. Fare clic su **Trasferisci modelli**. Viene visualizzata la finestra **Modelli di trasferimento** nella vista Contenuto.
4. La finestra **Modelli di trasferimento** elenca i seguenti dettagli relativi ai trasferimenti file:
   a) **Nome** Il nome del modello di trasferimento file.
   b) **Origine** Il nome dell'agent utilizzato per trasferire il file dal sistema di origine.
   c) **File di origine** Il nome del file da trasferire sul sistema host.
      Espandere le informazioni del modello di trasferimento per visualizzare questo campo.
   d) **Destinazione** Il nome dell'agent utilizzato per ricevere il file sul sistema di destinazione.
   e) **File di destinazione** Il nome del file dopo che è stato trasferito al sistema di destinazione.
      Espandere le informazioni del modello di trasferimento per visualizzare questo campo.
   f) **Inizio pianificato (fuso orario selezionato)** La data e l'ora in cui è pianificato l'avvio del trasferimento file nel fuso orario utilizzato dall'amministratore. Per modificare il fuso orario visualizzato, fare clic su **Finestra** > **Preferenze** > **IBM MQ Explorer** > **Managed File Transfer** e selezionare un fuso orario alternativo dall'elenco **Fuso orario:** . Fare clic su **OK**.
   g) **Eventi trigger** Il tipo di evento che attiva l'avvio del trasferimento file. Il tipo può essere uno dei seguenti valori: `esiste`, `non esiste` o `supera`.

Risultati

Per aggiornare quanto visualizzato nella finestra **Modelli di trasferimento** , fare clic sul pulsante Aggiorna  sulla barra degli strumenti della vista Contenuto.

Per inoltrare un template di trasferimento e avviare il trasferimento definito nel template, fare clic con il tasto destro del mouse sul nome del template e fare clic su **Inoltra**.

Per cambiare un modello di trasferimento, fare clic con il pulsante destro del mouse sul nome del modello e selezionare **Modifica**. Tutti i file inclusi nel modello originale sono elencati come parte di un gruppo di trasferimenti, anche se non sono stati inclusi come parte di un gruppo nel modello originale. Se si

desidera rimuovere un file dal modello, è necessario selezionare la specifica file dal gruppo e fare clic su **Rimuovi selezionati**. Se si desidera aggiungere nuove specifiche file al modello, utilizzare i campi nel pannello del modello e fare clic su **Aggiungi al gruppo** . Una volta apportate le proprie modifiche, viene richiesto di fornire un nuovo nome al template modificato.

Per creare un trasferimento file da un modello di trasferimento, fare clic con il tasto destro del mouse sul nome del modello e selezionare **Modifica come nuovo trasferimento**.

Per creare una copia duplicata di un modello di trasferimento, fare clic con il tasto destro del mouse sul nome del modello e selezionare **Duplica**. Il modello di trasferimento duplicato viene salvato automaticamente con lo stesso nome del modello originale, aggiunto con "(copia)".

Per eliminare un modello di trasferimento, fare clic con il tasto destro del mouse sul nome del modello e selezionare **Elimina**.

Attività correlate

“Creazione di un modello di trasferimento file utilizzando IBM MQ Explorer” a pagina 263
È possibile creare un modello di trasferimento file da IBM MQ Explorer o dalla riga comandi. È quindi possibile utilizzare tale modello per creare nuovi trasferimenti file utilizzando i relativi dettagli oppure inoltrare il modello per avviare il trasferimento file.

Riferimenti correlati

`fteCreateTemplate`: crea un nuovo modello di trasferimento file

Modelli fteList

Modelli fteDelete

Creazione di un modello di trasferimento file utilizzando IBM MQ Explorer

È possibile creare un modello di trasferimento file da IBM MQ Explorer o dalla riga comandi. È quindi possibile utilizzare tale modello per creare nuovi trasferimenti file utilizzando i relativi dettagli oppure inoltrare il modello per avviare il trasferimento file.

Informazioni su questa attività

Per creare un modello di trasferimento file dalla riga comandi, utilizzare il comando fteCreateTemplate .

Per creare un template di trasferimento file utilizzando la procedura guidata **Crea nuovo template per Managed File Transfer** in IBM MQ Explorer, effettuare le seguenti operazioni:

Procedura

1. Nella vista Navigator , fare clic su **Managed File Transfer**. **Managed File Transfer Central** viene visualizzato nella vista Contenuto.
2. Tutti i gestori code di coordinamento vengono visualizzati nella vista Navigator . Espandere il nome del gestore code di coordinamento utilizzato per il trasferimento pianificato. Se si desidera modificare il gestore code di coordinamento a cui si è connessi, fare clic con il tasto destro del mouse sul nome del gestore code di coordinamento che si desidera utilizzare nella vista Navigator e fare clic su **Connetti**.
3. Avviare la procedura guidata **Crea nuovo modello per Managed File Transfer** facendo clic con il pulsante destro del mouse su **Modelli di trasferimento** e facendo clic su **Nuovo modello**.
4. Seguire le istruzioni sui pannelli della procedura guidata. Per ogni pannello viene fornita una guida sensibile al contesto. Per accedere alla guida sensibile al contesto su Windows, premere F1. Su Linux, premere `Ctrl+F1` o `Shift+F1`.

   Se è stato creato un modello che contiene tutti i dettagli di trasferimento richiesti, assicurarsi di selezionare la casella di controllo **Salva impostazioni di trasferimento come modello** nella pagina **Riepilogo trasferimento** se questa casella di controllo non è già selezionata. Immettere anche un nome per il modello nel campo Nome. Se si crea un modello che non contiene ancora tutti i dettagli di trasferimento richiesti, la casella di spunta **Salva impostazioni di trasferimento come modello** viene selezionata automaticamente.

Attività correlate

"Utilizzo dei modelli di trasferimento file" a pagina 262

È possibile utilizzare i modelli di trasferimento file per memorizzare le impostazioni di trasferimento file comuni per trasferimenti ripetuti o complessi. Creare un modello di trasferimento dalla riga comandi utilizzando il comando **fteCreateTemplate** oppure utilizzare IBM MQ Explorer per creare un modello di trasferimento utilizzando la procedura guidata **Crea nuovo modello per il trasferimento file gestito** oppure salvare un modello mentre si sta creando un trasferimento file selezionando la casella di spunta **Salva impostazioni di trasferimento come modello** . La finestra **Template di trasferimento** visualizza tutti i template di trasferimento che sono stati creati nella rete Managed File Transfer .

Riferimenti correlati

fteCreateTemplate: crea un nuovo modello di trasferimento file

Modelli fteList

Modelli fteDelete

Backup di una definizione della maschera di trasferimento file

I modelli di trasferimento file contengono un documento XML che definisce le specifiche file di origine e di destinazione per il trasferimento. È possibile utilizzare questo file XML come input per il comando **fteCreateTemplate** per creare nuovamente un modello di trasferimento file.

Informazioni su questa attività

Per eseguire il backup del documento XML contenente le specifiche file di origine e di destinazione per un modello di trasferimento, utilizzare il comando fteCreateTransfer command o IBM MQ Explorer. Per creare un file di backup formattato XML del modello di trasferimento, utilizzare la seguente procedura:

Procedura

- Metodo uno: utilizzare il parametro **-gt** su un comando fteCreateTransfer per generare un messaggio XML del template di trasferimento in un nuovo file.
- Metodo due: creare il modello utilizzando IBM MQ Explorer.

  Quando si arriva alla pagina *Riepilogo modello di trasferimento* :

  a) Copiare l' *Anteprima XML del messaggio di richiesta*.

  b) Salvare questo messaggio XML modello di trasferimento in un nuovo file.
- Metodo tre: utilizzare IBM MQ Explorer per eseguire il backup dei modelli esistenti.

  a) Andare a **Managed File Transfer** > **Nome gestore code** > **Template di trasferimento**.

  b) Nel pannello Trasferisci, evidenziare il modello di cui è necessario eseguire il backup, fare clic con il pulsante destro del mouse e selezionare **Modifica** dal menu a comparsa.

  c) Fare clic su **Avanti** fino a quando non si arriva alla pagina *Riepilogo modello di trasferimento* .

  d) Copiare l' *Anteprima XML del messaggio di richiesta*.

  e) Salvare questo messaggio XML modello di trasferimento in un nuovo file.

Risultati

È possibile utilizzare il file di messaggi XML del modello di trasferimento, creato da uno dei metodi precedenti, come input per il comando fteCreateTemplate . Fare riferimento al comando **fteCreateTemplate** per i dettagli su come utilizzare questo comando.

Riferimenti correlati

Comando fteCreateTemplate

comando fteListtTmplates

Trasferimento dei dati dai file ai messaggi

È possibile utilizzare la funzione file - to - message di Managed File Transfer per trasferire i dati da un file a un singolo messaggio o a più messaggi su una coda IBM MQ .

Per informazioni sui trasferimenti da messaggio a file, consultare “Trasferimento dei dati dai messaggi ai file” a pagina 280.

L'agent di destinazione per un trasferimento file - to - message non può essere un agent bridge di protocollo o un agent bridge Connect:Direct .

È possibile trasferire i dati del file ai dati del messaggio IBM MQ . I messaggi IBM MQ possono essere letti e utilizzati dalle applicazioni. Sono supportati i seguenti tipi di trasferimento file - to - message:

- Da un singolo file ad un singolo messaggio. Il messaggio non dispone di un ID gruppo IBM MQ impostato.
- Da un singolo file a più messaggi, suddividendo il file in messaggi di una data lunghezza. I messaggi hanno tutti lo stesso ID gruppo IBM MQ .
- Da un singolo file a più messaggi, suddividendo un file di testo in un delimitatore di espressione regolare Java . I messaggi hanno tutti lo stesso ID gruppo IBM MQ .
- Da un singolo file a più messaggi, suddividendo un file binario con un delimitatore esadecimale. I messaggi hanno tutti lo stesso ID gruppo IBM MQ .

  Se si desidera suddividere un file binario utilizzando una sequenza di byte come delimitatore, utilizzare il parametro **-sqdb** del comando **fteCreateTransfer** . Per ulteriori informazioni, consultare parametro -sqdb.

Per impostazione predefinita, i messaggi creati da un trasferimento da file a messaggio sono persistenti. I messaggi possono essere impostati per essere non persistenti o per avere il valore di persistenza definito dalla coda di destinazione.

Se si specifica che un file è suddiviso in più messaggi, tutti i messaggi creati dal file hanno lo stesso ID gruppo IBM MQ . Se non si specifica che un file è suddiviso in più messaggi, solo un messaggio viene creato dal file e questo messaggio non ha l'ID gruppo IBM MQ impostato.

Se si stanno trasferendo file a messaggi di grandi dimensioni o a molti messaggi di piccole dimensioni, potrebbe essere necessario modificare alcune proprietà IBM MQ o Managed File Transfer . Per informazioni, consultare Guida all'impostazione degli attributi MQ e delle proprietà MFT associate alla dimensione del messaggio.

Nota: Se la coda di destinazione è una coda con cluster o un alias per una coda con cluster, si riceve un messaggio di errore durante il trasferimento di un file in una coda se l'output della proprietà dell'agent enableClusterQueueInputnon è stato impostato su true. Per ulteriori informazioni, fare riferimento a Operazioni da eseguire se la coda di destinazione è una coda con cluster o un alias per una coda con cluster

Attività correlate

“Configurazione di un agent per eseguire trasferimenti da file a messaggi” a pagina 266
Per impostazione predefinita, gli agent non possono eseguire trasferimenti da file a messaggio o da messaggio a file. Per abilitare questa funzione, è necessario impostare la proprietà dell'agent enableQueueInputOutput su true. Per abilitare la scrittura nelle code con cluster IBM MQ , è necessario impostare anche la proprietà dell'agente enableClusterQueueInputOutput su true.

“Esempio: trasferimento di un singolo file in un singolo messaggio” a pagina 268
È possibile specificare una coda come destinazione di trasferimento file utilizzando il parametro **-dq** con il comando **fteCreateTransfer** . Il file origine deve essere inferiore alla lunghezza massima del messaggio impostata sulla coda di destinazione. La coda di destinazione non deve trovarsi sullo stesso gestore code a cui si connette l'agent di destinazione, ma questi due gestori code devono essere in grado di comunicare.

“Esempio: suddivisione di un singolo file in più messaggi per lunghezza” a pagina 269

È possibile suddividere un file in più messaggi IBM MQ utilizzando il parametro **-qs** del comando **fteCreateTransfer** . Il file è suddiviso in sezioni a lunghezza fissa, ognuna delle quali viene scritta in un messaggio singolo.

“Esempio: divisione di un file di testo con un delimitatore di espressione regolare e inclusione del delimitatore nei messaggi” a pagina 272
Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java e includere la corrispondenza dell'espressione regolare nei messaggi risultanti. A tale scopo, utilizzare i parametri **-dqdt** e **-qi** del comando **fteCreateTransfer** .

“Esempio: suddivisione di un file di testo in più messaggi utilizzando un delimitatore di espressione regolare” a pagina 271
Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java . A tale scopo, utilizzare il parametro **-dqdt** del comando **fteCreateTransfer** .

“Esempio: impostazione delle proprietà del messaggio IBM MQ su un trasferimento file - a - messaggio” a pagina 275
È possibile utilizzare il parametro **-qmp** nel comando **fteCreateTransfer** per specificare se le proprietà del messaggio IBM MQ sono impostate sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ consentono all'applicazione di selezionare i messaggi da elaborare o di richiamare le informazioni su un messaggio senza accedere alle intestazioni IBM MQ Message Descriptor (MQMD) o MQRFH2 .

“Esempio: impostazione di proprietà definite dall'utente su un trasferimento file - a - messaggio” a pagina 276
I metadati definiti dall'utente sono impostati come una proprietà del messaggio IBM MQ sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ abilitano un'applicazione a selezionare i messaggi da elaborare o a recuperare informazioni su un messaggio senza accedere alle intestazioni IBM MQ MQMD (Message Descriptor) o MQRFH2 .

“Avvio di un nuovo trasferimento file” a pagina 218
È possibile avviare un nuovo trasferimento file da IBM MQ Explorer o dalla riga comandi ed è possibile scegliere di trasferire un singolo file o più file in un gruppo.

Riferimenti correlati

“Errore di trasferimento da file a messaggio” a pagina 279
Se un trasferimento file - a - messaggio ha esito negativo dopo che l'agent ha avviato la scrittura dei dati file nella coda di destinazione, l'agent scrive un messaggio nella coda per indicare a un'applicazione che utilizza i messaggi che si è verificato un errore.

Proprietà dei messaggi MQ impostate da MFT sui messaggi scritti nelle code di destinazione

Istruzioni per l'impostazione degli attributi MQ e delle proprietà MFT associate alla dimensione del messaggio

Configurazione di un agent per eseguire trasferimenti da file a messaggi

Per impostazione predefinita, gli agent non possono eseguire trasferimenti da file a messaggio o da messaggio a file. Per abilitare questa funzione, è necessario impostare la proprietà dell'agent enableQueueInputOutput su true. Per abilitare la scrittura nelle code con cluster IBM MQ , è necessario impostare anche la proprietà dell'agente enableClusterQueueInputOutput su true.

Informazioni su questa attività

Se si tenta di eseguire un trasferimento file - a - messaggio a un agent di destinazione che non ha la proprietà `enableQueueInputOutput` impostata su true, il trasferimento non riesce. Il messaggio del log di trasferimento pubblicato nel gestore code di coordinamento contiene il messaggio seguente:

```
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.
```

Per consentire all'agent di scrivere e leggere dalle code, effettuare le seguenti operazioni:

Procedura

1. Arrestare l'agente di destinazione utilizzando il comando **Agent fteStop** .
2. Modificare il file `agent.properties` per includere la riga `enableQueueInputOutput=true`.

   Il file `agent.properties` si trova nella directory *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*destination_agent_name*.
3. Opzionale: Modificare il file `agent.properties` per includere la riga `enableClusterQueueInputOutput=true`. Il file `agent.properties` si trova nella directory *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*destination_agent_name*.
4. Avviare l'agent di destinazione utilizzando il comando **fteStartAgent** .

Concetti correlati

“Trasferimento dei dati dai file ai messaggi” a pagina 265
È possibile utilizzare la funzione file - to - message di Managed File Transfer per trasferire i dati da un file a un singolo messaggio o a più messaggi su una coda IBM MQ .

Attività correlate

“Esempio: trasferimento di un singolo file in un singolo messaggio” a pagina 268
È possibile specificare una coda come destinazione di trasferimento file utilizzando il parametro **-dq** con il comando **fteCreateTransfer** . Il file origine deve essere inferiore alla lunghezza massima del messaggio impostata sulla coda di destinazione. La coda di destinazione non deve trovarsi sullo stesso gestore code a cui si connette l'agent di destinazione, ma questi due gestori code devono essere in grado di comunicare.

“Esempio: suddivisione di un singolo file in più messaggi per lunghezza” a pagina 269
È possibile suddividere un file in più messaggi IBM MQ utilizzando il parametro **-qs** del comando **fteCreateTransfer** . Il file è suddiviso in sezioni a lunghezza fissa, ognuna delle quali viene scritta in un messaggio singolo.

“Esempio: divisione di un file di testo con un delimitatore di espressione regolare e inclusione del delimitatore nei messaggi” a pagina 272
Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java e includere la corrispondenza dell'espressione regolare nei messaggi risultanti. A tale scopo, utilizzare i parametri **-dqdt** e **-qi** del comando **fteCreateTransfer** .

“Esempio: suddivisione di un file di testo in più messaggi utilizzando un delimitatore di espressione regolare” a pagina 271
Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java . A tale scopo, utilizzare il parametro **-dqdt** del comando **fteCreateTransfer** .

“Esempio: impostazione delle proprietà del messaggio IBM MQ su un trasferimento file - a - messaggio” a pagina 275
È possibile utilizzare il parametro **-qmp** nel comando **fteCreateTransfer** per specificare se le proprietà del messaggio IBM MQ sono impostate sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ consentono all'applicazione di selezionare i messaggi da elaborare o di richiamare le informazioni su un messaggio senza accedere alle intestazioni IBM MQ Message Descriptor (MQMD) o MQRFH2 .

“Esempio: impostazione di proprietà definite dall'utente su un trasferimento file - a - messaggio” a pagina 276
I metadati definiti dall'utente sono impostati come una proprietà del messaggio IBM MQ sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ abilitano un'applicazione a selezionare i messaggi da elaborare o a recuperare informazioni su un messaggio senza accedere alle intestazioni IBM MQ MQMD (Message Descriptor) o MQRFH2 .

Riferimenti correlati

fteStopAgent

fteStartAgent

Il file MFT agent.properties

“Errore di trasferimento da file a messaggio” a pagina 279
Se un trasferimento file - a - messaggio ha esito negativo dopo che l'agent ha avviato la scrittura dei dati file nella coda di destinazione, l'agent scrive un messaggio nella coda per indicare a un'applicazione che utilizza i messaggi che si è verificato un errore.

Esempio: trasferimento di un singolo file in un singolo messaggio

È possibile specificare una coda come destinazione di trasferimento file utilizzando il parametro **-dq** con il comando **fteCreateTransfer** . Il file origine deve essere inferiore alla lunghezza massima del messaggio impostata sulla coda di destinazione. La coda di destinazione non deve trovarsi sullo stesso gestore code a cui si connette l'agent di destinazione, ma questi due gestori code devono essere in grado di comunicare.

Informazioni su questa attività

Il file di origine è denominato /tmp/single_record.txt e si trova sullo stesso sistema dell'agent origine, AGENT_NEPTUNE. L'agent di origine, AGENT_NEPTUNE, utilizza il gestore code QM_NEPTUNE. L'agent di destinazione è AGENT_VENUS e questo agent si connette al gestore code QM_VENUS. La coda di destinazione, RICEVIING_QUEUE, si trova sul gestore code QM_MERCURY. QM_MERCURY si trova nella stessa rete IBM MQ del gestore code QM_VENUS a cui è possibile accedere.

Procedura

Immettere il seguente comando:

```
fteCreateTransfer -sa AGENT_NEPTUNE -sm QM_NEPTUNE -da AGENT_VENUS -dm QM_VENUS
                  -dq RECEIVING_QUEUE@QM_MERCURY /tmp/single_record.txt
```

Se la coda di destinazione si trova su un gestore code diverso da quello utilizzato dall'agente di destinazione, è necessario specificare il valore del parametro **-dq** nel seguente formato *nome_coda*@*nome_gestore_code*. Se non si specifica @*queue_manager_name* nel valore, l'agente di destinazione presuppone che la coda di destinazione si trovi sul gestore code dell'agente di destinazione. L'eccezione si verifica quando la proprietà dell'agent di output enableClusterQueueInputè stata impostata su true. In questo caso, l'agent di destinazione utilizzerà le procedure di risoluzione IBM MQ standard per determinare dove si trova la coda.

L'agent di origine, AGENT_NEPTUNE, legge i dati dal file /tmp/single_record.txt e li trasferisce all'agent di destinazione, AGENT_VENUS. L'agent di destinazione, AGENT_VENUS, invia i dati a un messaggio persistente sulla coda RICEVIING_QUEUE@QM_MERCURY. Il messaggio non dispone di un ID gruppo IBM MQ impostato.

Concetti correlati

“Trasferimento dei dati dai file ai messaggi” a pagina 265
È possibile utilizzare la funzione file - to - message di Managed File Transfer per trasferire i dati da un file a un singolo messaggio o a più messaggi su una coda IBM MQ .

Attività correlate

“Configurazione di un agent per eseguire trasferimenti da file a messaggi” a pagina 266
Per impostazione predefinita, gli agent non possono eseguire trasferimenti da file a messaggio o da messaggio a file. Per abilitare questa funzione, è necessario impostare la proprietà dell'agent enableQueueInputOutput su true. Per abilitare la scrittura nelle code con cluster IBM MQ , è necessario impostare anche la proprietà dell'agente enableClusterQueueInputOutput su true.

“Esempio: suddivisione di un singolo file in più messaggi per lunghezza” a pagina 269
È possibile suddividere un file in più messaggi IBM MQ utilizzando il parametro **-qs** del comando **fteCreateTransfer** . Il file è suddiviso in sezioni a lunghezza fissa, ognuna delle quali viene scritta in un messaggio singolo.

“Esempio: divisione di un file di testo con un delimitatore di espressione regolare e inclusione del delimitatore nei messaggi” a pagina 272
Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java e includere la corrispondenza dell'espressione regolare nei messaggi risultanti. A tale scopo, utilizzare i parametri **-dqdt** e **-qi** del comando **fteCreateTransfer** .

“Esempio: suddivisione di un file di testo in più messaggi utilizzando un delimitatore di espressione regolare” a pagina 271
Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java . A tale scopo, utilizzare il parametro **-dqdt** del comando **fteCreateTransfer** .

“Esempio: impostazione delle proprietà del messaggio IBM MQ su un trasferimento file - a - messaggio” a pagina 275
È possibile utilizzare il parametro **-qmp** nel comando **fteCreateTransfer** per specificare se le proprietà del messaggio IBM MQ sono impostate sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ consentono all'applicazione di selezionare i messaggi da elaborare o di richiamare le informazioni su un messaggio senza accedere alle intestazioni IBM MQ Message Descriptor (MQMD) o MQRFH2 .

“Esempio: impostazione di proprietà definite dall'utente su un trasferimento file - a - messaggio” a pagina 276
I metadati definiti dall'utente sono impostati come una proprietà del messaggio IBM MQ sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ abilitano un'applicazione a selezionare i messaggi da elaborare o a recuperare informazioni su un messaggio senza accedere alle intestazioni IBM MQ MQMD (Message Descriptor) o MQRFH2 .

“Avvio di un nuovo trasferimento file” a pagina 218
È possibile avviare un nuovo trasferimento file da IBM MQ Explorer o dalla riga comandi ed è possibile scegliere di trasferire un singolo file o più file in un gruppo.

Riferimenti correlati

“Errore di trasferimento da file a messaggio” a pagina 279
Se un trasferimento file - a - messaggio ha esito negativo dopo che l'agent ha avviato la scrittura dei dati file nella coda di destinazione, l'agent scrive un messaggio nella coda per indicare a un'applicazione che utilizza i messaggi che si è verificato un errore.

Esempio: suddivisione di un singolo file in più messaggi per lunghezza

È possibile suddividere un file in più messaggi IBM MQ utilizzando il parametro **-qs** del comando **fteCreateTransfer** . Il file è suddiviso in sezioni a lunghezza fissa, ognuna delle quali viene scritta in un messaggio singolo.

Informazioni su questa attività

Il file di origine è denominato `/tmp/source.file` e ha una dimensione di 36 KB. Il file di origine si trova sullo stesso sistema dell'agent di origine AGENT_NEPTUNE. L'agent di origine, AGENT_NEPTUNE, si connette al gestore code QM_NEPTUNE. L'agent di destinazione è AGENT_MERCURY, che si connette al gestore code QM_MERCURY. La coda di destinazione, RICEVING_QUEUE, si trova anche sul gestore code QM_MERCURY. Il trasferimento suddivide il file di origine in sezioni con una dimensione di 1 KB e scrive ciascuna di queste sezioni in un messaggio su RICEVIING_QUEUE.

Procedura

Immettere il seguente comando:

```
fteCreateTransfer -sa AGENT_NEPTUNE -sm QM_NEPTUNE -da AGENT_MERCURY -dm QM_MERCURY
                  -dq RECEIVING_QUEUE -qs 1K /tmp/source.file
```

L'agent di origine, AGENT_NEPTUNE, legge i dati dal file `/tmp/source.file` e li trasferisce all'agent di destinazione, AGENT_MERCURY. L'agent di destinazione, AGENT_MERCURY, scrive i dati in 36 messaggi

persistenti da 1 KB sulla coda RICEVIING_QUEUE@QM_MERCURY. Questi messaggi hanno tutti lo stesso IBM MQ ID gruppo e l'ultimo messaggio nel gruppo ha l'indicatore IBM MQ LAST_MSG_IN_GROUP impostato.

Concetti correlati

“Trasferimento dei dati dai file ai messaggi” a pagina 265
È possibile utilizzare la funzione file - to - message di Managed File Transfer per trasferire i dati da un file a un singolo messaggio o a più messaggi su una coda IBM MQ .

Attività correlate

“Configurazione di un agent per eseguire trasferimenti da file a messaggi” a pagina 266
Per impostazione predefinita, gli agent non possono eseguire trasferimenti da file a messaggio o da messaggio a file. Per abilitare questa funzione, è necessario impostare la proprietà dell'agent enableQueueInputOutput su true. Per abilitare la scrittura nelle code con cluster IBM MQ , è necessario impostare anche la proprietà dell'agente enableClusterQueueInputOutput su true.

“Esempio: trasferimento di un singolo file in un singolo messaggio” a pagina 268
È possibile specificare una coda come destinazione di trasferimento file utilizzando il parametro **-dq** con il comando **fteCreateTransfer** . Il file origine deve essere inferiore alla lunghezza massima del messaggio impostata sulla coda di destinazione. La coda di destinazione non deve trovarsi sullo stesso gestore code a cui si connette l'agent di destinazione, ma questi due gestori code devono essere in grado di comunicare.

“Esempio: divisione di un file di testo con un delimitatore di espressione regolare e inclusione del delimitatore nei messaggi” a pagina 272
Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java e includere la corrispondenza dell'espressione regolare nei messaggi risultanti. A tale scopo, utilizzare i parametri **-dqdt** e **-qi** del comando **fteCreateTransfer** .

“Esempio: suddivisione di un file di testo in più messaggi utilizzando un delimitatore di espressione regolare” a pagina 271
Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java . A tale scopo, utilizzare il parametro **-dqdt** del comando **fteCreateTransfer** .

“Esempio: impostazione delle proprietà del messaggio IBM MQ su un trasferimento file - a - messaggio” a pagina 275
È possibile utilizzare il parametro **-qmp** nel comando **fteCreateTransfer** per specificare se le proprietà del messaggio IBM MQ sono impostate sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ consentono all'applicazione di selezionare i messaggi da elaborare o di richiamare le informazioni su un messaggio senza accedere alle intestazioni IBM MQ Message Descriptor (MQMD) o MQRFH2 .

“Esempio: impostazione di proprietà definite dall'utente su un trasferimento file - a - messaggio” a pagina 276
I metadati definiti dall'utente sono impostati come una proprietà del messaggio IBM MQ sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ abilitano un'applicazione a selezionare i messaggi da elaborare o a recuperare informazioni su un messaggio senza accedere alle intestazioni IBM MQ MQMD (Message Descriptor) o MQRFH2 .

“Avvio di un nuovo trasferimento file” a pagina 218
È possibile avviare un nuovo trasferimento file da IBM MQ Explorer o dalla riga comandi ed è possibile scegliere di trasferire un singolo file o più file in un gruppo.

Riferimenti correlati

“Errore di trasferimento da file a messaggio” a pagina 279
Se un trasferimento file - a - messaggio ha esito negativo dopo che l'agent ha avviato la scrittura dei dati file nella coda di destinazione, l'agent scrive un messaggio nella coda per indicare a un'applicazione che utilizza i messaggi che si è verificato un errore.

Esempio: suddivisione di un file di testo in più messaggi utilizzando un delimitatore di espressione regolare

Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java . A tale scopo, utilizzare il parametro **-dqdt** del comando **fteCreateTransfer** .

Informazioni su questa attività

Il file viene suddiviso in sezioni a lunghezza variabile, ognuna delle quali viene scritta in un singolo messaggio. Il file di testo viene suddiviso in ogni punto in cui il testo nel file corrisponde a una determinata espressione regolare. Il file di origine è denominato `/tmp/names.text` e ha il seguente contenuto:

```
Jenny Jones,John Smith,Jane Brown
```

L'espressione regolare che specifica dove suddividere il file è il carattere virgola (`,`).

Il file di origine si trova sullo stesso sistema dell'agent di origine *AGENT_NEPTUNE*, che si connette al gestore code QM_NEPTUNE. La coda di destinazione, *RICEVIING_QUEUE*, si trova sul gestore code *QM_MERCURY*. *QM_MERCURY* è anche il gestore code utilizzato dall'agent di destinazione *AGENT_MERCURY*. Il trasferimento suddivide il file di origine in sezioni e scrive ognuna di queste sezioni in un messaggio su *RICEVERE la coda*.

Procedura

Immettere il seguente comando:

```
fteCreateTransfer -sa AGENT_NEPTUNE -sm QM_NEPTUNE -da AGENT_MERCURY -dm QM_MERCURY
                  -dq RECEIVING_QUEUE -t text -dqdp postfix -dqdt "," /tmp/names.text
```

L'agent di origine, AGENT_NEPTUNE, legge i dati dal file `/tmp/names.text` e li trasferisce all'agent di destinazione, AGENT_MERCURY. L'agent di destinazione, AGENT_MERCURY, scrive i dati in tre messaggi persistenti sulla coda *RICEVERE la coda*. Questi messaggi hanno tutti lo stesso IBM MQ ID gruppo e l'ultimo messaggio nel gruppo ha l'indicatore IBM MQ LAST_MSG_IN_GROUP impostato.

I dati nei messaggi sono i seguenti.

- Primo messaggio:

  ```
  Jenny Jones
  ```

- Secondo messaggio:

  ```
  John Smith
  ```

- Terzo messaggio:

  ```
  Jane Brown
  ```

Concetti correlati

“Trasferimento dei dati dai file ai messaggi” a pagina 265
È possibile utilizzare la funzione file - to - message di Managed File Transfer per trasferire i dati da un file a un singolo messaggio o a più messaggi su una coda IBM MQ .

Attività correlate

“Configurazione di un agent per eseguire trasferimenti da file a messaggi” a pagina 266

Per impostazione predefinita, gli agent non possono eseguire trasferimenti da file a messaggio o da messaggio a file. Per abilitare questa funzione, è necessario impostare la proprietà dell'agent enableQueueInputOutput su true. Per abilitare la scrittura nelle code con cluster IBM MQ , è necessario impostare anche la proprietà dell'agente enableClusterQueueInputOutput su true.

“Esempio: trasferimento di un singolo file in un singolo messaggio” a pagina 268
È possibile specificare una coda come destinazione di trasferimento file utilizzando il parametro **-dq** con il comando **fteCreateTransfer** . Il file origine deve essere inferiore alla lunghezza massima del messaggio impostata sulla coda di destinazione. La coda di destinazione non deve trovarsi sullo stesso gestore code a cui si connette l'agent di destinazione, ma questi due gestori code devono essere in grado di comunicare.

“Esempio: suddivisione di un singolo file in più messaggi per lunghezza” a pagina 269
È possibile suddividere un file in più messaggi IBM MQ utilizzando il parametro **-qs** del comando **fteCreateTransfer** . Il file è suddiviso in sezioni a lunghezza fissa, ognuna delle quali viene scritta in un messaggio singolo.

“Esempio: divisione di un file di testo con un delimitatore di espressione regolare e inclusione del delimitatore nei messaggi” a pagina 272
Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java e includere la corrispondenza dell'espressione regolare nei messaggi risultanti. A tale scopo, utilizzare i parametri **-dqdt** e **-qi** del comando **fteCreateTransfer** .

“Esempio: impostazione delle proprietà del messaggio IBM MQ su un trasferimento file - a - messaggio” a pagina 275
È possibile utilizzare il parametro **-qmp** nel comando **fteCreateTransfer** per specificare se le proprietà del messaggio IBM MQ sono impostate sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ consentono all'applicazione di selezionare i messaggi da elaborare o di richiamare le informazioni su un messaggio senza accedere alle intestazioni IBM MQ Message Descriptor (MQMD) o MQRFH2 .

“Esempio: impostazione di proprietà definite dall'utente su un trasferimento file - a - messaggio” a pagina 276
I metadati definiti dall'utente sono impostati come una proprietà del messaggio IBM MQ sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ abilitano un'applicazione a selezionare i messaggi da elaborare o a recuperare informazioni su un messaggio senza accedere alle intestazioni IBM MQ MQMD (Message Descriptor) o MQRFH2 .

“Avvio di un nuovo trasferimento file” a pagina 218
È possibile avviare un nuovo trasferimento file da IBM MQ Explorer o dalla riga comandi ed è possibile scegliere di trasferire un singolo file o più file in un gruppo.

Riferimenti correlati

“Errore di trasferimento da file a messaggio” a pagina 279
Se un trasferimento file - a - messaggio ha esito negativo dopo che l'agent ha avviato la scrittura dei dati file nella coda di destinazione, l'agent scrive un messaggio nella coda per indicare a un'applicazione che utilizza i messaggi che si è verificato un errore.

Espressioni regolari utilizzate da MFT

Esempio: divisione di un file di testo con un delimitatore di espressione regolare e inclusione del delimitatore nei messaggi

Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java e includere la corrispondenza dell'espressione regolare nei messaggi risultanti. A tale scopo, utilizzare i parametri **-dqdt** e **-qi** del comando **fteCreateTransfer** .

Informazioni su questa attività

Trasferire un singolo file di testo a più messaggi su una coda. Il file viene suddiviso in sezioni a lunghezza variabile, ognuna delle quali viene scritta in un singolo messaggio. Il file di testo viene suddiviso in

ogni punto in cui il testo nel file corrisponde a una determinata espressione regolare. Il file di origine è denominato `/tmp/customers.text` e ha il seguente contenuto:

```
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
```

L'espressione regolare che specifica dove dividere il file è `Customer\snumber:\s\d+`, che corrisponde al testo "Numero cliente: " seguito da qualsiasi numero di cifre. Le espressioni regolari specificate nella riga comandi devono essere racchiuse tra virgolette per evitare che la shell di comandi valuti l'espressione regolare. L'espressione regolare viene valutata come espressione regolare Java . Per ulteriori informazioni, consultare Espressioni regolari utilizzate da MFT.

Per impostazione predefinita, il numero di caratteri che un'espressione regolare può corrispondere è impostato su cinque. L'espressione regolare utilizzata in questo esempio corrisponde a stringhe più lunghe di cinque caratteri. Per abilitare le corrispondenze più lunghe di cinque caratteri, modificare il file delle proprietà dell'agent per includere la proprietà **maxDelimiterMatchLength**.

Per default, il testo che corrisponde all'espressione regolare non viene incluso nei messaggi. Per inserire il testo che corrisponde all'espressione regolare nei messaggi, come in questo esempio, utilizzare il parametro **-qi** . Il file di origine si trova sullo stesso sistema dell'agent di origine AGENT_NEPTUNE, che si connette al gestore code QM_NEPTUNE. La coda di destinazione, RICEVIING_QUEUE, si trova sul gestore code QM_MERCURY. QM_MERCURY è anche il gestore code utilizzato dall'agent di destinazione AGENT_MERCURY. Il trasferimento divide il file di origine in sezioni e scrive ognuna di queste sezioni in un messaggio su RICEVIING_QUEUE.

Procedura

1. Arrestare l'agent di destinazione utilizzando il seguente comando:

   ```
   fteStopAgent AGENT_MERCURY
   ```

2. Aggiungere la seguente riga al file delle proprietà agent per AGENT_MERCURY:

   ```
   maxDelimiterMatchLength=25
   ```

   Nota: L'aumento del valore di **maxDelimiterMatchLength** può ridurre le prestazioni.

3. Avviare l'agente di destinazione utilizzando il seguente comando:

   ```
   fteStartAgent AGENT_MERCURY
   ```

4. Immettere il seguente comando:

   ```
   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
   ```

   L'agent di origine, AGENT_NEPTUNE, legge i dati dal file `/tmp/customers.text` e li trasferisce all'agent di destinazione, AGENT_MERCURY. L'agent di destinazione, AGENT_MERCURY, scrive i dati in tre messaggi persistenti sulla coda RICEVIING_QUEUE. Questi messaggi hanno tutti lo stesso IBM MQ ID gruppo e l'ultimo messaggio nel gruppo ha l'indicatore IBM MQ LAST_MSG_IN_GROUP impostato.

   I dati nei messaggi sono i seguenti.

   - Primo messaggio:

```
Customer name: John Smith
Customer contact details: john@example.net
Customer number: 314
```

- Secondo messaggio:

```
Customer name: Jane Brown
Customer contact details: jane@example.com
Customer number: 42
```

- Terzo messaggio:

```
Customer name: James Jones
Customer contact details: jjones@example.net
Customer number: 26
```

Concetti correlati

“Trasferimento dei dati dai file ai messaggi” a pagina 265
È possibile utilizzare la funzione file - to - message di Managed File Transfer per trasferire i dati da un file a un singolo messaggio o a più messaggi su una coda IBM MQ .

Attività correlate

“Configurazione di un agent per eseguire trasferimenti da file a messaggi” a pagina 266
Per impostazione predefinita, gli agent non possono eseguire trasferimenti da file a messaggio o da messaggio a file. Per abilitare questa funzione, è necessario impostare la proprietà dell'agent enableQueueInputOutput su true. Per abilitare la scrittura nelle code con cluster IBM MQ , è necessario impostare anche la proprietà dell'agente enableClusterQueueInputOutput su true.

“Esempio: trasferimento di un singolo file in un singolo messaggio” a pagina 268
È possibile specificare una coda come destinazione di trasferimento file utilizzando il parametro **-dq** con il comando **fteCreateTransfer** . Il file origine deve essere inferiore alla lunghezza massima del messaggio impostata sulla coda di destinazione. La coda di destinazione non deve trovarsi sullo stesso gestore code a cui si connette l'agent di destinazione, ma questi due gestori code devono essere in grado di comunicare.

“Esempio: suddivisione di un singolo file in più messaggi per lunghezza” a pagina 269
È possibile suddividere un file in più messaggi IBM MQ utilizzando il parametro **-qs** del comando **fteCreateTransfer** . Il file è suddiviso in sezioni a lunghezza fissa, ognuna delle quali viene scritta in un messaggio singolo.

“Esempio: suddivisione di un file di testo in più messaggi utilizzando un delimitatore di espressione regolare” a pagina 271
Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java . A tale scopo, utilizzare il parametro **-dqdt** del comando **fteCreateTransfer** .

“Esempio: impostazione delle proprietà del messaggio IBM MQ su un trasferimento file - a - messaggio” a pagina 275
È possibile utilizzare il parametro **-qmp** nel comando **fteCreateTransfer** per specificare se le proprietà del messaggio IBM MQ sono impostate sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ consentono all'applicazione di selezionare i messaggi da elaborare o di richiamare le informazioni su un messaggio senza accedere alle intestazioni IBM MQ Message Descriptor (MQMD) o MQRFH2 .

“Esempio: impostazione di proprietà definite dall'utente su un trasferimento file - a - messaggio” a pagina 276
I metadati definiti dall'utente sono impostati come una proprietà del messaggio IBM MQ sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ abilitano un'applicazione a selezionare i messaggi da elaborare o a recuperare informazioni su un messaggio senza accedere alle intestazioni IBM MQ MQMD (Message Descriptor) o MQRFH2 .

“Avvio di un nuovo trasferimento file” a pagina 218

È possibile avviare un nuovo trasferimento file da IBM MQ Explorer o dalla riga comandi ed è possibile scegliere di trasferire un singolo file o più file in un gruppo.

Riferimenti correlati

Il file MFT `agent.properties`

Espressioni regolari utilizzate da MFT

Esempio: impostazione delle proprietà del messaggio IBM MQ su un trasferimento file - a - messaggio

È possibile utilizzare il parametro **-qmp** nel comando **fteCreateTransfer** per specificare se le proprietà del messaggio IBM MQ sono impostate sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ consentono all'applicazione di selezionare i messaggi da elaborare o di richiamare le informazioni su un messaggio senza accedere alle intestazioni IBM MQ Message Descriptor (MQMD) o MQRFH2 .

Informazioni su questa attività

Includere il parametro `-qmp true` nel comando **fteCreateTransfer** . In questo esempio, l'ID utente MQMD dell'utente che inoltra il comando è `larmer`.

Procedura

Immettere il seguente comando:

```
fteCreateTransfer -sa AGENT_JUPITER -da AGENT_SATURN -dq MY_QUEUE@MyQM -qmp true
      -t text /tmp/source_file.txt
```

Le proprietà IBM MQ del primo messaggio scritto dall'agent di destinazione, `AGENT_SATURN`, nella coda, `MY_QUEUE`, nel gestore code, `MyQM`, vengono impostate sui seguenti valori:

```
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
```

Concetti correlati

“Trasferimento dei dati dai file ai messaggi” a pagina 265

È possibile utilizzare la funzione file - to - message di Managed File Transfer per trasferire i dati da un file a un singolo messaggio o a più messaggi su una coda IBM MQ .

Attività correlate

“Configurazione di un agent per eseguire trasferimenti da file a messaggi” a pagina 266

Per impostazione predefinita, gli agent non possono eseguire trasferimenti da file a messaggio o da messaggio a file. Per abilitare questa funzione, è necessario impostare la proprietà dell'agent enableQueueInputOutput su true. Per abilitare la scrittura nelle code con cluster IBM MQ , è necessario impostare anche la proprietà dell'agente enableClusterQueueInputOutput su true.

“Esempio: trasferimento di un singolo file in un singolo messaggio” a pagina 268

È possibile specificare una coda come destinazione di trasferimento file utilizzando il parametro **-dq** con il comando **fteCreateTransfer** . Il file origine deve essere inferiore alla lunghezza massima del messaggio impostata sulla coda di destinazione. La coda di destinazione non deve trovarsi sullo stesso gestore code a cui si connette l'agent di destinazione, ma questi due gestori code devono essere in grado di comunicare.

“Esempio: suddivisione di un singolo file in più messaggi per lunghezza” a pagina 269

È possibile suddividere un file in più messaggi IBM MQ utilizzando il parametro **-qs** del comando **fteCreateTransfer** . Il file è suddiviso in sezioni a lunghezza fissa, ognuna delle quali viene scritta in un messaggio singolo.

“Esempio: divisione di un file di testo con un delimitatore di espressione regolare e inclusione del delimitatore nei messaggi” a pagina 272
Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java e includere la corrispondenza dell'espressione regolare nei messaggi risultanti. A tale scopo, utilizzare i parametri **-dqdt** e **-qi** del comando **fteCreateTransfer** .

“Esempio: suddivisione di un file di testo in più messaggi utilizzando un delimitatore di espressione regolare” a pagina 271
Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java . A tale scopo, utilizzare il parametro **-dqdt** del comando **fteCreateTransfer** .

“Esempio: impostazione di proprietà definite dall'utente su un trasferimento file - a - messaggio” a pagina 276
I metadati definiti dall'utente sono impostati come una proprietà del messaggio IBM MQ sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ abilitano un'applicazione a selezionare i messaggi da elaborare o a recuperare informazioni su un messaggio senza accedere alle intestazioni IBM MQ MQMD (Message Descriptor) o MQRFH2 .

“Avvio di un nuovo trasferimento file” a pagina 218
È possibile avviare un nuovo trasferimento file da IBM MQ Explorer o dalla riga comandi ed è possibile scegliere di trasferire un singolo file o più file in un gruppo.

Riferimenti correlati

“Errore di trasferimento da file a messaggio” a pagina 279
Se un trasferimento file - a - messaggio ha esito negativo dopo che l'agent ha avviato la scrittura dei dati file nella coda di destinazione, l'agent scrive un messaggio nella coda per indicare a un'applicazione che utilizza i messaggi che si è verificato un errore.

Proprietà dei messaggi MQ impostate da MFT sui messaggi scritti nelle code di destinazione

Esempio: impostazione di proprietà definite dall'utente su un trasferimento file - a - messaggio

I metadati definiti dall'utente sono impostati come una proprietà del messaggio IBM MQ sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ abilitano un'applicazione a selezionare i messaggi da elaborare o a recuperare informazioni su un messaggio senza accedere alle intestazioni IBM MQ MQMD (Message Descriptor) o MQRFH2 .

Informazioni su questa attività

Includere i parametri `-qmp true` e `-md account=123456` nel comando **fteCreateTransfer** , per impostare la proprietà `usr.account` su 123456 nell'intestazione RFH2 .

Procedura

Immettere il seguente comando:

```
fteCreateTransfer -sa AGENT_JUPITER -da AGENT_SATURN -dq MY_QUEUE@MyQM
                  -qmp true -md account=123456 /tmp/source_file.txt
```

Oltre alla serie standard di proprietà del messaggio IBM MQ , la proprietà definita dall'utente è impostata nell'intestazione del messaggio del primo messaggio scritto dall'agente di destinazione, `AGENT_SATURN`, nella coda, `MY_QUEUE`, sul gestore code, `MyQM`. L'intestazione è impostata sul valore seguente:

```
usr.account=123456
```

Il prefisso `usr` viene aggiunto all'inizio del nome dei metadati definiti dall'utente.

Concetti correlati

“Trasferimento dei dati dai file ai messaggi” a pagina 265
È possibile utilizzare la funzione file - to - message di Managed File Transfer per trasferire i dati da un file a un singolo messaggio o a più messaggi su una coda IBM MQ .

Attività correlate

“Configurazione di un agent per eseguire trasferimenti da file a messaggi” a pagina 266
Per impostazione predefinita, gli agent non possono eseguire trasferimenti da file a messaggio o da messaggio a file. Per abilitare questa funzione, è necessario impostare la proprietà dell'agent enableQueueInputOutput su true. Per abilitare la scrittura nelle code con cluster IBM MQ , è necessario impostare anche la proprietà dell'agente enableClusterQueueInputOutput su true.

“Esempio: trasferimento di un singolo file in un singolo messaggio” a pagina 268
È possibile specificare una coda come destinazione di trasferimento file utilizzando il parametro **-dq** con il comando **fteCreateTransfer** . Il file origine deve essere inferiore alla lunghezza massima del messaggio impostata sulla coda di destinazione. La coda di destinazione non deve trovarsi sullo stesso gestore code a cui si connette l'agent di destinazione, ma questi due gestori code devono essere in grado di comunicare.

“Esempio: suddivisione di un singolo file in più messaggi per lunghezza” a pagina 269
È possibile suddividere un file in più messaggi IBM MQ utilizzando il parametro **-qs** del comando **fteCreateTransfer** . Il file è suddiviso in sezioni a lunghezza fissa, ognuna delle quali viene scritta in un messaggio singolo.

“Esempio: divisione di un file di testo con un delimitatore di espressione regolare e inclusione del delimitatore nei messaggi” a pagina 272
Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java e includere la corrispondenza dell'espressione regolare nei messaggi risultanti. A tale scopo, utilizzare i parametri **-dqdt** e **-qi** del comando **fteCreateTransfer** .

“Esempio: suddivisione di un file di testo in più messaggi utilizzando un delimitatore di espressione regolare” a pagina 271
Trasferire un singolo file di testo a più messaggi suddividendo il file ad ogni corrispondenza di una determinata espressione regolare Java . A tale scopo, utilizzare il parametro **-dqdt** del comando **fteCreateTransfer** .

“Esempio: impostazione delle proprietà del messaggio IBM MQ su un trasferimento file - a - messaggio” a pagina 275
È possibile utilizzare il parametro **-qmp** nel comando **fteCreateTransfer** per specificare se le proprietà del messaggio IBM MQ sono impostate sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ consentono all'applicazione di selezionare i messaggi da elaborare o di richiamare le informazioni su un messaggio senza accedere alle intestazioni IBM MQ Message Descriptor (MQMD) o MQRFH2 .

“Avvio di un nuovo trasferimento file” a pagina 218
È possibile avviare un nuovo trasferimento file da IBM MQ Explorer o dalla riga comandi ed è possibile scegliere di trasferire un singolo file o più file in un gruppo.

Riferimenti correlati

Proprietà dei messaggi MQ impostate da MFT sui messaggi scritti nelle code di destinazione

Esempio: aggiunta di una proprietà del messaggio definita dall'utente per un trasferimento file - a - messaggio

Se si utilizza Managed File Transfer per i trasferimenti gestiti da messaggio a file, è possibile includere una proprietà del messaggio definita dall'utente per il messaggio risultante.

Informazioni su questa attività

È possibile utilizzare uno dei seguenti metodi per definire una proprietà del messaggio personalizzata:

- Specificare il parametro **-md** nella richiesta di trasferimento. Per ulteriori informazioni, fare riferimento a “Esempio: impostazione di proprietà definite dall'utente su un trasferimento file - a - messaggio” a pagina 276.
- Utilizzare un'attività Ant ; è possibile utilizzare fte: filecopy o fte:filemove. Il seguente esempio è un'attività 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>
```

- Utilizzare un controllo delle risorse e la sostituzione delle variabili. Il seguente esempio mostra alcune attività di trasferimento XML:

```
<?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>
```

Attività correlate

“Esempio: impostazione delle proprietà del messaggio IBM MQ su un trasferimento file - a - messaggio” a pagina 275

È possibile utilizzare il parametro **-qmp** nel comando **fteCreateTransfer** per specificare se le proprietà del messaggio IBM MQ sono impostate sul primo messaggio scritto nella coda di destinazione dal trasferimento. Le proprietà del messaggio IBM MQ consentono all'applicazione di selezionare i messaggi da elaborare o di richiamare le informazioni su un messaggio senza accedere alle intestazioni IBM MQ Message Descriptor (MQMD) o MQRFH2 .

Riferimenti correlati

attività fte: filecopy Ant

fte: filemove Ant attività

Errore di trasferimento da file a messaggio

Se un trasferimento file - a - messaggio ha esito negativo dopo che l'agent ha avviato la scrittura dei dati file nella coda di destinazione, l'agent scrive un messaggio nella coda per indicare a un'applicazione che utilizza i messaggi che si è verificato un errore.

Il messaggio scritto nella coda di destinazione se si verifica un errore:

- È vuoto
- Ha lo stesso ID gruppo IBM MQ del messaggio precedente scritto nella coda di destinazione dall'agente
- Ha l'indicatore IBM MQ LAST_MSG_IN_GROUP impostato
- Contiene ulteriori proprietà del messaggio IBM MQ , se le proprietà del messaggio sono abilitate. Per ulteriori informazioni, consultare l'argomento Proprietà dei messaggiMQ impostate da MFT sui messaggi scritti nelle code di destinazione.

Esempio

Un trasferimento viene richiesto eseguendo il seguente comando:

```
fteCreateTransfer -sa AGENT_JUPITER -da AGENT_SATURN -dq RECEIVING_QUEUE
                  -qmp true -qs 1K /tmp/source1.txt
```

Il file `source1.txt` è 48 KB. Il trasferimento divide questo file in messaggi da 1 KB e scrive tali messaggi nella coda di destinazione RICEVIING_QUEUE.

Mentre il trasferimento è in corso, dopo che l'agente ha scritto 16 messaggi in RICEVIING_QUEUE, si verifica un malfunzionamento sull'agente origine.

L'agent scrive un messaggio vuoto in RICEVIING_QUEUE. Oltre alla serie standard di proprietà del messaggio, il messaggio vuoto ha la seguente serie di proprietà del messaggio:

```
usr.WMQFTEResultCode = 40
usr.WMQFTESupplement = BFGTR0036I: The transfer failed to complete successfully.
```

Da IBM MQ 9.3.0, quando un trasferimento da un file non riesce, a causa di un errore di controllo della dimensione del delimitatore, viene inviato un solo messaggio vuoto. Inoltre, le proprietà utente vengono aggiunte a questo messaggio se l'errore di trasferimento è dovuto al fatto che il delimitatore supera la dimensione impostata sull'agente di destinazione.

Concetti correlati

“Trasferimento dei dati dai file ai messaggi” a pagina 265
È possibile utilizzare la funzione file - to - message di Managed File Transfer per trasferire i dati da un file a un singolo messaggio o a più messaggi su una coda IBM MQ .

Attività correlate

“Configurazione di un agent per eseguire trasferimenti da file a messaggi” a pagina 266
Per impostazione predefinita, gli agent non possono eseguire trasferimenti da file a messaggio o da messaggio a file. Per abilitare questa funzione, è necessario impostare la proprietà dell'agent enableQueueInputOutput su true. Per abilitare la scrittura nelle code con cluster IBM MQ , è necessario impostare anche la proprietà dell'agente enableClusterQueueInputOutput su true.

“Avvio di un nuovo trasferimento file” a pagina 218
È possibile avviare un nuovo trasferimento file da IBM MQ Explorer o dalla riga comandi ed è possibile scegliere di trasferire un singolo file o più file in un gruppo.

Riferimenti correlati

Il file MFT `agent.properties`

Proprietà dei messaggi MQ impostate da MFT sui messaggi scritti nelle code di destinazione

Trasferimento dei dati dai messaggi ai file

La funzione messaggio - a - file di Managed File Transfer consente di trasferire i dati da uno o più messaggi su una coda IBM MQ a un file, a un dataset (su z/OS) o a uno spazio file utente. Se si dispone di un'applicazione che crea o elabora messaggi IBM MQ , è possibile utilizzare la funzionalità messaggio - a - file di Managed File Transfer per trasferire questi messaggi a un file su qualsiasi sistema nella rete Managed File Transfer .

Per informazioni sui trasferimenti da file a messaggi, consultare “Trasferimento dei dati dai file ai messaggi” a pagina 265.

Attenzione: L'agent di origine per un trasferimento da messaggio a file non può essere un agent bridge di protocollo o un agent bridge Connect:Direct .

È possibile trasferire i dati del messaggio IBM MQ in un file. Sono supportati i seguenti tipi di trasferimento da messaggio a file:

- Da un singolo messaggio a un singolo file
- Da più messaggi a un singolo file
- Da più messaggi con lo stesso ID gruppo IBM MQ a un singolo file.
- Da più messaggi a un singolo file, incluso un delimitatore di testo o binario tra i dati di ciascun messaggio scritto nel file.

Se si stanno trasferendo file da messaggi di grandi dimensioni o da messaggi di piccole dimensioni, potrebbe essere necessario modificare alcune proprietà IBM MQ o Managed File Transfer . Per ulteriori informazioni, consultare Guida per l'impostazione degli attributi di MQ e delle proprietà MFT associate alla dimensione del messaggio.

In un trasferimento da messaggio a file, l'agent di origine sfoglia i messaggi dalla coda di origine, a differenza del GET distruttivo nelle versioni precedenti di IBM MQ. I messaggi vengono rimossi dalla coda di origine dopo che tutti i messaggi (in un gruppo se viene utilizzato il gruppo di messaggi) sono stati esaminati e i dati sono stati scritti nel file di destinazione. Ciò consente ai messaggi di rimanere nella coda di origine se un trasferimento non riesce o viene annullato. A causa di questa modifica, è necessario fornire anche l'autorizzazione a BROWSE insieme all'autorizzazione GET per eseguire i trasferimenti messaggio su file.

Managed File Transfer confronta l'ID trasferimento e il valore dell'attributo groupId all'interno del payload XML della richiesta di trasferimento. Se questi due identificatori sono equivalenti, l'agent di origine utilizza l'identificativo come opzione di corrispondenza dell'identificativo del messaggio (in contrapposizione ad un'opzione di corrispondenza dell'identificativo del gruppo) per il primo tentativo MQGET effettuato sulla coda di input per il trasferimento da messaggio a file.

Attività correlate

“Esempio: configurazione di una risorsa MFT” a pagina 241
È possibile specificare una coda IBM MQ come risorsa che deve essere monitorata da un monitoraggio risorse utilizzando il parametro **-mq** con il comando **fteCreateMonitor** .

Riferimenti correlati

Proprietà dei messaggi MQ lette da MFT dai messaggi sulle code origine

Istruzioni per l'impostazione degli attributi MQ e delle proprietà MFT associate alla dimensione del messaggio

Configurazione di un agent per eseguire trasferimenti da messaggio a file

Per impostazione predefinita, gli agent non possono eseguire trasferimenti da messaggio a file o da file a messaggio. Per abilitare questa funzione, è necessario impostare la proprietà dell'agente `enableQueueInputOutput` su true.

Informazioni su questa attività

Se si tenta di eseguire un messaggio per il trasferimento file da un agent di origine che non ha la proprietà `enableQueueInputOutput` impostata su true, il trasferimento non riesce. Il messaggio del log di trasferimento pubblicato nel gestore code di coordinamento contiene il messaggio seguente:

```
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.
```

Per consentire all'agent di scrivere e leggere dalle code, effettuare le seguenti operazioni:

Procedura

1. Arrestare l'agente di origine utilizzando il comando **fteStopAgent** .
2. Modificare il file `agent.properties` per includere la riga `enableQueueInputOutput=true`.

   Il file `agent.properties` si trova nella directory *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*source_agent_name*.
3. Avviare l'agente di origine utilizzando il comando **fteStartAgent** .

Concetti correlati

“Trasferimento dei dati dai messaggi ai file” a pagina 280
La funzione messaggio - a - file di Managed File Transfer consente di trasferire i dati da uno o più messaggi su una coda IBM MQ a un file, a un dataset (su z/OS) o a uno spazio file utente. Se si dispone di un'applicazione che crea o elabora messaggi IBM MQ , è possibile utilizzare la funzionalità messaggio - a - file di Managed File Transfer per trasferire questi messaggi a un file su qualsiasi sistema nella rete Managed File Transfer .

Attività correlate

“Esempio: trasferimento da una coda a un file singolo” a pagina 282

È possibile specificare una coda IBM MQ come origine di trasferimento file utilizzando il parametro **-sq** con il comando **fteCreateTransfer** .

“Esempio: trasferimento di un gruppo di messaggi da una coda a un unico file” a pagina 283
È possibile specificare un singolo gruppo completo su una coda IBM MQ come origine di un trasferimento file utilizzando i parametri **-sq** e **-sqgi** con il comando **fteCreateTransfer** .

“Esempio: inserimento di un delimitatore di testo prima dei dati da ciascun messaggio” a pagina 284
Quando si esegue il trasferimento in modalità testo da una coda di origine a un file, è possibile specificare che un delimitatore di testo venga inserito prima dei dati dei singoli messaggi utilizzando i parametri **-sq**, **-sqdt** e **-sqdp** con il comando **fteCreateTransfer** .

“Esempio: inserimento di un delimitatore binario dopo i dati da ciascun messaggio” a pagina 285
Quando si esegue il trasferimento in modo binario da una coda di origine a un file, è possibile specificare che venga inserito un delimitatore binario dopo i dati dei singoli messaggi utilizzando i parametri **-sq**, **-sqdb**e **-sqdp** con il comando **fteCreateTransfer** .

“Monitoraggio di una coda e utilizzo della sostituzione di variabili” a pagina 247
È possibile monitorare una coda e trasferire messaggi dalla coda monitorata a un file utilizzando il comando **fteCreateMonitor** . Il valore di qualsiasi proprietà del messaggio IBM MQ nel primo messaggio da leggere dalla coda monitorata può essere sostituito nella definizione XML dell'attività e utilizzato per definire il comportamento del trasferimento.

“Esempio: errore di trasferimento da messaggio a file utilizzando le proprietà del messaggio IBM MQ” a pagina 289
È possibile causare l'esito negativo del trasferimento file di un messaggio impostando la proprietà del messaggio `usr.UserReturnCode` IBM MQ su un valore diverso da zero. È inoltre possibile specificare ulteriori informazioni sul motivo dell'errore impostando la proprietà del messaggio `usr.UserSupplement` IBM MQ .

Riferimenti correlati

Il file MFT `agent.properties`

Esempio: trasferimento da una coda a un file singolo

È possibile specificare una coda IBM MQ come origine di trasferimento file utilizzando il parametro **-sq** con il comando **fteCreateTransfer** .

Informazioni su questa attività

I dati di origine sono contenuti in tre messaggi sulla coda START_QUEUE. Questa coda deve essere sul gestore code dell'agente di origine, *QM_NEPTUNE*.

Procedura

Immettere il seguente comando:

```
fteCreateTransfer -sa AGENT_NEPTUNE -sm QM_NEPTUNE
                  -da AGENT_VENUS -df /out/three_to_one.txt
                  -sq START_QUEUE
```

I dati nei messaggi sulla coda *START_QUEUE* vengono scritti nel file `/out/three_to_one.txt` sul sistema su cui è in esecuzione AGENT_VENUS .

Concetti correlati

“Trasferimento dei dati dai messaggi ai file” a pagina 280
La funzione messaggio - a - file di Managed File Transfer consente di trasferire i dati da uno o più messaggi su una coda IBM MQ a un file, a un dataset (su z/OS) o a uno spazio file utente. Se si dispone di un'applicazione che crea o elabora messaggi IBM MQ , è possibile utilizzare la funzionalità messaggio - a - file di Managed File Transfer per trasferire questi messaggi a un file su qualsiasi sistema nella rete Managed File Transfer .

Attività correlate

“Configurazione di un agent per eseguire trasferimenti da messaggio a file” a pagina 281
Per impostazione predefinita, gli agent non possono eseguire trasferimenti da messaggio a file o da file a messaggio. Per abilitare questa funzione, è necessario impostare la proprietà dell'agente `enableQueueInputOutput` su true.

“Esempio: trasferimento di un gruppo di messaggi da una coda a un unico file” a pagina 283
È possibile specificare un singolo gruppo completo su una coda IBM MQ come origine di un trasferimento file utilizzando i parametri **-sq** e **-sqgi** con il comando **fteCreateTransfer** .

“Esempio: inserimento di un delimitatore di testo prima dei dati da ciascun messaggio” a pagina 284
Quando si esegue il trasferimento in modalità testo da una coda di origine a un file, è possibile specificare che un delimitatore di testo venga inserito prima dei dati dei singoli messaggi utilizzando i parametri **-sq**, **-sqdt** e **-sqdp** con il comando **fteCreateTransfer** .

“Esempio: inserimento di un delimitatore binario dopo i dati da ciascun messaggio” a pagina 285
Quando si esegue il trasferimento in modo binario da una coda di origine a un file, è possibile specificare che venga inserito un delimitatore binario dopo i dati dei singoli messaggi utilizzando i parametri **-sq**, **-sqdb**e **-sqdp** con il comando **fteCreateTransfer** .

“Monitoraggio di una coda e utilizzo della sostituzione di variabili” a pagina 247
È possibile monitorare una coda e trasferire messaggi dalla coda monitorata a un file utilizzando il comando **fteCreateMonitor** . Il valore di qualsiasi proprietà del messaggio IBM MQ nel primo messaggio da leggere dalla coda monitorata può essere sostituito nella definizione XML dell'attività e utilizzato per definire il comportamento del trasferimento.

“Esempio: errore di trasferimento da messaggio a file utilizzando le proprietà del messaggio IBM MQ” a pagina 289
È possibile causare l'esito negativo del trasferimento file di un messaggio impostando la proprietà del messaggio `usr.UserReturnCode` IBM MQ su un valore diverso da zero. È inoltre possibile specificare ulteriori informazioni sul motivo dell'errore impostando la proprietà del messaggio `usr.UserSupplement` IBM MQ .

Riferimenti correlati

Proprietà dei messaggi MQ lette da MFT dai messaggi sulle code origine
fteCreateTransfer: avviare un nuovo trasferimento file

Esempio: trasferimento di un gruppo di messaggi da una coda a un unico file

È possibile specificare un singolo gruppo completo su una coda IBM MQ come origine di un trasferimento file utilizzando i parametri **-sq** e **-sqgi** con il comando **fteCreateTransfer** .

Informazioni su questa attività

In questo esempio, ci sono dieci messaggi sulla coda START_QUEUE. Questa coda deve essere sul gestore code dell'agente di origine, *QM_NEPTUNE*. I primi tre messaggi appartengono a un gruppo con IBM MQ ID gruppo `41424b3ef3a22020202020202020202020202020201111`; questo gruppo non è un gruppo completo. I successivi cinque messaggi appartengono ad un gruppo con IBM MQ ID gruppo `41424b3ef3a22020202020202020202020202020202222`; questo gruppo è completo. I restanti due messaggi appartengono ad un gruppo con IBM MQ ID gruppo `41424b3ef3a22020202020202020202020202020203333`; questo gruppo è completo.

Procedura

Immettere il seguente comando:

```
fteCreateTransfer -sa AGENT_NEPTUNE -sm QM_NEPTUNE -da AGENT_VENUS
                  -df /out/group.txt -sqgi -sq START_QUEUE
```

I dati nei messaggi che appartengono al primo gruppo completo nella coda *START_QUEUE*, il gruppo con IBM MQ ID gruppo 41424b3ef3a220202020202020202020202020202020202222, vengono scritti nel file `/out/group.txt` sul sistema su cui è in esecuzione `AGENT_VENUS` .

Concetti correlati

“Trasferimento dei dati dai messaggi ai file” a pagina 280
La funzione messaggio - a - file di Managed File Transfer consente di trasferire i dati da uno o più messaggi su una coda IBM MQ a un file, a un dataset (su z/OS) o a uno spazio file utente. Se si dispone di un'applicazione che crea o elabora messaggi IBM MQ , è possibile utilizzare la funzionalità messaggio - a - file di Managed File Transfer per trasferire questi messaggi a un file su qualsiasi sistema nella rete Managed File Transfer .

Attività correlate

“Configurazione di un agent per eseguire trasferimenti da messaggio a file” a pagina 281
Per impostazione predefinita, gli agent non possono eseguire trasferimenti da messaggio a file o da file a messaggio. Per abilitare questa funzione, è necessario impostare la proprietà dell'agente `enableQueueInputOutput` su true.

“Esempio: trasferimento da una coda a un file singolo” a pagina 282
È possibile specificare una coda IBM MQ come origine di trasferimento file utilizzando il parametro **-sq** con il comando **fteCreateTransfer** .

“Esempio: inserimento di un delimitatore di testo prima dei dati da ciascun messaggio” a pagina 284
Quando si esegue il trasferimento in modalità testo da una coda di origine a un file, è possibile specificare che un delimitatore di testo venga inserito prima dei dati dei singoli messaggi utilizzando i parametri **-sq**, **-sqdt** e **-sqdp** con il comando **fteCreateTransfer** .

“Esempio: inserimento di un delimitatore binario dopo i dati da ciascun messaggio” a pagina 285
Quando si esegue il trasferimento in modo binario da una coda di origine a un file, è possibile specificare che venga inserito un delimitatore binario dopo i dati dei singoli messaggi utilizzando i parametri **-sq**, **-sqdb**e **-sqdp** con il comando **fteCreateTransfer** .

“Monitoraggio di una coda e utilizzo della sostituzione di variabili” a pagina 247
È possibile monitorare una coda e trasferire messaggi dalla coda monitorata a un file utilizzando il comando **fteCreateMonitor** . Il valore di qualsiasi proprietà del messaggio IBM MQ nel primo messaggio da leggere dalla coda monitorata può essere sostituito nella definizione XML dell'attività e utilizzato per definire il comportamento del trasferimento.

“Esempio: errore di trasferimento da messaggio a file utilizzando le proprietà del messaggio IBM MQ” a pagina 289
È possibile causare l'esito negativo del trasferimento file di un messaggio impostando la proprietà del messaggio `usr.UserReturnCode` IBM MQ su un valore diverso da zero. È inoltre possibile specificare ulteriori informazioni sul motivo dell'errore impostando la proprietà del messaggio `usr.UserSupplement` IBM MQ .

Riferimenti correlati

fteCreateTransfer: avviare un nuovo trasferimento file

Esempio: inserimento di un delimitatore di testo prima dei dati da ciascun messaggio

Quando si esegue il trasferimento in modalità testo da una coda di origine a un file, è possibile specificare che un delimitatore di testo venga inserito prima dei dati dei singoli messaggi utilizzando i parametri **-sq**, **-sqdt** e **-sqdp** con il comando **fteCreateTransfer** .

Informazioni su questa attività

In questo esempio, ci sono quattro messaggi sulla coda START_QUEUE. Questa coda è sul gestore code dell'agente di origine, *QM_NEPTUNE*. Il delimitatore di testo da inserire prima dei dati da ciascun messaggio può essere espresso come una stringa letterale Java , ad esempio `\n\u002D\u002D\u002D\n`.

Procedura

Immettere il seguente comando:

```
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
```

Il delimitatore di testo viene aggiunto all'inizio dei dati da ognuno dei quattro messaggi su START_QUEUE dall'agente origine, AGENT_NEPTUNE. Questi dati vengono scritti sul file di destinazione, `/out/output.txt`.

Concetti correlati

“Trasferimento dei dati dai messaggi ai file” a pagina 280
La funzione messaggio - a - file di Managed File Transfer consente di trasferire i dati da uno o più messaggi su una coda IBM MQ a un file, a un dataset (su z/OS) o a uno spazio file utente. Se si dispone di un'applicazione che crea o elabora messaggi IBM MQ , è possibile utilizzare la funzionalità messaggio - a - file di Managed File Transfer per trasferire questi messaggi a un file su qualsiasi sistema nella rete Managed File Transfer .

Attività correlate

“Configurazione di un agent per eseguire trasferimenti da messaggio a file” a pagina 281
Per impostazione predefinita, gli agent non possono eseguire trasferimenti da messaggio a file o da file a messaggio. Per abilitare questa funzione, è necessario impostare la proprietà dell'agente `enableQueueInputOutput` su true.

“Esempio: trasferimento da una coda a un file singolo” a pagina 282
È possibile specificare una coda IBM MQ come origine di trasferimento file utilizzando il parametro **-sq** con il comando **fteCreateTransfer** .

“Esempio: trasferimento di un gruppo di messaggi da una coda a un unico file” a pagina 283
È possibile specificare un singolo gruppo completo su una coda IBM MQ come origine di un trasferimento file utilizzando i parametri **-sq** e **-sqgi** con il comando **fteCreateTransfer** .

“Esempio: inserimento di un delimitatore binario dopo i dati da ciascun messaggio” a pagina 285
Quando si esegue il trasferimento in modo binario da una coda di origine a un file, è possibile specificare che venga inserito un delimitatore binario dopo i dati dei singoli messaggi utilizzando i parametri **-sq**, **-sqdb**e **-sqdp** con il comando **fteCreateTransfer** .

“Monitoraggio di una coda e utilizzo della sostituzione di variabili” a pagina 247
È possibile monitorare una coda e trasferire messaggi dalla coda monitorata a un file utilizzando il comando **fteCreateMonitor** . Il valore di qualsiasi proprietà del messaggio IBM MQ nel primo messaggio da leggere dalla coda monitorata può essere sostituito nella definizione XML dell'attività e utilizzato per definire il comportamento del trasferimento.

“Esempio: errore di trasferimento da messaggio a file utilizzando le proprietà del messaggio IBM MQ” a pagina 289
È possibile causare l'esito negativo del trasferimento file di un messaggio impostando la proprietà del messaggio `usr.UserReturnCode` IBM MQ su un valore diverso da zero. È inoltre possibile specificare ulteriori informazioni sul motivo dell'errore impostando la proprietà del messaggio `usr.UserSupplement` IBM MQ .

Riferimenti correlati

fteCreateTransfer: avviare un nuovo trasferimento file

Esempio: inserimento di un delimitatore binario dopo i dati da ciascun messaggio

Quando si esegue il trasferimento in modo binario da una coda di origine a un file, è possibile specificare che venga inserito un delimitatore binario dopo i dati dei singoli messaggi utilizzando i parametri **-sq**, **-sqdb**e **-sqdp** con il comando **fteCreateTransfer** .

Informazioni su questa attività

In questo esempio, ci sono tre messaggi sulla coda START_QUEUE. Questa coda è sul gestore code dell'agente di origine, *QM_NEPTUNE*. Il delimitatore binario da inserire dopo i dati da ogni messaggio deve essere espresso come un elenco separato da virgole di byte esadecimali, ad esempio `x34,xE7,xAE`.

Procedura

Immettere il seguente comando:

```
fteCreateTransfer -sa AGENT_NEPTUNE -sm QM_NEPTUNE -da AGENT_VENUS -df /out/binary.file
                  -sqdp postfix -sqdb x34,xE7,xAE -sq START_QUEUE
```

Il delimitatore binario viene aggiunto ai dati da ognuno dei tre messaggi su START_QUEUE dall'agent di origine AGENT_NEPTUNE. Questi dati vengono scritti sul file di destinazione, `/out/binary.file`.

Concetti correlati

“Trasferimento dei dati dai messaggi ai file” a pagina 280
La funzione messaggio - a - file di Managed File Transfer consente di trasferire i dati da uno o più messaggi su una coda IBM MQ a un file, a un dataset (su z/OS) o a uno spazio file utente. Se si dispone di un'applicazione che crea o elabora messaggi IBM MQ , è possibile utilizzare la funzionalità messaggio - a - file di Managed File Transfer per trasferire questi messaggi a un file su qualsiasi sistema nella rete Managed File Transfer .

Attività correlate

“Configurazione di un agent per eseguire trasferimenti da messaggio a file” a pagina 281
Per impostazione predefinita, gli agent non possono eseguire trasferimenti da messaggio a file o da file a messaggio. Per abilitare questa funzione, è necessario impostare la proprietà dell'agente `enableQueueInputOutput` su true.

“Esempio: trasferimento da una coda a un file singolo” a pagina 282
È possibile specificare una coda IBM MQ come origine di trasferimento file utilizzando il parametro **-sq** con il comando **fteCreateTransfer** .

“Esempio: trasferimento di un gruppo di messaggi da una coda a un unico file” a pagina 283
È possibile specificare un singolo gruppo completo su una coda IBM MQ come origine di un trasferimento file utilizzando i parametri **-sq** e **-sqgi** con il comando **fteCreateTransfer** .

“Esempio: inserimento di un delimitatore di testo prima dei dati da ciascun messaggio” a pagina 284
Quando si esegue il trasferimento in modalità testo da una coda di origine a un file, è possibile specificare che un delimitatore di testo venga inserito prima dei dati dei singoli messaggi utilizzando i parametri **-sq**, **-sqdt** e **-sqdp** con il comando **fteCreateTransfer** .

“Monitoraggio di una coda e utilizzo della sostituzione di variabili” a pagina 247
È possibile monitorare una coda e trasferire messaggi dalla coda monitorata a un file utilizzando il comando **fteCreateMonitor** . Il valore di qualsiasi proprietà del messaggio IBM MQ nel primo messaggio da leggere dalla coda monitorata può essere sostituito nella definizione XML dell'attività e utilizzato per definire il comportamento del trasferimento.

“Esempio: errore di trasferimento da messaggio a file utilizzando le proprietà del messaggio IBM MQ” a pagina 289
È possibile causare l'esito negativo del trasferimento file di un messaggio impostando la proprietà del messaggio `usr.UserReturnCode` IBM MQ su un valore diverso da zero. È inoltre possibile specificare ulteriori informazioni sul motivo dell'errore impostando la proprietà del messaggio `usr.UserSupplement` IBM MQ .

Riferimenti correlati

fteCreateTransfer: avviare un nuovo trasferimento file

Monitoraggio di una coda e utilizzo della sostituzione di variabili

È possibile monitorare una coda e trasferire messaggi dalla coda monitorata a un file utilizzando il comando **fteCreateMonitor** . Il valore di qualsiasi proprietà del messaggio IBM MQ nel primo messaggio da leggere dalla coda monitorata può essere sostituito nella definizione XML dell'attività e utilizzato per definire il comportamento del trasferimento.

Informazioni su questa attività

In questo esempio, l'agent di origine è denominato AGENT_VENUS, che si connette a QM_VENUS. La coda monitorata da AGENT_VENUS è denominata START_QUEUE e si trova su QM_VENUS. L'agent esegue il polling della coda ogni 30 minuti.

Quando un gruppo completo di messaggi viene scritto nella coda, l'attività di monitoraggio invia il gruppo di messaggi a uno dei diversi agent di destinazione, tutti connessi al gestore code QM_MARS. Il nome del file a cui viene trasferito il gruppo di messaggi viene definito dalla proprietà IBM MQ message `usr.fileName` sul primo messaggio del gruppo. Il nome dell'agent a cui viene inviato il gruppo di messaggi è definito dalla IBM MQ proprietà del messaggio `usr.toAgent` nel primo messaggio del gruppo. Se l'intestazione `usr.toAgent` non è impostata, il valore predefinito da utilizzare per l'agent di destinazione è AGENT_MAGENTA.

Quando si specifica `useGroups="true"`, se non si specifica anche `groupId="${GROUPID}"`, il trasferimento acquisisce solo il primo messaggio sulla coda. Ad esempio, se si sta utilizzando la sostituzione della variabile per generare il fileName, è possibile che il contenuto di a `.txt` non sia corretto. Ciò è dovuto al fatto che fileName viene generato dal monitoraggio, ma il trasferimento in realtà riceve un messaggio che non è quello che dovrebbe generare il file denominato fileName.

Procedura

1. Creare l'attività XML che definisce l'attività che il monitoraggio esegue quando viene attivato.

```
<?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>
```

   Le variabili sostituite con i valori delle intestazioni dei messaggi IBM MQ vengono evidenziati in **grassetto**. Questa attività XML viene salvata nel file `/home/USER1/task.xml`

2. Creare un monitoraggio risorse per monitorare la coda START_QUEUE.

   Immettere il seguente comando:

```
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 utente o un programma scrive un gruppo di messaggi nella coda START_QUEUE.

Il primo messaggio in questo gruppo ha le seguenti proprietà del messaggio IBM MQ :

```
usr.fileName=larmer
usr.toAgent=AGENT_VIOLET
```

4. Il controllo viene attivato quando viene scritto il gruppo completo. L'agent sostituisce le proprietà del messaggio IBM MQ nell'XML dell'attività.

   Ciò determina la trasformazione dell'attività XML in:

```
<?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>
```

Risultati

Viene effettuato il trasferimento definito dall'attività XML. Il gruppo completo di messaggi letti da START_QUEUE da AGENT_VENUS viene scritto in un file denominato `/reports/larmer.rpt` sul sistema su cui è in esecuzione AGENT_VIOLET.

Operazioni successive

Trasferimento di ogni messaggio in un file separato

Se si desidera monitorare una coda e trasferire ogni messaggio in un file separato, è possibile utilizzare una tecnica simile a quella descritta in precedenza in questo argomento.

1. Creare il controllo come descritto in precedenza, specificando il parametro **`-tr completeGroups`** nel comando **`fteCreateMonitor`** .
2. Nell'XML dell'attività specificare quanto segue:

```
<queue useGroups="true" groupId="${GROUPID}">START_QUEUE</queue>
```

Tuttavia, quando si inserono i messaggi nella coda di origine, non inserirli in un gruppo IBM MQ . Aggiungere le proprietà del messaggio IBM MQ a ciascun messaggio. Ad esempio, specificare la proprietà `usr.filename` con un valore di nome file univoco per ogni messaggio. Ciò fa sì che Managed File Transfer Agent consideri ciascun messaggio sulla coda di origine come un gruppo separato.

Concetti correlati

“Trasferimento dei dati dai messaggi ai file” a pagina 280

La funzione messaggio - a - file di Managed File Transfer consente di trasferire i dati da uno o più messaggi su una coda IBM MQ a un file, a un dataset (su z/OS) o a uno spazio file utente. Se si dispone di un'applicazione che crea o elabora messaggi IBM MQ , è possibile utilizzare la funzionalità messaggio - a - file di Managed File Transfer per trasferire questi messaggi a un file su qualsiasi sistema nella rete Managed File Transfer .

“Personalizzazione delle attività di monitoraggio delle risorse MFT con la sostituzione della variabile” a pagina 242
Quando le condizioni del trigger di un controllo risorse attivo vengono soddisfatte, viene richiamata l'attività definita. Oltre a richiamare l'attività di trasferimento o di comando con lo stesso agent di destinazione o con lo stesso nome file di destinazione ogni volta, è anche possibile modificare la definizione dell'attività in fase di runtime. A tale scopo, inserire i nomi delle variabili nel file XML di definizione attività. Quando il controllo determina che le condizioni di trigger sono soddisfatte e che la definizione dell'attività contiene nomi di variabili, sostituisce i nomi di variabile con valori di variabile e richiama l'attività.

Cosa fare se i file di destinazione creati da un trasferimento avviato da un controllo risorse della coda contengono dati errati

Attività correlate

“Configurazione delle attività di monitoraggio MFT per avviare comandi e script” a pagina 235
I monitoraggi risorse non sono limitati all'esecuzione di trasferimenti file come attività associata. È anche possibile configurare il monitor per richiamare altri comandi dall'agent di monitoraggio, inclusi i programmi eseguibili, gli script Ant o i lavori JCL. Per richiamare i comandi, modificare l'XML di definizione dell'attività di monitoraggio in modo da includere uno o più elementi di comando con i parametri di chiamata del comando corrispondenti, ad esempio argomenti e proprietà.

“Esempio: configurazione di una risorsa MFT” a pagina 241
È possibile specificare una coda IBM MQ come risorsa che deve essere monitorata da un monitoraggio risorse utilizzando il parametro **-mq** con il comando **fteCreateMonitor** .

Riferimenti correlati

fteCreateMonitor: crea un monitoraggio risorse MFT

Proprietà dei messaggi MQ lette da MFT dai messaggi sulle code origine

Esempio: errore di trasferimento da messaggio a file utilizzando le proprietà del messaggio IBM MQ

È possibile causare l'esito negativo del trasferimento file di un messaggio impostando la proprietà del messaggio `usr.UserReturnCode` IBM MQ su un valore diverso da zero. È inoltre possibile specificare ulteriori informazioni sul motivo dell'errore impostando la proprietà del messaggio `usr.UserSupplement` IBM MQ .

Informazioni su questa attività

In questo esempio, è in corso un trasferimento tra la coda INPUT_QUEUE e il file `/home/user/output.file`.

Un utente sta creando messaggi e li sta collocando nella coda INPUT_QUEUE. L'agent di origine sta utilizzando i messaggi dalla coda INPUT_QUEUE e sta inviando i dati di trasferimento all'agent di destinazione. L'agent di destinazione sta scrivendo questi dati nel file `/home/user/output.file`.

L'utente che scrive i messaggi nella coda INPUT_QUEUE desidera arrestare il trasferimento in corso ed eliminare i dati già scritti nel file di destinazione.

Procedura

1. L'utente scrive un messaggio nella coda INPUT_QUEUE con le seguenti proprietà del messaggio IBM MQ impostate:

   ```
   usr.UserReturnCode=1
   usr.UserSupplement="Cancelling transfer - sent wrong data."
   ```

2. L'agent di origine legge le proprietà del messaggio IBM MQ e arresta l'elaborazione dei messaggi dalla coda. L'agent di destinazione elimina tutti i dati del file che sono stati scritti nella directory di destinazione.

3. L'agent di origine invia un messaggio di log di trasferimento al gestore code di coordinamento che riporta l'errore di trasferimento.

   Il messaggio contiene le seguenti informazioni:

```
<?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>
```

Concetti correlati

“Trasferimento dei dati dai messaggi ai file” a pagina 280
La funzione messaggio - a - file di Managed File Transfer consente di trasferire i dati da uno o più messaggi su una coda IBM MQ a un file, a un dataset (su z/OS) o a uno spazio file utente. Se si dispone di un'applicazione che crea o elabora messaggi IBM MQ , è possibile utilizzare la funzionalità messaggio - a - file di Managed File Transfer per trasferire questi messaggi a un file su qualsiasi sistema nella rete Managed File Transfer .

Attività correlate

“Configurazione di un agent per eseguire trasferimenti da messaggio a file” a pagina 281
Per impostazione predefinita, gli agent non possono eseguire trasferimenti da messaggio a file o da file a messaggio. Per abilitare questa funzione, è necessario impostare la proprietà dell'agente `enableQueueInputOutput` su true.

“Esempio: trasferimento da una coda a un file singolo” a pagina 282
È possibile specificare una coda IBM MQ come origine di trasferimento file utilizzando il parametro **-sq** con il comando **fteCreateTransfer** .

“Esempio: trasferimento di un gruppo di messaggi da una coda a un unico file” a pagina 283
È possibile specificare un singolo gruppo completo su una coda IBM MQ come origine di un trasferimento file utilizzando i parametri **-sq** e **-sqgi** con il comando **fteCreateTransfer** .

“Esempio: inserimento di un delimitatore di testo prima dei dati da ciascun messaggio” a pagina 284
Quando si esegue il trasferimento in modalità testo da una coda di origine a un file, è possibile specificare che un delimitatore di testo venga inserito prima dei dati dei singoli messaggi utilizzando i parametri **-sq**, **-sqdt** e **-sqdp** con il comando **fteCreateTransfer** .

“Esempio: inserimento di un delimitatore binario dopo i dati da ciascun messaggio” a pagina 285

Quando si esegue il trasferimento in modo binario da una coda di origine a un file, è possibile specificare che venga inserito un delimitatore binario dopo i dati dei singoli messaggi utilizzando i parametri **-sq**, **-sqdb**e **-sqdp** con il comando **fteCreateTransfer** .

“Monitoraggio di una coda e utilizzo della sostituzione di variabili” a pagina 247
È possibile monitorare una coda e trasferire messaggi dalla coda monitorata a un file utilizzando il comando **fteCreateMonitor** . Il valore di qualsiasi proprietà del messaggio IBM MQ nel primo messaggio da leggere dalla coda monitorata può essere sostituito nella definizione XML dell'attività e utilizzato per definire il comportamento del trasferimento.

Riferimenti correlati

Proprietà dei messaggi MQ lette da MFT dai messaggi sulle code origine

Il bridge di protocollo

Il bridge di protocollo consente alla rete Managed File Transfer (MFT) di accedere ai file memorizzati su un server di file esterno alla rete MFT , nel dominio locale o in un'ubicazione remota. Questo server di file può utilizzare i protocolli di rete FTP, FTPS o SFTP. Ogni server di file richiede almeno un agent dedicato. L'agent dedicato è noto come agent bridge di protocollo. Un agent bridge può interagire con più server di file.

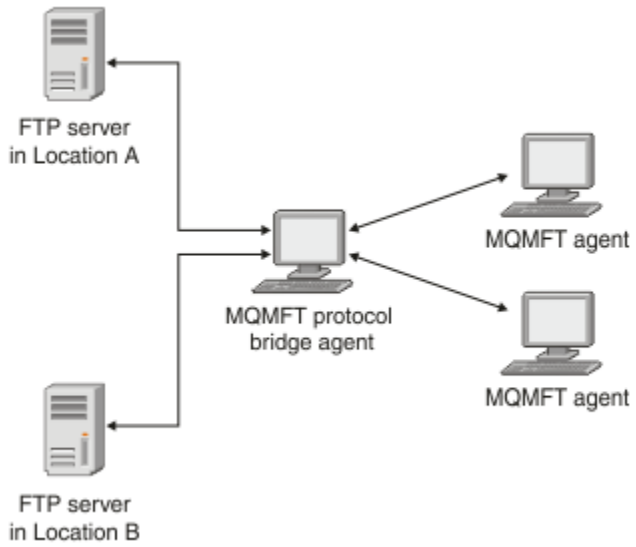
Il bridge di protocollo è disponibile come parte del componente Service di Managed File Transfer. È possibile disporre di più agent dedicati su un singolo sistema su cui è in esecuzione MFT che si collegano a server di file diversi.

È possibile utilizzare un agent bridge di protocollo per trasferire file a più endpoint contemporaneamente. MFT fornisce un file denominato `ProtocolBridgeProperties.xml` che è possibile modificare per definire i diversi server di file del protocollo a cui si desidera trasferire i file. Il comando **fteCreateBridgeAgent** aggiunge automaticamente i dettagli del server di file del protocollo predefinito a `ProtocolBridgeProperties.xml` . Questo file è descritto in Formato file delle proprietà del bridge di protocollo.

È possibile utilizzare l'agent bridge di protocollo per eseguire le azioni riportate di seguito:

- Caricare i file dalla rete MFT su un server remoto utilizzando FTP, FTPS o SFTP.
- Scaricare i file da un server remoto, utilizzando FTP, FTPS o SFTP, nella rete MFT

Nota: L'agent bridge di protocollo può supportare solo server FTP, FTPS o SFTP che consentono l'accesso ai file tramite il percorso file assoluto. Se viene specificato un percorso file relativo in una richiesta di trasferimento, l'agent bridge di protocollo tenterà di convertire il percorso relativo in un percorso file assoluto basato sulla directory home utilizzata per accedere al server del protocollo. I server di protocollo che consentono l'accesso ai file basati solo sulla directory corrente non sono supportati dall'agente bridge di protocollo.



Il diagramma mostra due server FTP, in diverse posizioni. I server FTP vengono utilizzati per scambiare file con gli agenti Managed File Transfer . L'agent bridge di protocollo si trova tra i server FTP e il resto della rete MFT ed è configurato per comunicare con entrambi i server FTP.

Assicurarsi di disporre di un altro agent nella rete MFT in aggiunta all'agent bridge di protocollo. L'agent bridge di protocollo è un bridge solo per il server FTP, FTPS o SFTP e non scrive i file trasferiti sul disco locale. Se si desidera trasferire i file da o verso il server FTP, FTPS o SFTP, è necessario utilizzare l'agent bridge di protocollo come destinazione o origine per il trasferimento file (che rappresenta il server FTP, FTPS o SFTP) e un altro agent standard come origine o destinazione corrispondente.

Quando si trasferiscono i file utilizzando il bridge di protocollo, il bridge deve disporre dell'autorizzazione per leggere la directory di origine o di destinazione contenente i file che si desidera trasferire. Ad esempio, se si desidera trasferire i file dalla directory `/home/fte/bridge` che dispone solo di autorizzazioni di esecuzione (d -- x -- x -- x), i trasferimenti tentati da questa directory hanno esito negativo con il seguente messaggio di errore:

```
BFGBR0032E: Attempt to read filename from the protocol file server
  has failed with server error 550. Failed to open file.
```

Configurazione di un agent bridge di protocollo

Un agent bridge di protocollo è come un agent MFT standard. Creare un agent bridge di protocollo utilizzando il comando **`fteCreateBridgeAgent`** . È possibile configurare un agent bridge di protocollo utilizzando il file `ProtocolBridgeProperties.xml` , descritto in Formato file delle proprietà bridge di protocollo. Se si utilizza una versione precedente, configurare l'agent utilizzando le proprietà bridge di protocollo specifiche descritte in Proprietà agent avanzate: bridge di protocollo e Proprietà agent avanzate: registrazione agent bridge di protocollo. Per tutte le versioni, è anche possibile configurare un'associazione credenziali come descritto in “Associazione delle credenziali per un server di file” a pagina 300. Dopo aver configurato un agent bridge di protocollo per un particolare server di file di protocollo, è possibile utilizzare tale agent solo per tale scopo.

Ripristino bridge di protocollo

Se l'agent bridge di protocollo non è in grado di collegarsi al server di file perché il server di file non è disponibile, tutte le richieste di trasferimento file vengono accodate fino a quando il server di file non diventa disponibile. Se l'agent bridge di protocollo non è in grado di collegarsi al server di file perché l'agent sta utilizzando le credenziali errate, il trasferimento non riesce e il messaggio del log di trasferimento riflette questo errore. Se l'agent bridge di protocollo viene terminato per qualsiasi ragione, tutti i trasferimenti file richiesti vengono conservati e continuano quando il bridge di protocollo viene riavviato.

Durante il trasferimento file, i file vengono generalmente scritti come file temporanei nella destinazione e vengono ridenominati quando il trasferimento è completo. Tuttavia, se la destinazione di trasferimento è un server di file di protocollo configurato come scrittura limitata (gli utenti possono caricare i file sul server di file di protocollo ma non possono modificare in alcun modo i file caricati; in effetti gli utenti possono scrivere una sola volta), i file trasferiti vengono scritti direttamente nella destinazione. Ciò significa che se si verifica un problema durante il trasferimento, i file scritti parzialmente rimangono sul server di file del protocollo di destinazione e Managed File Transfer non può eliminare o modificare tali file. In questa situazione, il trasferimento non riesce.

Attività correlate

"Esempio: come configurare un agent bridge di protocollo per utilizzare le credenziali della chiave privata con un server SFTP UNIX" a pagina 305
Questo esempio dimostra come generare e configurare il file `ProtocolBridgeCredentials.xml` . Questo esempio è un esempio tipico e i dettagli possono variare in base alla propria piattaforma, ma i principi rimangono gli stessi.

"Definizione delle proprietà per i server di file del protocollo utilizzando il file ProtocolBridgeProperties.xml" a pagina 293
Definire le proprietà di uno o più server di file di protocollo a cui si desidera trasferire i file utilizzando il file `ProtocolBridgeProperties.xml` , fornito da Managed File Transfer nella directory di configurazione dell'agent.

Riferimenti correlati

fteCreateBridgeAgent (creazione e configurazione di un agent bridge di protocollo MFT)

"Associazione delle credenziali per un server di file" a pagina 300
Associare le credenziali utente in Managed File Transfer alle credenziali utente sul server di file utilizzando la funzione di associazione credenziali predefinita dell'agent bridge di protocollo o scrivendo la propria uscita utente. Managed File Transfer fornisce un'uscita utente di esempio che esegue l'associazione delle credenziali utente.

Interfaccia ProtocolBridgeCredentialExit.java

User exit delle credenziali del bridge di protocollo di esempio

Supporto server FTPS dal bridge di protocollo

Definizione delle proprietà per i server di file del protocollo utilizzando il file `ProtocolBridgeProperties.xml`

Definire le proprietà di uno o più server di file di protocollo a cui si desidera trasferire i file utilizzando il file `ProtocolBridgeProperties.xml` , fornito da Managed File Transfer nella directory di configurazione dell'agent.

Informazioni su questa attività

Il comando **fteCreateBridgeAgent** crea il file `ProtocolBridgeProperties.xml` nella directory di configurazione dell'agente *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*bridge_agent_name*. Il comando crea anche una voce nel file per il server del file del protocollo predefinito, se è stato specificato un valore predefinito quando è stato eseguito il comando.

Il messaggio BFGCL0392I fornisce l'ubicazione del file `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>
```

Il comando può generare il seguente messaggio:BFGCL0532I:

```
Perché l'agent funzioni, è necessario creare manualmente un file delle credenziali aggiuntivo.
Per impostazione predefinita, questo file è denominato ProtocolBridgeCredentials.xml e si trova
nella home
della directory dell'utente che avvia l'agente. Ad esempio, se questo utente ha avviato l'agent
l'ubicazione è: $HOME/ProtocolBridgeCredentials.xml
```

Se si utilizza un file delle credenziali:

1. Consultare il seguente testo per ulteriori informazioni su come crearne uno.
2. Il file delle credenziali deve trovarsi in una directory con autorizzazioni limitate. Ad esempio, non ci deve essere alcun accesso in lettura per altri utenti.
3. Specificare l'ubicazione della directory per il file delle credenziali nella variabile di ambiente $HOME per l'ID utente dell'agent avviato oppure modificare il file `ProtocolBridgeProperties.xml` e specificare l'ubicazione in:

   ```
   <tns:credentialsFile path="/test/ProtocolBridgeCredentials.xml"/>
   ```

Se si desidera aggiungere ulteriori server di protocollo non predefiniti, modificare questo file per definirne le proprietà. Questo esempio aggiunge un server FTP aggiuntivo.

Nota: L'agent bridge di protocollo non supporta il blocco dei file. Ciò è dovuto al fatto che Managed File Transfer non supporta il meccanismo di blocco file su un server di file.

Procedura

1. Definire un server di file di protocollo inserendo le righe seguenti nel file come elemento child di `<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. Quindi modificare il valore degli attributi:
   - `name` è il nome del server di file del protocollo
   - `host` è il nome host o l'indirizzo IP del server di file del protocollo
   - `port` è il numero di porta del server di file del protocollo
   - `platform` è la piattaforma su cui viene eseguito il server di file del protocollo
   - `timeZone` è il fuso orario in cui viene eseguito il server di file del protocollo
   - `locale` è la lingua utilizzata sul server di file del protocollo
   - `fileEncoding` è la codifica dei caratteri del server di file del protocollo
   - `listFormat` è il formato di elenco file restituito dal server di file del protocollo
   - `limitedWrite` determina se seguire la modalità predefinita durante la scrittura su un server di file, che consiste nel creare un file temporaneo e ridenominare tale file una volta completato il trasferimento. Per un server di file configurato come sola scrittura, il file viene creato direttamente con il nome finale. Il valore di questa proprietà può essere true o false. L'attributo `limitedWrite` e la proprietà dell'agent `doNotUseTempOutputFile` vengono utilizzati insieme nel caso di agent bridge di protocollo. Se si desidera utilizzare i file temporanei, non è necessario impostare il valore di `doNotUseTempOutputFile`e il valore di `limitedWrite` deve essere impostato su false. Qualsiasi altra combinazione di impostazioni significa che i file temporanei non verranno utilizzati.
   - `maxListFileNames` è il numero massimo di nomi raccolti durante la scansione di una directory sul server di file del protocollo per i nomi file.
   - `maxListDirectoryLevels` è il numero massimo di livelli di directory da ricorre quando si esegue la scansione di una directory sul server di file del protocollo per i nomi file.

   Per ulteriori dettagli su questi attributi, incluso se sono obbligatori o facoltativi e i relativi valori predefiniti, consultare Formato file delle proprietà del bridge di protocollo.

Riferimenti correlati

Formato file delle proprietà bridge di protocollo

Espressioni regolari utilizzate da MFT

Ricerca delle proprietà del server di file del protocollo: ProtocolBridgePropertiesExit2

Se si dispone di un numero elevato di server di file di protocollo, è possibile implementare l'interfaccia `com.ibm.wmqfte.exitroutine.api.ProtocolBridgePropertiesExit2` per ricercare le proprietà del server di file di protocollo a cui si fa riferimento nei trasferimenti. È possibile implementare questa interfaccia preferendo mantenere un file `ProtocolBridgeProperties.xml` .

Informazioni su questa attività

Managed File Transfer fornisce un'uscita utente di esempio che ricerca le proprietà del server di file di protocollo. Per ulteriori informazioni, consultare “Utilizzo dell'uscita utente di esempio per la ricerca delle proprietà del server di file del protocollo” a pagina 297.

Qualsiasi uscita utente che ricerca le proprietà bridge di protocollo deve implementare l'interfaccia `com.ibm.wmqfte.exitroutine.api.ProtocolBridgePropertiesExit2`. Per ulteriori informazioni, consultare ProtocolBridgePropertiesExit.java interface.

È possibile concatenare più uscite delle proprietà del server di protocollo in modo simile ad altre uscite utente. Le uscite vengono richiamati nell'ordine in cui vengono specificate utilizzando la proprietà `protocolBridgePropertiesExitClasses` del file delle proprietà agent. I metodi di inizializzazione vengono restituiti tutti separatamente e se uno o più restituisce un valore false, l'agent non viene avviato. L'errore viene riportato nella registrazione eventi dell'agent.

Viene restituito un unico risultato complessivo per i metodi getProtocolServerProperties di tutte le uscite. Se il metodo restituisce un oggetto proprietà come codice risultato, questo valore è il risultato restituito e i metodi getProtocolServerProperties delle uscite successive non vengono richiamati. Se il metodo restituisce un valore null come codice di risultato, viene richiamato il metodo getProtocolServerProperties dell'uscita successiva. Se non vi è alcuna uscita successiva, viene restituito il risultato null. Un codice di risultato globale null viene considerato come un errore di ricerca dall'agente bridge di protocollo.

Si consiglia di utilizzare l'interfaccia ProtocolBridgePropertiesExit2.java , ma per informazioni sull'interfaccia ProtocolBridgePropertiesExit.java, consultare “Ricerca delle proprietà del server di file del protocollo: ProtocolBridgePropertiesExit” a pagina 298.

Per eseguire l'uscita, completare la seguente procedura:

Procedura

1. Compilare l'uscita utente delle proprietà del server di protocollo.
2. Creare un file JAR (Java archive) contenente l'exit compilata e la relativa struttura del package.
3. Inserire il file JAR contenente la classe di uscita nella directory `exits` dell'agent bridge di protocollo. Questa directory si trova nella directory *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*bridge_agent_name* .
4. Modificare il file delle proprietà dell'agent bridge di protocollo per includere la proprietà `protocolBridgePropertiesExitClasses`. Per il valore di questa proprietà, specificare un elenco separato da virgole di classi che implementano un'uscita utente delle proprietà del server bridge di protocollo. Le classi di uscita vengono richiamate nell'ordine in cui sono specificate in questo elenco. Per ulteriori informazioni, vedere Il file MFT `agent.properties`.
5. Facoltativamente, è possibile specificare la proprietà `protocolBridgePropertiesConfiguration` . Il valore specificato per questa proprietà viene passato come stringa al metodo initialize () delle classi di uscita specificate da `protocolBridgePropertiesExitClasses`. Per ulteriori informazioni, vedere Il file MFT `agent.properties`.

Utilizzo dell'uscita utente di esempio per la ricerca delle proprietà del server di file del protocollo

Managed File Transfer fornisce un'uscita utente di esempio che ricerca le proprietà del server di file di protocollo.

Informazioni su questa attività

Un'uscita utente di esempio che cerca le proprietà del bridge di protocollo viene fornita nella directory *MQ_INSTALLATION_PATH*`/mqft/samples/protocolBridge` e nell'uscita utente Proprietà bridge di protocollo di esempio.

L'uscita `SamplePropertiesExit2.java` legge un file delle proprietà che contiene le proprietà per i server di protocolli. Il formato di ciascuna voce nel file delle proprietà è il seguente:

```
serverName=type://host:port
```

L'ubicazione del file delle proprietà viene presa dalla proprietà dell'agent bridge di protocollo `protocolBridgePropertiesConfiguration`.

Per eseguire l'uscita utente di esempio, completare la seguente procedura:

Procedura

1. Compilare il file `SamplePropertiesExit2.java` .
2. Creare un file JAR contenente l'uscita compilata e la relativa struttura del package.
3. Inserire il file JAR nella directory *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*bridge_agent*`/exits` .

4. Modificare il file *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*bridge_agent_name*/agent.properties in modo che contenga la riga:

```
protocolBridgePropertiesExitClasses=SamplePropertiesExit2
```

5. Creare un file delle proprietà del bridge di protocollo, ad esempio protocol_bridge_properties.properties, nella directory *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*bridge_agent*. Modificare questo file per includere le voci nel formato:

```
serverName=type://host:port
```

6. Modificare il file *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*bridge_agent*/agent.properties in modo che contenga la riga:

```
protocolBridgePropertiesConfiguration=MQ_DATA_PATH/mqft/config/coordination_queue_manager/
agents/bridge_agent/protocol_bridge_properties.properties
```

   È necessario utilizzare il percorso assoluto del file protocol_bridge_properties.properties .
7. Avviare l'agent bridge di protocollo utilizzando il comando **fteStartAgent** .

Concetti correlati

“Il bridge di protocollo” a pagina 291
Il bridge di protocollo consente alla rete Managed File Transfer (MFT) di accedere ai file memorizzati su un server di file esterno alla rete MFT , nel dominio locale o in un'ubicazione remota. Questo server di file può utilizzare i protocolli di rete FTP, FTPS o SFTP. Ogni server di file richiede almeno un agent dedicato. L'agent dedicato è noto come agent bridge di protocollo. Un agent bridge può interagire con più server di file.

Riferimenti correlati

Interfaccia ProtocolBridgePropertiesExit.java

User exit delle proprietà del bridge di protocollo di esempio

Il file MFT agent.properties

fteCreateBridgeAgent (creazione e configurazione di un agent bridge di protocollo MFT)

Ricerca delle proprietà del server di file del protocollo: ProtocolBridgePropertiesExit

Se si dispone di un numero elevato di server di file di protocollo, è possibile implementare l'interfaccia com.ibm.wmqfte.exitroutine.api.ProtocolBridgePropertiesExit per ricercare le proprietà del server di file di protocollo a cui si fa riferimento nei trasferimenti.

Informazioni su questa attività

È possibile implementare l'interfaccia com.ibm.wmqfte.exitroutine.api.ProtocolBridgePropertiesExit piuttosto che gestire un file ProtocolBridgeProperties.xml . Utilizzare l'interfaccia ProtocolBridgePropertiesExit2.java . Il metodo **getCredentialLocation** in ProtocolBridgePropertiesExit2.java utilizza l'ubicazione predefinita del file ProtocolBridgeCredentials.xml , che è la propria directory home.

Qualsiasi uscita utente che ricerca le proprietà bridge di protocollo deve implementare l'interfaccia 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);

}
```

È possibile concatenare più uscite delle proprietà del server di protocollo in modo simile ad altre uscite utente. Le uscite vengono richiamati nell'ordine in cui vengono specificate utilizzando la proprietà `protocolBridgePropertiesExitClasses` del file delle proprietà agent. I metodi di inizializzazione vengono restituiti tutti separatamente e se uno o più restituisce un valore false, l'agent non viene avviato. L'errore viene riportato nella registrazione eventi dell'agent.

Viene restituito un unico risultato complessivo per i metodi getProtocolServerProperties di tutte le uscite. Se il metodo restituisce un oggetto proprietà come codice risultato, questo valore è il risultato restituito e i metodi getProtocolServerProperties delle uscite successive non vengono richiamati. Se il metodo restituisce un valore null come codice di risultato, viene richiamato il metodo getProtocolServerProperties dell'uscita successiva. Se non vi è alcuna uscita successiva, viene restituito il risultato null. Un codice di risultato globale null viene considerato come un errore di ricerca dall'agente bridge di protocollo.

Procedura

Per eseguire l'uscita, completare la seguente procedura:

1. Compilare l'uscita utente delle proprietà del server di protocollo.
2. Creare un file JAR (Java archive) contenente l'exit compilata e la relativa struttura del package.
3. Inserire il file JAR contenente la classe di uscita nella directory `exits` dell'agent bridge di protocollo.

Questa directory si trova nella directory *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*bridge_agent_name* .

4. Modificare il file delle proprietà dell'agent bridge di protocollo per includere la proprietà protocolBridgePropertiesExitClasses.

   Per il valore di questa proprietà, specificare un elenco separato da virgole di classi che implementano un'uscita utente delle proprietà del server bridge di protocollo. Le classi di uscita vengono richiamate nell'ordine in cui sono specificate in questo elenco. Per ulteriori informazioni, vedere Il file MFT agent.properties.
5. Facoltativamente, è possibile specificare la proprietà protocolBridgePropertiesConfiguration .

   Il valore specificato per questa proprietà viene passato come stringa al metodo initialize () delle classi di uscita specificate da protocolBridgePropertiesExitClasses. Per ulteriori informazioni, vedere Il file MFT agent.properties.

Associazione delle credenziali per un server di file

Associare le credenziali utente in Managed File Transfer alle credenziali utente sul server di file utilizzando la funzione di associazione credenziali predefinita dell'agent bridge di protocollo o scrivendo la propria uscita utente. Managed File Transfer fornisce un'uscita utente di esempio che esegue l'associazione delle credenziali utente.

Concetti correlati

“Il bridge di protocollo” a pagina 291
Il bridge di protocollo consente alla rete Managed File Transfer (MFT) di accedere ai file memorizzati su un server di file esterno alla rete MFT , nel dominio locale o in un'ubicazione remota. Questo server di file può utilizzare i protocolli di rete FTP, FTPS o SFTP. Ogni server di file richiede almeno un agent dedicato. L'agent dedicato è noto come agent bridge di protocollo. Un agent bridge può interagire con più server di file.

Attività correlate

“Associazione delle credenziali per un server di file utilizzando il file ProtocolBridgeCredentials.xml” a pagina 300
Associare le credenziali utente in Managed File Transfer alle credenziali utente sul file server utilizzando la funzione di associazione credenziali predefinita dell'agent bridge di protocollo. Managed File Transfer fornisce un file XML che è possibile modificare per includere le informazioni sulle credenziali.

“Associazione delle credenziali per un server di file utilizzando le classi di uscita” a pagina 302
Se non si desidera utilizzare la funzione di associazione credenziali predefinita dell'agent bridge di protocollo, è possibile associare le credenziali utente in Managed File Transfer alle credenziali utente sul server di file scrivendo la propria uscita utente. Se si configurano le uscite utente di associazione credenziali, esse sostituiscono la funzione di associazione credenziali predefinita.

“Esempio: come configurare un agent bridge di protocollo per utilizzare le credenziali della chiave privata con un server SFTP UNIX” a pagina 305
Questo esempio dimostra come generare e configurare il file ProtocolBridgeCredentials.xml . Questo esempio è un esempio tipico e i dettagli possono variare in base alla propria piattaforma, ma i principi rimangono gli stessi.

Riferimenti correlati

Interfaccia ProtocolBridgeCredentialExit.java

User exit delle credenziali del bridge di protocollo di esempio

Il file MFT agent.properties

Associazione delle credenziali per un server di file utilizzando il file ProtocolBridgeCredentials.xml

Associare le credenziali utente in Managed File Transfer alle credenziali utente sul file server utilizzando la funzione di associazione credenziali predefinita dell'agent bridge di protocollo. Managed File Transfer fornisce un file XML che è possibile modificare per includere le informazioni sulle credenziali.

Informazioni su questa attività

Il file `ProtocolBridgeCredentials.xml` deve essere creato manualmente dall'utente. Per impostazione predefinita, il percorso di questo file è la directory home dell'utente che ha avviato l'agent bridge di protocollo, ma può essere memorizzato in qualsiasi punto del file system a cui l'agent può accedere. Per specificare un'altra posizione, aggiungere l'elemento <credentialsFile> al file `ProtocolBridgeProperties.xml` . Ad esempio:

```
<tns:credentialsFile path="/example/path/to/ProtocolBridgeCredentials.xml"/>
```

Prima di poter utilizzare un agent bridge di protocollo, impostare l'associazione delle credenziali modificando questo file per includere le informazioni su host, utente e credenziali. Per ulteriori informazioni ed esempi, consultare Formato del file delle credenziali bridge del protocollo.

Procedura

1. • Modificare la linea <tns:server name="*server name*"> per modificare il valore dell'attributo name nel nome server nel file `ProtocolBridgeProperties.xml` .

   È possibile utilizzare l'attributo del pattern per specificare che è stato utilizzato un nome server che contiene caratteri jolly o espressioni regolari. Ad esempio:

   ```
   <tns:server name="serverA*" pattern="wildcard">
   ```

2. Inserire le informazioni sulle credenziali e sull'ID utente nel file come elementi child di `<tns:server>`.

   È possibile inserire uno o più dei seguenti elementi nel file:

   • Se il server del file di protocollo è un server FTP, FTPS o SFTP, è possibile utilizzare le password per autenticare l'utente che richiede il trasferimento. Inserire le righe seguenti nel file:

   ```
   <tns:user name="FTE User ID"
     serverUserId="Server User ID"
     serverPassword="Server Password">
   </tns:user>
   ```

   Quindi, modificare il valore degli attributi.

   – name è un'espressione regolare Java che corrisponde all'ID utente MQMD associato alla richiesta di trasferimento MFT

   – `serverUserId` è il valore passato al server di file del protocollo come ID utente di login. Se l'attributo `serverUserId` non viene specificato, viene utilizzato invece l'ID utente MQMD associato alla richiesta di trasferimento MFT

   – `serverPassword` è la password associata a `serverUserId`.

   L'attributo name può contenere un'espressione regolare Java . Il programma di associazione credenziali tenta di mettere in corrispondenza l'ID utente MQMD della richiesta di trasferimento MFT con questa espressione regolare. L'agent bridge di protocollo tenta di mettere in corrispondenza l'ID utente MQMD con l'espressione regolare nell'attributo del nome degli elementi < tns: user> nell'ordine in cui gli elementi esistono nel file. Quando viene trovata una corrispondenza, l'agent bridge di protocollo non cerca ulteriori corrispondenze. Se viene trovata una corrispondenza, i corrispondenti valori `serverUserId` e `serverPassword` vengono passati al server di file del protocollo come ID utente e password di collegamento. Le corrispondenze ID utente MQMD sono sensibili al maiuscolo / minuscolo.

   • Se il server file del protocollo è un server SFTP, è possibile utilizzare le chiavi pubbliche e private per autenticare l'utente che richiede il trasferimento. Inserire le seguenti righe nel file e modificare il valore degli attributi. L'elemento < tns: user> può contenere uno o più elementi < 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 è un'espressione regolare Java che corrisponde all'ID utente MQMD associato alla richiesta di trasferimento MFT
- serverUserId è il valore passato al server di file del protocollo come ID utente di login. Se l'attributo serverUserId non viene specificato, viene utilizzato invece l'ID utente MQMD associato alla richiesta di trasferimento MFT
- hostKey è la chiave prevista restituita dal server durante l'accesso
- key è la chiave privata di serverUserId
- keyPassword è la password per la chiave per generare le chiavi pubbliche
- associationName è un valore utilizzato per l'identificazione a scopo di traccia e registrazione

L'attributo name può contenere un'espressione regolare Java . Il programma di associazione credenziali tenta di mettere in corrispondenza l'ID utente MQMD della richiesta di trasferimento MFT con questa espressione regolare. L'agent bridge di protocollo tenta di mettere in corrispondenza l'ID utente MQMD con l'espressione regolare nell'attributo del nome degli elementi < tns: user> nell'ordine in cui gli elementi esistono nel file. Quando viene trovata una corrispondenza, l'agent bridge di protocollo non cerca ulteriori corrispondenze. Se viene trovata una corrispondenza, i corrispondenti valori serverUserId e key vengono utilizzati per autenticare l'utente MFT con il server di file del protocollo. Le corrispondenze ID utente MQMD sono sensibili al maiuscolo / minuscolo.

Per ulteriori informazioni sull'utilizzo delle chiavi private con un agent bridge di protocollo, consultare “Esempio: come configurare un agent bridge di protocollo per utilizzare le credenziali della chiave privata con un server SFTP UNIX” a pagina 305.

Nota: z/OS

Quando la richiesta di trasferimento viene scritta nella coda comandi, l'ID utente MQMD potrebbe essere convertito in maiuscolo se la coda comandi dell'agent di origine si trova su un sistema z/OS o IBM i . Di conseguenza, l'ID utente MQMD per lo stesso utente di origine potrebbe arrivare all'uscita delle credenziali nel caso originale o convertito in maiuscolo a seconda dell'agente di origine specificato nella richiesta di trasferimento. L'uscita di associazione credenziali predefinita esegue corrispondenze sensibili al maiuscolo / minuscolo rispetto all'ID utente MQMD fornito, che potrebbe essere necessario consentire nel file di associazione.

Riferimenti correlati

Formato file credenziali bridge di protocollo

Formato file delle proprietà bridge di protocollo

Espressioni regolari utilizzate da MFT

Associazione delle credenziali per un server di file utilizzando le classi di uscita

Se non si desidera utilizzare la funzione di associazione credenziali predefinita dell'agent bridge di protocollo, è possibile associare le credenziali utente in Managed File Transfer alle credenziali utente sul server di file scrivendo la propria uscita utente. Se si configurano le uscite utente di associazione credenziali, esse sostituiscono la funzione di associazione credenziali predefinita.

Informazioni su questa attività

Managed File Transfer fornisce un'uscita utente di esempio che esegue l'associazione delle credenziali utente. Per ulteriori informazioni, consultare “Utilizzo dell'uscita utente delle credenziali del bridge di protocollo di esempio” a pagina 304.

Un'uscita utente per la mappatura delle credenziali del bridge di protocollo deve implementare una delle seguenti interfacce:

- `com.ibm.wmqfte.exitroutine.api.ProtocolBridgeCredentialExit`, che consente a un agent bridge di protocollo di trasferire i file da e verso un server di file di protocollo predefinito
- `com.ibm.wmqfte.exitroutine.api.ProtocolBridgeCredentialExit2`, che consente di trasferire file da e verso più endpoint.

L'interfaccia `com.ibm.wmqfte.exitroutine.api.ProtocolBridgeCredentialExit2` contiene la stessa funzione di `com.ibm.wmqfte.exitroutine.api.ProtocolBridgeCredentialExit` e include anche la funzione estesa. Per ulteriori informazioni, vedi ProtocolBridgeCredentialExit.java interface e l'interfaccia ProtocolBridgeCredentialExit2.java.

Le uscite delle credenziali possono essere concatenate insieme in modo simile ad altre uscite utente. Le uscite vengono richiamati nell'ordine in cui vengono specificate utilizzando la proprietà `protocolBridgeCredentialConfiguration` del file delle proprietà agent. I metodi di inizializzazione vengono restituiti tutti separatamente e se uno o più restituisce un valore false, l'agent non viene avviato. L'errore viene riportato nella registrazione eventi dell'agent.

Viene restituito un solo risultato complessivo per i metodi ID mapMQUserdi tutte le uscite, come riportato di seguito:

- Se il metodo restituisce un valore USER_SUCCESSFULLY_MAPPED o USER_DENIED_ACCESS come codice risultato, questo valore è il risultato restituito e i metodi ID mapMQUserdelle uscite successive non vengono richiamati.
- Se il metodo restituisce un valore NO_MAPPING_FOUND come codice di risultato, viene richiamato il metodo ID mqMQUserdell'uscita successiva.
- Se non vi è alcuna uscita successiva, viene restituito il risultato NO_MAPPING_FOUND.
- Un codice di risultato generale USER_DENIED_ACCESS o NO_MAPPING_FOUND viene considerato come un errore di trasferimento dall'agent bridge.

Per eseguire l'uscita, completare la seguente procedura:

Procedura

1. Compilare la user exit delle credenziali del bridge di protocollo.
2. Creare un file JAR (Java archive) contenente l'uscita compilata e la relativa struttura del package.
3. Inserire il file JAR che contiene la classe di uscita nella directory `exits` dell'agent bridge. La directory si trova nella directory *MQ_DATA_PATH*`/mqft/config/`*coordination_queue_manager*`/agents/`*bridge_agent_name* .
4. Modificare il file delle proprietà dell'agent bridge di protocollo per includere la proprietà `protocolBridgeCredentialExitClasses`. Per il valore di questa proprietà, specificare un elenco separato da virgole di classi che implementano una routine di uscita delle credenziali del bridge di protocollo. Le classi di uscita vengono richiamate nell'ordine in cui sono specificate in questo elenco. Per ulteriori informazioni, vedere Il file MFT `agent.properties`.
5. Modificare il file delle proprietà dell'agent bridge di protocollo per includere:

   ```
   exitClassPath=IBM MQ
   installation_directory\mqft\config\configuration_queue_manager\agents\protocol_bridge_agent_n
   ame\exits\SampleCredentialExit.jar
   ```

   Il file `agent.properties` per un agente si trova nella directory *MQ_DATA_PATH*`/mqft/config/`*coordination_qmgr_name*`/agents/`*bridge_agent_name* .

   Se si modifica il file `agent.properties` , è necessario riavviare l'agent per rendere effettive le modifiche.
6. Facoltativamente, è possibile specificare la proprietà `protocolBridgeCredentialConfiguration` . Il valore specificato per questa proprietà viene passato come oggetto String al metodo initialize () delle classi di uscita specificate da `protocolBridgeCredentialExitClasses`. Per ulteriori informazioni, vedere Il file MFT `agent.properties`.

7. Avviare l'agent bridge di protocollo con il comando **fteStartAgent** .

Utilizzo dell'uscita utente delle credenziali del bridge di protocollo di esempio
Managed File Transfer fornisce un'uscita utente di esempio che esegue l'associazione delle credenziali utente.

Informazioni su questa attività

Un'uscita credenziali bridge di protocollo di esempio viene fornita nella directory *MQ_INSTALLATION_PATH*/mqft/samples/protocolBridge e nell'argomento Uscita utente credenziali bridge di protocollo di esempio. Questo esempio si basa sull'interfaccia com.ibm.wmqfte.exitroutine.api.ProtocolBridgeCredentialExit .

L'uscita SampleCredentialExit.java legge un file delle proprietà che associa gli ID utente MQMD associati alle richieste di trasferimento agli ID utente server e alle password server. L'ubicazione del file delle proprietà viene presa dalla proprietà dell'agent bridge di protocollo protocolBridgeCredentialConfiguration.

Per eseguire l'uscita utente di esempio, completare la seguente procedura:

Procedura

1. Compilare il file SampleCredentialExit.java .
2. Creare un file JAR che contiene l'uscita compilata e la relativa struttura del package.
3. Inserire il file JAR nella directory *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*bridge_agent_name*/exits .
4. Modificare il file *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*bridge_agent_name*/agent.properties in modo che contenga la riga:

   ```
   protocolBridgeCredentialExitClasses=SampleCredentialExit
   ```

5. Modificare il file delle proprietà dell'agent bridge di protocollo per includere:

   ```
   exitClassPath=IBM MQ
   installation_directory\mqft\config\configuration_queue_manager\agents\protocol_bridge_agent_n
   ame\exits\SampleCredentialExit.jar
   ```

   Il file agent.properties per un agente si trova nella directory *MQ_DATA_PATH*/mqft/config/*coordination_qmgr_name*/agents/*agent_name* .

   Se si modifica il file agent.properties , è necessario riavviare l'agent per rendere effettive le modifiche.
6. Creare un file delle proprietà delle credenziale (credentials.properties) nella directory *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*bridge_agent* e modificarlo per includere le voci nel formato:

   ```
   mqUserId=serverUserId,serverPassword
   ```

7. Modificare il file *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*bridge_agent_name*/agent.properties in modo che contenga la riga:

   ```
   protocolBridgeCredentialConfiguration=MQ_DATA_PATH/mqft/
   config/coordination_queue_manager/agents/bridge_agent_name/credentials.properties
   ```

   È necessario utilizzare il percorso assoluto del file credentials.properties .
8. Avviare l'agent bridge di protocollo utilizzando il comando **fteStartAgent** .

Concetti correlati
“Il bridge di protocollo” a pagina 291

Il bridge di protocollo consente alla rete Managed File Transfer (MFT) di accedere ai file memorizzati su un server di file esterno alla rete MFT , nel dominio locale o in un'ubicazione remota. Questo server di file può utilizzare i protocolli di rete FTP, FTPS o SFTP. Ogni server di file richiede almeno un agent dedicato. L'agent dedicato è noto come agent bridge di protocollo. Un agent bridge può interagire con più server di file.

Riferimenti correlati

Interfaccia ProtocolBridgeCredentialExit.java

Interfaccia ProtocolBridgeCredentialExit2.java

User exit delle credenziali del bridge di protocollo di esempio

Il file MFT `agent.properties`

fteCreateBridgeAgent (creazione e configurazione di un agent bridge di protocollo MFT)

Esempio: come configurare un agent bridge di protocollo per utilizzare le credenziali della chiave privata con un server SFTP UNIX

Questo esempio dimostra come generare e configurare il file `ProtocolBridgeCredentials.xml` . Questo esempio è un esempio tipico e i dettagli possono variare in base alla propria piattaforma, ma i principi rimangono gli stessi.

Informazioni su questa attività

Procedura

1. Generare una chiave pubblica e privata da utilizzare per l'autenticazione con il server SFTP.

   Ad esempio, su un sistema host Linux , è possibile utilizzare lo strumento **ssh-keygen**, fornito come parte del pacchettoopenssh', per creare la coppia di chiavi pubblica / privata.

   Per impostazione assunta, senza argomenti, il comando **ssh-keygen** richiede un'ubicazione e una passphrase per i due file chiave, che per impostazione predefinita sono i nomi:

   ```
   id_rsa      <-- Private key
   id_rsa.pub  <-- Public key
   ```

   Attenzione: Se si sta utilizzando il comando **ssh-keygen** da una versione recente di OpenSSH, come quella fornita con RHEL 8, il formato chiave utilizzato non è compatibile con l'agent bridge di protocollo e i tentativi di trasferimento al server SFTP hanno esito negativo con il messaggio:

   ```
   BFGBR0216E: Authentication to protocol server 'sftp.host.address' failed
   because of invalid private key.
   ```

   Per creare una chiave privata compatibile con queste versioni più recenti di OpenSSH, specifica il formato della chiave con il seguente argomento nel comando **ssh-keygen** :

   ```
   ssh-keygen -m PEM
   ```

   Il contenuto della chiave privata `id_rsa` ha quindi la prima e l'ultima riga di:

   ```
   -----BEGIN RSA PRIVATE KEY-----
   ... ... ...
   -----END RSA PRIVATE KEY-----
   ```

   compatibile con l'agent bridge di protocollo.
2. Copiare l'intero contenuto del file `id_rsa.pub` nel file `~/.ssh/authorized_keys` dell'utente SFTP sul server SFTP.

Assicurarsi che le autorizzazioni del file su questo file e la directory ~/.ssh siano impostati in modo appropriato per il server SFTP per consentire l'autenticazione della chiave. Queste autorizzazioni sono generalmente:

```
~/.ssh                    Mode 700
~/.ssh/authorized_keys    Mode 600
```

3. Managed File Transfer richiede un'impronta digitale ssh host generata utilizzando l'algoritmo MD5 . Eseguire uno dei seguenti comandi per ottenere l'impronta digitale ssh dell'host del server SFTP.
   - Per Red Hat® Enterprise Linux versione 6.x e precedenti e Linux Ubuntu 14.04, immetti il seguente comando:

     ```
     ssh-keygen -l -f /etc/ssh/ssh_host_rsa_key.pub
     ```

   - A cominciare da Red Hat Enterprise Linux 7.x, Linux Ubuntu 16.04 e SuSE Linux 12.4, il comando ssh - keygen genera, per impostazione predefinita, l'impronta digitale ssh utilizzando l'algoritmo SHA256 . Per generare l'impronta SSH utilizzando l'algoritmo MD5 , eseguire il seguente comando:

     ```
     ssh-keygen -l -E MD5 -f /etc/ssh/ssh_host_rsa_key.pub
     ```

     L'output del comando sarà simile al seguente esempio:

     ```
     2048 MD5:64:39:f5:49:41:10:55:d2:0b:81:42:5c:87:62:9d:27 no comment (RSA)
     ```

     Estrarre la parte esadecimale solo dell'output da utilizzare come hostKey nel file ProtocolBridgeCredentials.xml (consultare il passo “4” a pagina 306). Pertanto, in questo esempio, è possibile estrarre 64:39:f5:49:41:10:55:d2:0b:81:42:5c:87:62:9d:27.
4. Sul sistema agent bridge di protocollo, modificare il file ProtocolBridgeCredentials.xml . Sostituire i valori mostrati in corsivo nel seguente esempio con i propri valori:

```
<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>
```

dove:

- *Agent_name* è il nome dell'agent bridge di protocollo.
- *SFTP_host_name* è il nome del server SFTP come mostrato nel file ProtocolBridgeProperties.xml .
- *mq_User_ID* è l'ID utente MQMD associato alla richiesta di trasferimento.
- *SFTP_user_ID* è l'ID utente SFTP utilizzato nel passo 2. È il valore passato all'SFTP utilizzato come ID utente di login.
- *ssh_host_finger* è l'impronta digitale raccolta nel passo 3.
- *name* è un nome che è possibile specificare per essere utilizzato per scopi di traccia e registrazione.

- *pass_phrase* è la passphrase che hai fornito nel ssh - keygen nel passo 1.
- *Contenuto completo del file id_rsa* è il contenuto completo del file `id_rsa` generato dal passo 1. Per impedire un errore di connessione, assicurarsi di includere entrambe le seguenti voci:

```
-----BEGIN RSA PRIVATE KEY-----

-----END RSA PRIVATE KEY-----
```

È possibile aggiungere ulteriori chiavi duplicando l'elemento < tns: privatekey>.

5. Avviare l'agent bridge di protocollo se l'agent non è già avviato. In alternativa, l'agent bridge di protocollo esegue periodicamente il polling del file `ProtocolBridgeCredentials.xml` e apporta le modifiche.

Riferimenti correlati

Formato file credenziali bridge di protocollo

fteCreateBridgeAgent (creazione e configurazione di un agent bridge di protocollo MFT)

Il file MFT `agent.properties`

“Associazione delle credenziali per un server di file” a pagina 300

Associare le credenziali utente in Managed File Transfer alle credenziali utente sul server di file utilizzando la funzione di associazione credenziali predefinita dell'agent bridge di protocollo o scrivendo la propria uscita utente. Managed File Transfer fornisce un'uscita utente di esempio che esegue l'associazione delle credenziali utente.

Configurazione di un bridge di protocollo per un server FTPS

Configurare un server FTPS nello stesso modo in cui si configura un server FTP: creare un agent bridge per il server, definire le proprietà del server e associare le credenziali utente.

Informazioni su questa attività

Per configurare un server FTPS, completare la seguente procedura:

Procedura

1. Creare un agent bridge di protocollo per il server FTPS utilizzando il comando **`fteCreateBridgeAgent`** . I parametri applicabili a FTP sono applicabili anche a FTPS, ma ci sono anche tre parametri richiesti specifici per FTPS:
   a) Il parametro **`-bt`** . Specificare FTPS come valore di questo parametro.
   b) Il parametro **`-bts`** per il file truststore. Il comando presuppone che sia richiesta solo l'autenticazione del server ed è necessario specificare l'ubicazione del file truststore.

   Il formato esplicito del protocollo FTPS è configurato dal comando **`fteCreateBridgeAgent`** per impostazione predefinita, ma è possibile configurare il formato implicito modificando il file delle proprietà del bridge di protocollo. Il bridge di protocollo si connette sempre ai server FTPS in modalità passiva.

   Per ulteriori informazioni sul comando **`fteCreateBridgeAgent`** , consultare fteCreateBridgeAgent (creare e configurare un agent bridge di protocollo MFT).

   Per istruzioni su come creare i file truststore, consultare le informazioni su keytool nella documentazione diOracle keytool.
2. Definire le proprietà del server FTPS all'interno di un elemento <ftpsServer> nel file delle proprietà del bridge di protocollo: `ProtocolBridgeProperties.xml`. Per ulteriori informazioni, fare riferimento a “Definizione delle proprietà per i server di file del protocollo utilizzando il file ProtocolBridgeProperties.xml” a pagina 293. È anche possibile abilitare l'autenticazione client modificando il file delle proprietà del bridge di protocollo. Per i dettagli di tutte le opzioni di configurazione, vedere Formato file delle proprietà del bridge di protocollo.

3. Associare le credenziali utente in Managed File Transfer alle credenziali utente sul server FTPS utilizzando la funzione di associazione credenziali predefinita dell'agent bridge di protocollo o scrivendo la propria uscita utente. Per ulteriori informazioni, consultare "Associazione delle credenziali per un server di file" a pagina 300.
4. Per impostazione predefinita, il file truststore è configurato per avere il formato JKS; se si desidera modificare il formato, modificare il file delle proprietà del bridge di protocollo.

Esempio

Di seguito è riportata una voce di esempio per un server FTPS nel file delle proprietà del bridge di protocollo:

```
<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>
```

Operazioni successive

Per informazioni relative alle parti del protocollo FTPS supportate e non supportate, consultare Supporto server FTPS dal bridge di protocollo.

Concetti correlati

"Il bridge di protocollo" a pagina 291
Il bridge di protocollo consente alla rete Managed File Transfer (MFT) di accedere ai file memorizzati su un server di file esterno alla rete MFT , nel dominio locale o in un'ubicazione remota. Questo server di file può utilizzare i protocolli di rete FTP, FTPS o SFTP. Ogni server di file richiede almeno un agent dedicato. L'agent dedicato è noto come agent bridge di protocollo. Un agent bridge può interagire con più server di file.

Attività correlate

"Associazione delle credenziali per un server di file utilizzando il file ProtocolBridgeCredentials.xml" a pagina 300
Associare le credenziali utente in Managed File Transfer alle credenziali utente sul file server utilizzando la funzione di associazione credenziali predefinita dell'agent bridge di protocollo. Managed File Transfer fornisce un file XML che è possibile modificare per includere le informazioni sulle credenziali.

"Definizione delle proprietà per i server di file del protocollo utilizzando il file ProtocolBridgeProperties.xml" a pagina 293
Definire le proprietà di uno o più server di file di protocollo a cui si desidera trasferire i file utilizzando il file `ProtocolBridgeProperties.xml` , fornito da Managed File Transfer nella directory di configurazione dell'agent.

Riferimenti correlati

fteCreateBridgeAgent (creazione e configurazione di un agent bridge di protocollo MFT)
Formato file credenziali bridge di protocollo
Formato file delle proprietà bridge di protocollo
Supporto server FTPS dal bridge di protocollo

Scenari ed esempi per limitare il numero di trasferimenti file a singoli server di file

Come funziona l'agent bridge di protocollo revisionato con gli attributi **maxActiveDestinationTransfers** e **failTransferWhenCapacityReached** , insieme ad alcuni esempi.

Scenari che mostrano il funzionamento dell'agent bridge di protocollo basato sul valore `maxActiveDestinationTransfers`

Scenario 1

Il file `ProtocolBridgeProperties.xml` per un agent bridge di protocollo contiene due definizioni di server di file:

- Non è stato impostato l'attributo **maxActiveDestinationTransfers** globale.
- Non è stato impostato l'attributo **maxActiveDestinationTransfers** su `fileServerA` e `FileServerB`.
- L'attributo **maxDestinationTransfers** dell'agent bridge di protocollo è stato impostato sul valore predefinito.

Se l'attributo **maxDestinationTransfers** dell'agent bridge di protocollo è stato impostato sul valore predefinito di 25,:

- L'agent di destinazione inizia l'elaborazione di due trasferimenti gestiti in `fileServerA`.
- Entrambi i trasferimenti sono completi.

  A questo punto, il client si rende conto che `fileServerA` non è riuscito e imposta i seguenti valori per `fileServerA` nel file `ProtocolBridgeProperties.xml` :

  maxActiveDestinationTransfers = 0
  failTransferWhenCapacityReached = vero
- Un altro trasferimento arriva per `fileServerA` e alcuni per `fileServerB`:

  In base alle proprietà impostate nel passo precedente, il trasferimento gestito a `fileServerA` viene rifiutato e contrassegnato come non riuscito, mentre il trasferimento per `fileServerB` viene gestito nel flusso esistente standard.
- Dopo qualche tempo, il client scopre che `fileServerA` è di nuovo in esecuzione, quindi il client rimuove o commenta il valore aggiunto precedentemente in `ProtocolBridgeProperties.xml`. Arriva un nuovo trasferimento gestito per `fileServerA` e viene gestito nel flusso esistente standard.

Scenario 2

- È stato impostato l'attributo **maxActiveDestinationTransfers** per un server di file e non l'attributo **failTransferWhenCapacityReached** .
- L'agent bridge di protocollo agisce come agent di destinazione per questo numero di trasferimenti gestiti al server di file.
- Il valore dell'attributo **maxActiveDestinationTransfers** è ridotto di 1.

L'agent bridge di protocollo aggiorna dinamicamente la relativa configurazione e imposta **maxActiveDestinationTransfers** sul nuovo valore mentre è ancora attivo. I trasferimenti gestiti in corso non sono interessati da questo aggiornamento e possono essere completati.

Scenario 3

Il file ProtocolBridgeProperties.xml per un agent bridge di protocollo contiene due definizioni di file server:

- Non è stato impostato l'attributo **maxActiveDestinationTransfers** globale.
- L'attributo **failTransferWhenCapacityReached** non è stato impostato.
- **maxActiveDestinationTransfers** è stato impostato su 1 su `fileServerA`.
- L'attributo **maxActiveDestinationTransfers** non è stato impostato su `fileServerB`.

Se l'agent bridge di protocollo ha l'attributo **maxDestinationTransfers** impostato su 5:

- Il numero massimo di trasferimenti di destinazione attivi dall'agent bridge di protocollo a `fileServerA` è 1 (sebbene l'agent di destinazione abbia 5 slot di trasferimento di destinazione, solo 1 può essere utilizzato per i trasferimenti gestiti a `fileServerA`).

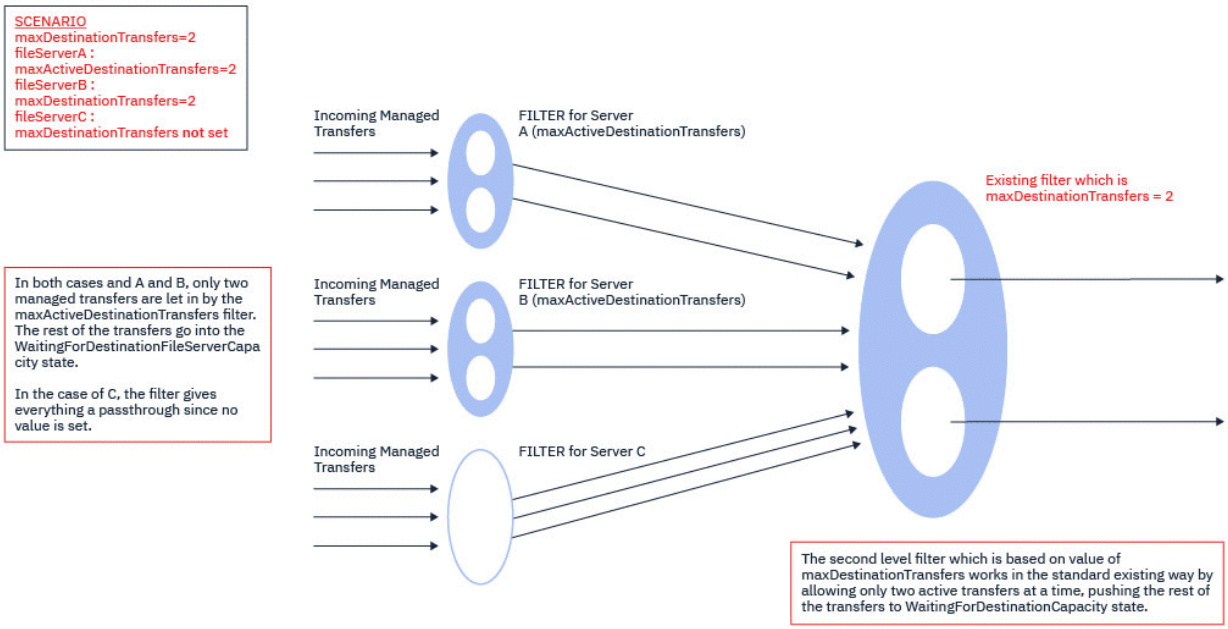
  Ciò è utile quando `fileServerA` non riesce. Una volta che `fileServerA` è di nuovo in esecuzione, il valore di **maxActiveDestinationTransfers** può essere aumentato a 5 per consentire la piena capacità dei trasferimenti di destinazione consentiti.
- Il numero massimo di trasferimenti di destinazione attivi dall'agent bridge di protocollo a `fileServerB` è 5.

  Poiché **maxActiveDestinationTransfers** non è impostato per questo server di file, l'agent bridge di protocollo può utilizzare tutti i 5 slot di trasferimento di destinazione per i trasferimenti gestiti.

Scenario 4

Nel seguente diagramma:

- Si è impostato l'attributo **maxDestinationTransfers** su 2 nel file `agent.properties` .
- Il **maxActiveDestinationTransfers** è stato impostato su 2 su `fileServerA`.
- L'attributo **maxActiveDestinationTransfers** è stato impostato su 2 su `fileServerB`.
- L'attributo **maxActiveDestinationTransfers** non è stato impostato su `fileServerC`.



Come mostra il diagramma, gli attributi **maxActiveDestinationTransfers** e **maxDestinationTransfers** sono indipendenti l'uno dall'altro.

Vengono controllati i valori per **maxActiveDestinationTransfers** per ciascuno dei server. In base a questo valore, i trasferimenti possono continuare o essere spostati allo stato **WaitingForDestinationFileServerCapacity** .

I trasferimenti consentiti passano quindi attraverso il flusso standard esistente di verifica rispetto a **maxDestinationTransfers**.

Scenario 5

Attenzione: Prestare attenzione quando si impostano i valori degli attributi **maxActiveDestinationTransfers** , poiché è necessario tenere presente il valore dell'attributo **maxDestinationTransfers** .

Se non si esegue questa operazione, può verificarsi una situazione come descritto nel testo seguente:

- Non è stato impostato un valore per l'attributo **maxActiveDestinationTransfers** globale.
- È stato impostato il valore **maxDestinationTransfers**= 2 nel file `agent.properties` .
- È stato impostato il valore **maxActiveDestinationTransfers**= 2 su `fileServerA`.
- Non è stato impostato un valore per **maxActiveDestinationTransfers** su `fileServerB`.

Si supponga che si verifichi la seguente sequenza di eventi:

- L'agent bridge di protocollo riceve una richiesta di trasferimento di un file a `fileServerA`. L'agent bridge di protocolli non sta attualmente eseguendo alcuna operazione, quindi accetta questa richiesta di trasferimento gestito.

  Gli slot di trasferimento ora hanno il seguente aspetto:

  - Trasferimenti di destinazione: 1
  - Trasferimenti di destinazione per `fileServerA`: 1
  - Trasferimenti di destinazione per `fileServerB`: 0
- A questo punto, l'agent bridge di protocollo riceve un'ulteriore richiesta di agire come agent di destinazione per un trasferimento gestito che coinvolge `fileServerA`. Ancora una volta, accetta questa richiesta e quindi gli slot di trasferimento si assomigliano a questo:

  - Trasferimenti di destinazione: 2
  - Trasferimenti di destinazione per `fileServerA`: 2
  - Trasferimenti di destinazione per `fileServerB`: 0

I due slot `Destination Transfer` nell'agent sono ora occupati e quindi l'agent non può partecipare a ulteriori trasferimenti gestiti fino a quando uno dei trasferimenti a `fileServerA` non è terminato.

- Un breve periodo di tempo dopo, `fileServerA` non riesce, causando il ripristino dei due trasferimenti gestiti. Gli slot `Destination transfer` utilizzati da questi trasferimenti gestiti rimangono in uso durante questo tempo.
- Successivamente, l'agente bridge di protocollo riceve una richiesta di trasferimento di un file a `fileServerB`. C'è uno spazio per questo trasferimento negli slot `Destination Transfers for fileServerB` , tuttavia, vengono utilizzati tutti gli slot `Destination Transfer` per l'agent e quindi il trasferimento viene inserito nel backlog in modo che possa essere ritentato in un secondo momento.

Di conseguenza, il trasferimento a `fileServerB` viene bloccato fino a che almeno uno dei trasferimenti a `fileServerA` non ha completato e rilasciato il suo slot `Destination Transfer` .

Per evitare che questa situazione si verifichi:

- Impostare il valore di **maxActiveDestinationTransfers** sui server di file in modo che sia inferiore al valore **maxDestinationTransfers** , in modo che rimangano gli slot liberi.
- In alternativa, distribuire in modo uniforme il valore dell'attributo **maxActiveDestinationTransfers** tra tutti i server endpoint.

Comportamento dell'agent bridge di protocollo basato sui valori dell'attributo `maxActiveDestinationTransfers`

Nota: In tutti i casi di errore elencati nella seguente tabella, se l'attributo di **`maxActiveDestinationTransfers`** è impostato su un valore non valido, l'agent bridge di protocollo presuppone che questo attributo non sia impostato.

maxActiveDestinationTransfers	Valore di esempio	Descrizione
Non specificato	Non specificato	I trasferimenti vanno come al solito. Non esiste alcun limite al numero di trasferimenti per l'endpoint * ftp *.
Specificata	0	Nessun trasferimento consentito a questo endpoint * ftp * specifico.
Valore negativo	-1	Errore registrato in output0.log Value -1 non è valido per un numero intero non negativo. L'agent bridge di protocollo presuppone che l'attributo non sia impostato.
Valore non intero	abc	Errore registrato in output0.log Valore abc non valido per un numero intero. L'agent bridge di protocollo presuppone che l'attributo non sia impostato.
Vuoto	""	Il valore ' ' dell'attributo maxActiveDestinationTransfers non è valido per un intero non negativo.
Specificata	5	Consente di eseguire solo cinque trasferimenti attivi in qualsiasi momento per questo endpoint * ftp *. I trasferimenti eccessivi vengono ritentati o rifiutati, in base al valore dell'attributo failTransferWhenCapacityReached.

Funzionamento dell'agent bridge di protocollo per la combinazione di attributi `maxActiveDestinationTransfers` e `failTransferWhenCapacityReached`

failTransferWhenCapacityValore raggiunto	Valore maxActiveDestinationTransfers	Risultato
No	3	Sono consentiti tre trasferimenti attivi a questo server endpoint. Qualsiasi altro trasferimento viene ritentato.
Vero	3	Sono consentiti tre trasferimenti attivi a questo server endpoint. Eventuali altri trasferimenti vengono rifiutati e contrassegnati come non riusciti.
Non specificato	3	Viene considerato il valore predefinito **false** per

failTransferWhenCapacityValore raggiunto	Valore maxActiveDestinationTransfers	Risultato
		failTransferWhenCapacityReached. Il risultato è che tre trasferimenti attivi sono consentiti a questo server endpoint. Qualsiasi altro trasferimento viene ritentato.
Valori diversi dal valore booleano	Specificata	Errore registrato in output.log. Il valore specificato per failTransferWhenCapacityRaggiunto non è un valore booleano. Viene considerato il valore predefinito per failTransferWhenCapacityReached.

Funzionamento dell'agent bridge di protocollo per la combinazione di attributi `maxDestinationTransfers` e `failTransferWhenCapacityReached`

failTransferWhenCapacityValore raggiunto	maxDestinationValore trasferimenti	Risultato
Vero	10	Quando il numero di trasferimenti attivi simultanei raggiunge 10, l'11^{esimo} trasferimento gestito non riesce dall'agent bridge di protocollo.
No	10	Comportamento esistente. Quando il numero di trasferimenti attivi simultanei raggiunge 10, l'11^{esimo} trasferimento gestito viene accodato in attesa che venga liberato uno slot.
Non specificato	10	Comportamento esistente

Messaggi di errore

Messaggio esistente:

BFGS0082I
Viene registrato nel file output0.log dell'agent di origine quando l'agent bridge di protocollo rifiuta il trasferimento, quando l'agent bridge di protocollo sta già eseguendo il numero massimo di trasferimenti definiti nell'attributo **`maxDestinationTransfers`** .

Nuovi messaggi:

BFGSS0085I
Viene registrato nel file output0.log dell'agente di origine quando l'agente bridge di protocollo rifiuta e ritenta un trasferimento gestito,

BFGSS0086I
Viene registrato nel file output0.log dell'agente di origine quando l'agente bridge di protocollo rifiuta e ritenta un trasferimento gestito e l'elemento di destinazione non include il nome del server di file

BFGSS0084E
Viene registrato nel file Explorer e audit.xml quando l'agent bridge di protocollo rifiuta, per il superamento del numero massimo di trasferimenti simultanei specificato nell'attributo **maxActiveDestinationTransfers** e contrassegna un trasferimento gestito come non riuscito.

BFGSS0087E
Viene registrato nel file Explorer e audit.xml quando l'agent bridge di protocollo rifiuta, per aver superato il numero massimo di trasferimenti di destinazione specificati nell'attributo **maxActiveDestinationTransfers** e contrassegna un trasferimento gestito come non riuscito.

BFGSS0088W
Viene registrato in output0.log, quando il valore dell'attributo **maxActiveDestinationTransfers** supera il valore dell'attributo **maxDestinationTransfers** .

BFGSS0089I
Viene registrato nel file output0.log dell'agent bridge di protocollo di destinazione, quando sta utilizzando un agent di origine che non si trova in IBM MQ 9.3.0o in una versione successiva.

Concetti correlati

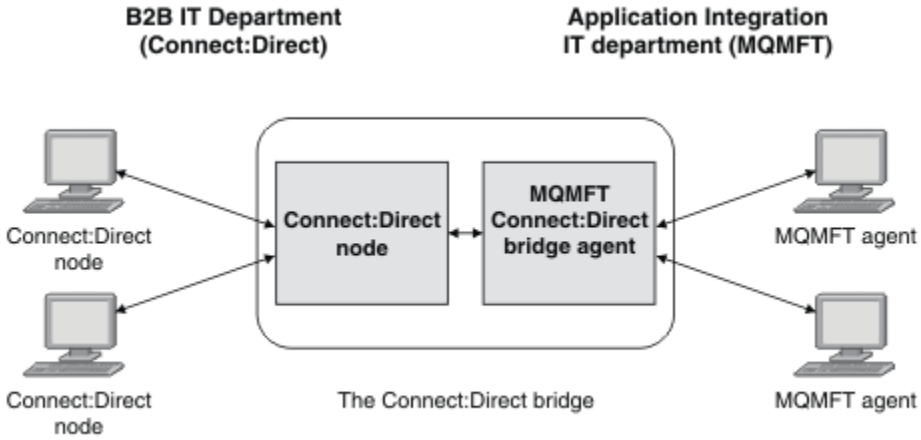
"Il bridge di protocollo" a pagina 291
Il bridge di protocollo consente alla rete Managed File Transfer (MFT) di accedere ai file memorizzati su un server di file esterno alla rete MFT , nel dominio locale o in un'ubicazione remota. Questo server di file può utilizzare i protocolli di rete FTP, FTPS o SFTP. Ogni server di file richiede almeno un agent dedicato. L'agent dedicato è noto come agent bridge di protocollo. Un agent bridge può interagire con più server di file.

Attività correlate

"Definizione delle proprietà per i server di file del protocollo utilizzando il file ProtocolBridgeProperties.xml" a pagina 293
Definire le proprietà di uno o più server di file di protocollo a cui si desidera trasferire i file utilizzando il file `ProtocolBridgeProperties.xml` , fornito da Managed File Transfer nella directory di configurazione dell'agent.

Il bridge Connect:Direct

È possibile trasferire i file a e da una rete IBM Sterling Connect:Direct esistente. Utilizzare il bridge Connect:Direct , che è un componente di Managed File Transfer, per trasferire i file tra MFT e IBM Sterling Connect:Direct.



Il diagramma mostra un ponte MFT Connect:Direct tra due dipartimenti, il dipartimento IT B2B e il reparto IT di Application Integration. Il reparto di IT B2B utilizza Connect:Direct per trasferire i file da e verso i business partner dell'azienda. Il reparto IT di Application Integration utilizza IBM MQ come propria infrastruttura di messaggistica e quindi ha recentemente scelto Managed File Transfer come propria soluzione di trasferimento file.

Utilizzando il bridge MFT Connect:Direct , i due dipartimenti possono trasferire file tra la rete Connect:Direct nel dipartimento IT B2B e la rete MFT nel dipartimento IT di Application Integration. Il bridge di Connect:Direct è un componente di Managed File Transfer, che include un agente MFT che comunica con un nodo Connect:Direct . L'agent MFT è dedicato ai trasferimenti con il nodo Connect:Direct ed è noto come agent bridge Connect:Direct .

Il bridge di Connect:Direct è disponibile come parte dei componenti Service e Agent di Managed File Transfere può essere utilizzato per le seguenti attività:

1. Utilizzare i comandi Managed File Transfer per avviare un trasferimento di un file, o di più file, da un agent MFT a un nodo Connect:Direct .
2. Utilizzare i comandi Managed File Transfer per avviare un trasferimento di un file, o più file, da un nodo Connect:Direct a un agent MFT .
3. Utilizzare i comandi Managed File Transfer per avviare un trasferimento file che avvia un processo Connect:Direct definito dall'utente.
4. Utilizzare il processo Connect:Direct per inoltrare una richiesta di trasferimento file MFT .

Un bridge Connect:Direct può trasferire i file solo verso o da nodi Connect:Direct . Il bridge Connect:Direct può trasferire i file da o verso il proprio file system locale solo come parte di un trasferimento inoltrato da un processo Connect:Direct .

z/OS È possibile utilizzare il bridge Connect:Direct per il trasferimento a o da un dataset che si trova su un nodo Connect:Direct su un sistema z/OS . Ci sono alcune differenze nel comportamento rispetto ai trasferimenti di dataset che coinvolgono solo gli agenti Managed File Transfer . Per ulteriori informazioni, consultare z/OS Trasferimento di dataset a e da nodi Connect:Direct.

Piattaforme supportate

Il bridge Connect:Direct è costituito da un agent bridge MFT Connect:Direct e da un nodo Connect:Direct . L'agente è supportato su Windows e Linux per x86-64. Il nodo è supportato sulle piattaforme supportate per IBM Sterling Connect:Direct per Windows e IBM Sterling Connect:Direct per UNIX. Per istruzioni sulla creazione dell'agent bridge Connect:Direct e sulla configurazione di un nodo Connect:Direct con cui l'agent deve comunicare, consultare Configurazione del bridge Connect:Direct.

Il bridge Connect:Direct può trasferire i file da e verso i nodi Connect:Direct in esecuzione come parte di un'installazione del servizio Connect:Direct per Windows o Connect:Direct per UNIX o Connect:Direct per z/OS . Per dettagli sulle versioni di Connect:Direct supportate, consultare la pagina Web Requisiti di sistema per IBM MQ.

L'agent e il nodo che costituiscono il bridge Connect:Direct devono essere sullo stesso sistema o avere accesso allo stesso file system, ad esempio tramite un montaggio NFS condiviso. Questo file system viene utilizzato per memorizzare temporaneamente i file durante i trasferimenti di file che coinvolgono il bridge Connect:Direct, in una directory definita dal parametro **cdTmpDir**. L'agent bridge Connect:Direct e il nodo bridge Connect:Direct devono essere in grado di raggiungere questa directory utilizzando lo stesso nome percorso. Ad esempio, se l'agent e il nodo si trovano su sistemi Windows separati, i sistemi devono utilizzare la stessa lettera di unità per il montaggio del file system condiviso. Le seguenti configurazioni consentono all'agent e al nodo di utilizzare lo stesso nome percorso:

- L'agent e il nodo si trovano sullo stesso sistema, che è in esecuzione in Windows o Linux per x86-64
- L'agent si trova su Linux per x86-64 e il nodo è su AIX
- L'agent si trova su un sistema Windows e il nodo si trova su un altro sistema Windows

Le seguenti configurazioni non consentono all'agent e al nodo di utilizzare lo stesso nome percorso:

- L'agent si trova su Linux per x86-64 e il nodo è su Windows
- L'agent si trova su Windows e il nodo su UNIX

Considerare questa limitazione quando si pianifica l'installazione del bridge Connect:Direct.

Concetti correlati

“Ripristino e riavvio per i trasferimenti da e verso i nodi Connect:Direct” a pagina 323
Managed File Transfer potrebbe non essere in grado di connettersi al tuo nodo IBM Sterling Connect:Direct durante un trasferimento; ad esempio, se il nodo diventa non disponibile. Managed File Transfer tenta di ripristinare il trasferimento oppure il trasferimento ha esito negativo e viene prodotto un messaggio di errore.

“Inoltro di un processo Connect:Direct definito dall'utente da una richiesta di trasferimento file” a pagina 324
È possibile inviare una richiesta di trasferimento per un trasferimento che passa attraverso l'agent bridge Connect:Direct che richiama un processo Connect:Direct definito dall'utente come parte del trasferimento file.

“Utilizzo dei processi Connect:Direct per inoltrare richieste di trasferimento Managed File Transfer” a pagina 328
È possibile inviare una richiesta di trasferimento all'agent bridge Connect:Direct da un processo Connect:Direct . Managed File Transfer fornisce comandi che possono essere richiamati da un'istruzione **RUN TASK** in un processo Connect:Direct .

Attività correlate

Configurazione del bridge Connect:Direct

“Trasferimento di un file in un nodo Connect:Direct” a pagina 316
È possibile trasferire un file da un agent Managed File Transfer a un nodo Connect:Direct utilizzando il bridge Connect:Direct . Specificare un nodo Connect:Direct come destinazione del trasferimento specificando l'agent bridge Connect:Direct come agent di destinazione e specificando il file di destinazione nel formato *connect_direct_node_name*:*file_path*.

“Trasferimento di un file da un nodo Connect:Direct” a pagina 317
Puoi trasferire un file da un nodo Connect:Direct a un Managed File Transfer Agent utilizzando il bridge Connect:Direct . È possibile specificare un nodo di Connect:Direct come origine del trasferimento specificando l'agent bridge Connect:Direct come agent di origine e specificando la specifica di origine nel formato *connect_direct_node_name*:*file_path*.

“Trasferimento di più file in un nodo Connect:Direct” a pagina 319
È possibile trasferire più file da un Managed File Transfer Agent ad un nodo Connect:Direct utilizzando il bridge Connect:Direct . Per utilizzare un nodo Connect:Direct come destinazione del trasferimento di più file, specificare l'agent bridge Connect:Direct come agent di destinazione e specificare la directory di destinazione nel formato *connect_direct_node_name*:*directory_path*.

“Transferring multiple files from a Connect:Direct node” a pagina 320
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*.

“Trasferimento di più file in Connect:Direct utilizzando i caratteri jolly” a pagina 321
Per trasferire più file da un agent Managed File Transfer a un nodo Connect:Direct , utilizzare il bridge Connect:Direct . È possibile utilizzare i caratteri jolly nella specifica di origine fornita al comando **fteCreateTransfer** . Come per tutti i trasferimenti Managed File Transfer che implicano caratteri jolly, solo l'ultima parte del percorso file può contenere un carattere jolly. Ad esempio, `/abc/def*` è un percorso file valido e `/abc*/def` non è valido.

Risoluzione dei problemi relativi al bridge Connect:Direct

Riferimenti correlati

fteCreateCDAgent: crea un agent bridge Connect:Direct

Limitazioni dell'agent bridge Connect:Direct

Trasferimento di un file in un nodo Connect:Direct

È possibile trasferire un file da un agent Managed File Transfer a un nodo Connect:Direct utilizzando il bridge Connect:Direct . Specificare un nodo Connect:Direct come destinazione del trasferimento

specificando l'agent bridge Connect:Direct come agent di destinazione e specificando il file di destinazione nel formato *connect_direct_node_name*:*file_path*.

Prima di iniziare

Prima di trasferire un file, è necessario configurare il bridge Connect:Direct , che è un componente di Managed File Transfer. Per ulteriori informazioni, consultare Configurazione del bridge Connect:Direct.

Informazioni su questa attività

In questo esempio, l'agent bridge Connect:Direct è denominato CD_BRIDGE. L'agent di origine è denominato FTE_AGENT e può essere una qualsiasi versione di WMQFTE. Il nodo di destinazione Connect:Direct è denominato CD_NODE1. Il file da trasferire si trova nel percorso file `/home/helen/file.log` sul sistema in cui si trova FTE_AGENT. Il file viene trasferito al percorso file `/files/data.log` sul sistema su cui è in esecuzione CD_NODE1 .

Procedura

1. Utilizzare il comando fteCreateTransfer con il valore per il parametro **-df** (file di destinazione) nel formato *connect_direct_node_name*:*file_path* e il valore del parametro **-da** (agent di destinazione) specificato come nome dell'agent bridge Connect:Direct .

   Nota: Il nodo Connect:Direct specificato da *connect_direct_node_name* è il nodo a cui si desidera trasferire il file, non il nodo Connect:Direct che opera come parte del bridge Connect:Direct .

   ```
   fteCreateTransfer -sa FTE_AGENT -da CD_BRIDGE
     -df CD_NODE1:/files/data.log /home/helen/file.log
   ```

   Per ulteriori informazioni, consultare **fteCreateTransfer**: avviare un nuovo trasferimento file.
2. L'agent di origine FTE_AGENT trasferisce il file all'agent bridge Connect:Direct CD_BRIDGE. Il file viene memorizzato temporaneamente sul sistema su cui è in esecuzione l'agent bridge Connect:Direct , nell'ubicazione definita dalla proprietà dell'agent Dir cdTmp. L'agent bridge Connect:Direct trasferisce il file al nodo Connect:Direct CD_NODE1.

Concetti correlati

“Il bridge Connect:Direct” a pagina 314
È possibile trasferire i file a e da una rete IBM Sterling Connect:Direct esistente. Utilizzare il bridge Connect:Direct , che è un componente di Managed File Transfer, per trasferire i file tra MFT e IBM Sterling Connect:Direct.

Attività correlate

“Trasferimento di un file da un nodo Connect:Direct” a pagina 317
Puoi trasferire un file da un nodo Connect:Direct a un Managed File Transfer Agent utilizzando il bridge Connect:Direct . È possibile specificare un nodo di Connect:Direct come origine del trasferimento specificando l'agent bridge Connect:Direct come agent di origine e specificando la specifica di origine nel formato *connect_direct_node_name*:*file_path*.

Riferimenti correlati

Il file MFT `agent.properties`

Trasferimento di un file da un nodo Connect:Direct

Puoi trasferire un file da un nodo Connect:Direct a un Managed File Transfer Agent utilizzando il bridge Connect:Direct . È possibile specificare un nodo di Connect:Direct come origine del trasferimento specificando l'agent bridge Connect:Direct come agent di origine e specificando la specifica di origine nel formato *connect_direct_node_name*:*file_path*.

Prima di iniziare

Prima di trasferire un file, è necessario configurare il bridge Connect:Direct , che è un componente di Managed File Transfer. Consultare Configurazione del bridge Connect:Direct.

Informazioni su questa attività

In questo esempio, l'agent bridge Connect:Direct è denominato CD_BRIDGE. L'agent di destinazione è denominato FTE_AGENT e può essere una qualsiasi versione di Managed File Transfer. Il nodo Connect:Direct di origine è denominato CD_NODE1. Il file da trasferire si trova nel percorso file `/home/brian/in.file` sul sistema in cui si trova CD_NODE1 . Il file viene trasferito al percorso file `/files/out.file` sul sistema su cui è in esecuzione FTE_AGENT.

Procedura

Utilizzare il comando **fteCreateTransfer** con il valore per la specifica di origine nel formato *connect_direct_node_name*:*file_path* e il valore del parametro **-sa** specificato come nome dell'agente bridge Connect:Direct .

Nota: Il nodo Connect:Direct , specificato da *connect_direct_node_name* , è il nodo da cui si desidera trasferire il file e non il nodo Connect:Direct che opera come parte del bridge Connect:Direct . Ad esempio:

```
fteCreateTransfer -sa CD_BRIDGE -da FTE_AGENT
                  -df /files/out.file CD_NODE1:/home/brian/in.file
```

Per ulteriori informazioni, consultare **fteCreateTransfer**: avviare un nuovo trasferimento file.

Risultati

L'agent bridge Connect:Direct CD_BRIDGE richiede il file dal nodo Connect:Direct CD_NODE1. Il nodo Connect:Direct invia il file al bridge Connect:Direct . Durante il trasferimento del file dal nodo Connect:Direct , il bridge Connect:Direct memorizza il file temporaneamente nell'ubicazione definita dalla proprietà dell'agente `cdTmpDir` . Una volta terminato il trasferimento del file dal nodo Connect:Direct al bridge Connect:Direct , il bridge Connect:Direct invia il file all'agent di destinazione FTE_AGENT ed elimina il file dall'ubicazione temporanea.

Concetti correlati

“Il bridge Connect:Direct” a pagina 314

È possibile trasferire i file a e da una rete IBM Sterling Connect:Direct esistente. Utilizzare il bridge Connect:Direct , che è un componente di Managed File Transfer, per trasferire i file tra MFT e IBM Sterling Connect:Direct.

Riferimenti correlati

Il file MFT `agent.properties`

z/OS Trasferimento di un dataset a un nodo Connect:Direct su z/OS

È possibile trasferire un dataset da un agent Managed File Transfer su z/OS a un nodo Connect:Direct su z/OS utilizzando un bridge Connect:Direct ubicato su un sistema Windows o Linux .

Prima di iniziare

Prima di trasferire un file, è necessario configurare il bridge Connect:Direct , che è un componente di Managed File Transfer. Consultare Configurazione del bridge Connect:Direct.

Informazioni su questa attività

In questo esempio, il parametro **-df** viene usato per specificare la destinazione del trasferimento. Il parametro **-df** è valido per l'utilizzo quando l'agent di origine del trasferimento è una qualsiasi versione di Managed File Transfer. È possibile utilizzare il parametro **-ds** . L'agent di origine è denominato FTE_ZOS1 ed è un agent Managed File Transfer . L'agent bridge Connect:Direct è denominato CD_BRIDGE e si trova su un sistema Linux . Il nodo di destinazione Connect:Direct è denominato CD_ZOS2. Sia l'agent di origine che il nodo Connect:Direct di destinazione si trovano su sistemi z/OS . Il dataset da trasferire si trova in `//FTEUSER.SOURCE.LIB` sul sistema in cui si trova FTE_ZOS1 . Il dataset viene trasferito al dataset `//CDUSER.DEST.LIB` sul sistema in cui si trova CD_ZOS2 .

Procedura

1. Utilizzare il comando di trasferimento fteCreatecon il valore per il parametro **-df** nel formato: *connect_direct_node_name*:*data_set_name*;*attributes* e il valore del parametro **-da** (agent di destinazione) specificato come nome dell'agent bridge Connect:Direct .

   Il nodo Connect:Direct specificato da *connect_direct_node_name* è il nodo a cui si desidera trasferire il dataset, non il nodo Connect:Direct che opera come parte del bridge Connect:Direct .

   Il nome dataset specificato da *nome_serie_dati* deve essere assoluto, non relativo. Connect:Direct non antepone il nome del dataset al nome dell'utente.

   ```
   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'
   ```

   Per ulteriori informazioni, consultare **fteCreateTransfer**: avviare un nuovo trasferimento file.
2. L'agent di origine FTE_ZOS1 trasferisce i dati nel dataset all'agent bridge Connect:Direct CD_BRIDGE. I dati vengono memorizzati temporaneamente come file flat sul sistema su cui è in esecuzione l'agente bridge Connect:Direct , nell'ubicazione definita dalla proprietà dell'agente Dir cdTmp. L'agent bridge Connect:Direct trasferisce i dati al nodo Connect:Direct CD_ZOS2. Una volta completato il trasferimento, il file flat viene eliminato dal sistema su cui è in esecuzione l'agent bridge Connect:Direct .

Concetti correlati

“Il bridge Connect:Direct” a pagina 314
È possibile trasferire i file a e da una rete IBM Sterling Connect:Direct esistente. Utilizzare il bridge Connect:Direct , che è un componente di Managed File Transfer, per trasferire i file tra MFT e IBM Sterling Connect:Direct.

Attività correlate

Trasferimento di dataset da e verso nodi Connect:Direct

Riferimenti correlati

Proprietà BPXWDYN che non devono essere utilizzate con MFT

Trasferimento di più file in un nodo Connect:Direct

È possibile trasferire più file da un Managed File Transfer Agent ad un nodo Connect:Direct utilizzando il bridge Connect:Direct . Per utilizzare un nodo Connect:Direct come destinazione del trasferimento di più file, specificare l'agent bridge Connect:Direct come agent di destinazione e specificare la directory di destinazione nel formato *connect_direct_node_name*:*directory_path*.

Prima di iniziare

Prima di trasferire i file, configurare il bridge Connect:Direct , che è un componente di Managed File Transfer. Consultare Configurazione del bridge Connect:Direct.

Informazioni su questa attività

In questo esempio, l'agent di origine è denominato FTE_AGENT. L'agent bridge Connect:Direct è denominato CD_BRIDGE. Il nodo di destinazione Connect:Direct è denominato CD_NODE1. I file da trasferire sono `/home/jack/data.log`, `/logs/log1.txte` `/results/latest` sul sistema in cui si trova FTE_AGENT. I file vengono trasferiti alla directory `/in/files` sul sistema su cui è in esecuzione CD_NODE1 .

Procedura

Utilizzare il comando di trasferimento fteCreatecon il valore per il parametro **-dd** (directory di destinazione) nel formato *connect_direct_node_name*:*directory_path*. Specificare il valore del parametro **-da** (agent di destinazione) come nome dell'agent bridge Connect:Direct .

Nota: Il nodo Connect:Direct specificato da *connect_direct_node_name* è il nodo in cui si desidera trasferire i file, non il nodo Connect:Direct che opera come parte del bridge Connect:Direct .

```
fteCreateTransfer -sa FTE_AGENT -da CD_BRIDGE
                  -dd CD_NODE1:/in/files /home/jack/data.log
                  /logs/log1.txt /results/latest
```

Per ulteriori informazioni, consultare **fteCreateTransfer**: avviare un nuovo trasferimento file.

Risultati

L'agent di origine FTE_AGENT trasferisce il primo file all'agent bridge Connect:Direct CD_BRIDGE. L'agent bridge Connect:Direct memorizza temporaneamente il file nell'ubicazione definita dalla proprietà cdTmpDir. Quando il file è stato completamente trasferito dall'agent di origine al bridge Connect:Direct , l'agent bridge Connect:Direct invia il file al nodo Connect:Direct definito dalla proprietà dell'agent cdNode . Questo nodo invia il file al nodo di destinazione Connect:Direct CD_NODE1. L'agent bridge Connect:Direct elimina il file dall'ubicazione temporanea quando il trasferimento tra due nodi Connect:Direct viene completato. Questo processo viene ripetuto per ogni file origine specificato.

Concetti correlati

"Il bridge Connect:Direct" a pagina 314
È possibile trasferire i file a e da una rete IBM Sterling Connect:Direct esistente. Utilizzare il bridge Connect:Direct , che è un componente di Managed File Transfer, per trasferire i file tra MFT e IBM Sterling Connect:Direct.

Attività correlate

"Trasferimento di un file in un nodo Connect:Direct" a pagina 316
È possibile trasferire un file da un agent Managed File Transfer a un nodo Connect:Direct utilizzando il bridge Connect:Direct . Specificare un nodo Connect:Direct come destinazione del trasferimento specificando l'agent bridge Connect:Direct come agent di destinazione e specificando il file di destinazione nel formato *connect_direct_node_name*:*file_path*.

"Trasferimento di più file in Connect:Direct utilizzando i caratteri jolly" a pagina 321
Per trasferire più file da un agent Managed File Transfer a un nodo Connect:Direct , utilizzare il bridge Connect:Direct . È possibile utilizzare i caratteri jolly nella specifica di origine fornita al comando **fteCreateTransfer** . Come per tutti i trasferimenti Managed File Transfer che implicano caratteri jolly, solo l'ultima parte del percorso file può contenere un carattere jolly. Ad esempio, `/abc/def*` è un percorso file valido e `/abc*/def` non è valido.

"Trasferimento di un file da un nodo Connect:Direct" a pagina 317
Puoi trasferire un file da un nodo Connect:Direct a un Managed File Transfer Agent utilizzando il bridge Connect:Direct . È possibile specificare un nodo di Connect:Direct come origine del trasferimento specificando l'agent bridge Connect:Direct come agent di origine e specificando la specifica di origine nel formato *connect_direct_node_name*:*file_path*.

"Transferring multiple files from a Connect:Direct node" a pagina 320
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*.

Riferimenti correlati

Il file 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

“Il bridge Connect:Direct” on page 314

È possibile trasferire i file a e da una rete IBM Sterling Connect:Direct esistente. Utilizzare il bridge Connect:Direct , che è un componente di Managed File Transfer, per trasferire i file tra MFT e IBM Sterling Connect:Direct.

Related reference

The MFT `agent.properties` file

Trasferimento di più file in Connect:Direct utilizzando i caratteri jolly

Per trasferire più file da un agent Managed File Transfer a un nodo Connect:Direct , utilizzare il bridge Connect:Direct . È possibile utilizzare i caratteri jolly nella specifica di origine fornita al comando **`fteCreateTransfer`** . Come per tutti i trasferimenti Managed File Transfer che implicano caratteri jolly, solo l'ultima parte del percorso file può contenere un carattere jolly. Ad esempio, `/abc/def*` è un percorso file valido e `/abc*/def` non è valido.

Prima di iniziare

Prima di trasferire un file, è necessario configurare il bridge Connect:Direct , che è un componente di Managed File Transfer. Per ulteriori informazioni, consultare Configurazione del bridge Connect:Direct.

Informazioni su questa attività

In questo esempio, l'agent di origine è denominato FTE_AGENT e l'agent bridge Connect:Direct è denominato CD_BRIDGE. Il nodo di destinazione Connect:Direct è denominato CD_NODE1. I file da trasferire si trovano nella directory `/reports` sul sistema in cui si trova FTE_AGENT. Vengono trasferiti solo i file con nomi che iniziano con `report`, seguiti da due caratteri e dal suffisso `.log`. Ad esempio, il file `/reports/report01.log` viene trasferito, ma il file `/reports/report1.log` non viene trasferito. I file vengono trasferiti alla directory `/home/fred` sul sistema su cui è in esecuzione CD_NODE1 .

Procedura

1. Utilizzare il comando di trasferimento fteCreatecon il valore per il parametro **-dd** (directory di destinazione) nel formato *connect_direct_node_name*:*directory_path*. Per il parametro **-da** (agent di destinazione), specificare l'agent bridge Connect:Direct .

   Nota: Il nodo Connect:Direct specificato da *connect_direct_node_name* è il nodo in cui si desidera trasferire i file, non il nodo Connect:Direct che opera come parte del bridge Connect:Direct .

   ```
   fteCreateTransfer -sa FTE_AGENT -da CD_BRIDGE
                     -dd CD_NODE1:/home/fred "/reports/report??.log"
   ```

   Per ulteriori informazioni, consultare **fteCreateTransfer**: avviare un nuovo trasferimento file.
2. L'agent di origine FTE_AGENT trasferisce il primo file che corrisponde al modello `/reports/report??.log` all'agent bridge Connect:Direct CD_BRIDGE. L'agent bridge Connect:Direct memorizza temporaneamente il file nell'ubicazione definita dalla proprietà cdTmpDir. Quando il file è stato completamente trasferito dall'agent di origine al bridge Connect:Direct , l'agent bridge Connect:Direct invia il file al nodo Connect:Direct definito dalla proprietà dell'agent cdNode . Questo nodo invia il file al nodo di destinazione Connect:Direct CD_NODE1. L'agent bridge Connect:Direct elimina il file dall'ubicazione temporanea quando il trasferimento tra due nodi Connect:Direct viene completato.Questo processo è ripetuto per ogni file di origine che corrisponde al modello carattere jolly `/reports/report??.log`.

   Nota: L'elenco di file che corrispondono al modello `/reports/report??.log` varia a seconda del sistema operativo del sistema in cui si trova l'agent di origine FTE_AGENT.

   - Se l'agent di origine si trova su un sistema con un sistema operativo Windows , la corrispondenza del modello non è sensibile al maiuscolo / minuscolo. Il modello associa tutti i file nella directory `/reports` con un nome file nel formato `report` seguito da due caratteri e un suffisso `.log`, indipendentemente dal caso in cui si trovano le lettere. Ad esempio, `Report99.Log` è una corrispondenza.
   - Se l'agent di origine si trova su un sistema con un sistema operativo Linux o UNIX , la corrispondenza del modello è sensibile al maiuscolo / minuscolo. Il modello corrisponde solo a quei file nella directory `/reports` con un nome file nel formato `report` seguito da due caratteri e un suffisso `.log`. Ad esempio, `reportAB.log` è una corrispondenza, ma `reportAB.LOG` e `Report99.Log` non sono corrispondenze.

Concetti correlati

“Il bridge Connect:Direct” a pagina 314
È possibile trasferire i file a e da una rete IBM Sterling Connect:Direct esistente. Utilizzare il bridge Connect:Direct , che è un componente di Managed File Transfer, per trasferire i file tra MFT e IBM Sterling Connect:Direct.

Attività correlate

Utilizzo di caratteri jolly con MFT

“Trasferimento di un file in un nodo Connect:Direct” a pagina 316
È possibile trasferire un file da un agent Managed File Transfer a un nodo Connect:Direct utilizzando il bridge Connect:Direct . Specificare un nodo Connect:Direct come destinazione del trasferimento specificando l'agent bridge Connect:Direct come agent di destinazione e specificando il file di destinazione nel formato *connect_direct_node_name*:*file_path*.

“Trasferimento di più file in un nodo Connect:Direct” a pagina 319

È possibile trasferire più file da un Managed File Transfer Agent ad un nodo Connect:Direct utilizzando il bridge Connect:Direct . Per utilizzare un nodo Connect:Direct come destinazione del trasferimento di più file, specificare l'agent bridge Connect:Direct come agent di destinazione e specificare la directory di destinazione nel formato *connect_direct_node_name*:*directory_path*.

“Transferring multiple files from a Connect:Direct node” a pagina 320
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*.

Riferimenti correlati

Il file MFT `agent.properties`

Ripristino e riavvio per i trasferimenti da e verso i nodi Connect:Direct

Managed File Transfer potrebbe non essere in grado di connettersi al tuo nodo IBM Sterling Connect:Direct durante un trasferimento; ad esempio, se il nodo diventa non disponibile. Managed File Transfer tenta di ripristinare il trasferimento oppure il trasferimento ha esito negativo e viene prodotto un messaggio di errore.

Se il nodo Connect:Direct diventa non disponibile

Se il nodo Connect:Direct diventa non disponibile; ad esempio, a causa di un'interruzione di rete o di alimentazione, Managed File Transfer recupera un trasferimento file nei seguenti modi:

- Se Managed File Transfer non è stato precedentemente connesso correttamente al nodo Connect:Direct come parte di questa richiesta di trasferimento, il trasferimento viene ritentato per un periodo di tempo determinato dai valori di **cdMaxConnectionRetries** e **recoverableTransferRetryInterval properties**. Queste proprietà sono specificate nel file `agent.properties` per l'agent bridge Connect:Direct . Il trasferimento non riesce e viene prodotto un messaggio di errore, dopo che il numero di tentativi non riusciti raggiunge il valore di **cdMaxConnectionRetries property**. Per impostazione predefinita, il trasferimento viene tentato indefinitamente, con 60 secondi tra i tentativi.
- Se Managed File Transfer si è precedentemente collegato correttamente al nodo Connect:Direct come parte di questa richiesta di trasferimento, il trasferimento viene ritentato per un periodo di tempo determinato dai valori delle proprietà **cdMaxPartialWorkConnectionRetries** e **recoverableTransferRetryInterval** . Il trasferimento non riesce e viene prodotto un messaggio di errore, dopo che il numero di tentativi non riusciti raggiunge il valore della proprietà **cdMaxPartialWorkConnectionRetries** . Per impostazione predefinita, il trasferimento viene tentato indefinitamente, con 60 secondi tra i tentativi.
- Per alcuni tipi di errore del nodo Connect:Direct , ad esempio il nodo che viene arrestato in modo forzato, i processi Connect:Direct passano allo stato `Held Due to Error (HE)` quando il nodo viene ripristinato. Una volta ripristinato il nodo, Managed File Transfer riprende automaticamente tutti i processi Connect:Direct correlati al trasferimento file e con lo stato HE.
- Se il trasferimento ha esito negativo, tutti i file temporanei relativi al trasferimento vengono eliminati dal sistema che ospita il bridge Connect:Direct . La posizione di questi file temporanei è definita dalla proprietà **cdTmpDir** .
- Se il trasferimento è da Managed File Transfer a Connect:Directe viene specificata una disposizione di origine di eliminazione, i file di origine non vengono eliminati se il trasferimento non riesce.

Se le credenziali utente del nodo Connect:Direct non sono valide

Se Managed File Transfer non riesce a connettersi al nodo Connect:Direct perché le credenziali dell'utente vengono rifiutate dal nodo, il trasferimento non riesce e viene prodotto un messaggio di errore. In questa situazione, verificare di aver fornito le credenziali utente corrette per il nodo Connect:Direct . Per ulteriori informazioni, consultare Associazione delle credenziali per Connect:Direct.

Se l'agent bridge Connect:Direct diventa non disponibile

Se l'agent bridge Connect:Direct diventa non disponibile, i trasferimenti file in corso vengono ripristinati allo stesso modo dei trasferimenti Managed File Transfer standard. Per ulteriori informazioni, consultare “Ripristino e riavvio di MFT” a pagina 330.

Concetti correlati

“Il bridge Connect:Direct” a pagina 314
È possibile trasferire i file a e da una rete IBM Sterling Connect:Direct esistente. Utilizzare il bridge Connect:Direct , che è un componente di Managed File Transfer, per trasferire i file tra MFT e IBM Sterling Connect:Direct.

“Ripristino e riavvio di MFT” a pagina 330
Se l'agent o il gestore code non sono disponibili per qualsiasi motivo, ad esempio a causa di un errore di alimentazione o di rete, Managed File Transfer esegue il ripristino come riportato di seguito in questi scenari:

Attività correlate

Configurazione del bridge Connect:Direct

Riferimenti correlati

Il file MFT `agent.properties`

Inoltro di un processo Connect:Direct definito dall'utente da una richiesta di trasferimento file

È possibile inviare una richiesta di trasferimento per un trasferimento che passa attraverso l'agent bridge Connect:Direct che richiama un processo Connect:Direct definito dall'utente come parte del trasferimento file.

Per impostazione predefinita, quando si inoltra una richiesta di trasferimento file per un trasferimento che passa attraverso il bridge Connect:Direct , l'agent bridge Connect:Direct genera il processo Connect:Direct utilizzato per trasferire il file al o dal nodo Connect:Direct remoto.

Tuttavia, è possibile configurare l'agent bridge Connect:Direct per richiamare un processo Connect:Direct definito dall'utente utilizzando il file `ConnectDirectProcessDefinition.xml` .

Il file `ConnectDirectProcessDefinition.xml`

Il comando **`fteCreateCDAgent`** crea il file `ConnectDirectProcessDefinitions.xml` nella directory di configurazione dell'agente *`MQ_DATA_PATH`*`/mqft/config/`*`coordination_queue_manager`*`/agents/`*`cd_bridge_agent_name`*. Prima di poter richiamare i processi Connect:Direct definiti dall'utente dall'agent bridge Connect:Direct , è necessario impostare le definizioni dei processi modificando questo file.

Il file definisce una o più serie di processi che includono l'ubicazione di uno o più processi Connect:Direct richiamati come parte di un trasferimento. Ogni serie di processi include un numero di condizioni. Se il trasferimento soddisfa tutte le condizioni della serie di processi, la serie di processi viene utilizzata per specificare quali processi Connect:Direct vengono richiamati dal trasferimento. Per ulteriori informazioni, consultare “Specifica del processo Connect:Direct da iniziare utilizzando il file ConnectDirectProcessDefinition.xml” a pagina 325.

Variabili simboliche intrinseche

È possibile utilizzare le variabili simboliche intrinseche definite da Managed File Transfer per sostituire i valori nei processi Connect:Direct definiti dall'utente. Per seguire la convenzione di denominazione Connect:Direct , tutte le variabili simboliche intrinseche utilizzate da Managed File Transfer hanno il formato %FTE seguito da cinque caratteri alfanumerici maiuscoli.

Quando si crea un processo per trasferire i file da un nodo Connect:Direct al sistema di bridge Connect:Direct, è necessario utilizzare la variabile intrinseca %FTETFILE come valore di TO FILE nel processo Connect:Direct. Quando si crea un processo per trasferire i file a un nodo Connect:Direct dal

sistema di bridge Connect:Direct, è necessario utilizzare la variabile intrinseca %FTEFFILE come valore di FROM FILE nel processo Connect:Direct. Queste variabili contengono i percorsi di file temporanei che l'agent bridge Connect:Direct utilizza per i trasferimenti all'interno e all'esterno della rete Managed File Transfer.

Per ulteriori informazioni sulle variabili simboliche intrinseche, consultare la documentazione del prodotto Connect:Direct .

Processi Connect:Direct di esempio

Managed File Transfer fornisce processi Connect:Direct di esempio. Questi esempi si trovano nella seguente directory: *MQ_INSTALLATION_PATH*/mqft/samples/ConnectDirectProcessTemplates.

Attività correlate

"Specifica del processo Connect:Direct da iniziare utilizzando il file ConnectDirectProcessDefinition.xml" a pagina 325
Specificare quale processo Connect:Direct avviare come parte di un trasferimento Managed File Transfer . Managed File Transfer fornisce un file XML che è possibile modificare per specificare definizioni di processo.

"Utilizzo di variabili simboliche intrinseche nei processi Connect:Direct richiamati da Managed File Transfer" a pagina 326
È possibile richiamare un processo Connect:Direct definito dall'utente da un trasferimento Managed File Transfer e trasmettere le informazioni dal trasferimento al processo Connect:Direct utilizzando variabili simboliche intrinseche nella definizione del processo.

Riferimenti correlati

Formato file di definizione del processo Connect:Direct

Variabili di sostituzione da utilizzare con processi Connect:Direct definiti dall'utente

Specifica del processo Connect:Direct da iniziare utilizzando il file ConnectDirectProcessDefinition.xml

Specificare quale processo Connect:Direct avviare come parte di un trasferimento Managed File Transfer . Managed File Transfer fornisce un file XML che è possibile modificare per specificare definizioni di processo.

Informazioni su questa attività

Il comando **fteCreateCDAgent** crea il file ConnectDirectProcessDefinitions.xml nella directory di configurazione dell'agente *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*cd_bridge_agent_name*. Prima di poter richiamare i processi Connect:Direct definiti dall'utente dall'agent bridge Connect:Direct , è necessario impostare le definizioni dei processi modificando questo file.

Per ogni processo che si desidera specificare di richiamare come parte di un trasferimento tramite il bridge Connect:Direct , effettuare quanto segue:

Procedura

1. Definire il processo Connect:Direct che si desidera venga richiamato dall'agent bridge Connect:Direct come parte del trasferimento e salvare il template del processo nel file.
2. Aprire il file *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*cd_bridge_agent_name*/ConnectDirectProcessDefinitions.xml in un editor di testo.
3. Creare un elemento <processSet> .
4. All'interno dell'elemento <processSet> , creare un elemento <condition> .
5. All'interno dell'elemento <condition> , creare uno o più elementi che definiscono una condizione a cui la richiesta di trasferimento deve corrispondere per richiamare il processo Connect:Direct definito nel passo 1. Questi elementi possono essere elementi <match> o <defined> .

- Utilizzare un elemento <match> per indicare che il valore di una variabile deve corrispondere a un modello. Creare l'elemento <match> con i seguenti attributi:
  - variable - il nome della variabile di cui viene confrontato il valore. La variabile è un simbolo intrinseco. Per ulteriori informazioni, consultare Variabili di sostituzione da utilizzare con processi Connect:Direct definiti dall'utente.
  - value - il modello da confrontare con il valore della variabile specificata.
  - Facoltativo: pattern - il tipo di pattern utilizzato dal valore dell'attributo value . Questo tipo di pattern può essere wildcard o regex. Questo attributo è facoltativo e il valore predefinito è wildcard.
- Utilizzare un elemento <defined> per specificare che una variabile deve avere un valore definito. Creare l'elemento <defined> con il seguente attributo:
  - variable - il nome della variabile che deve avere un valore definito. La variabile è un simbolo intrinseco. Per ulteriori informazioni, consultare Variabili di sostituzione da utilizzare con processi Connect:Direct definiti dall'utente.

Le condizioni specificate nell'elemento <condition> sono combinate con un AND logico. Tutte le condizioni devono essere soddisfatte affinché l'agent bridge Connect:Direct richiami il processo specificato da questo elemento <processSet> . Se non si specifica un elemento <condition> , la serie di processi corrisponde a tutti i trasferimenti.

6. All'interno dell'elemento <processSet> , creare un elemento <process> .
7. All'interno dell'elemento <process> , creare un elemento <transfer> .

   L'elemento transfer specifica il processo Connect:Direct che l'agent bridge Connect:Direct richiama come parte del trasferimento. Creare l'elemento <transfer> con il seguente attributo:

   - process- - l'ubicazione del processo Connect:Direct definito nel passo 1. L'ubicazione di questo file è specificata con un percorso assoluto o relativo alla directory *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*cd_bridge_agent_name* .

Risultati

Quando si cerca una corrispondenza di condizioni, l'agent bridge Connect:Direct esegue la ricerca dall'inizio del file alla fine del file. La prima corrispondenza trovata è quella utilizzata.

Attività correlate

Configurazione del bridge Connect:Direct

Riferimenti correlati

Formato file di definizione del processo Connect:Direct

fteCreateCDAgent: crea un agent bridge Connect:Direct

Utilizzo di variabili simboliche intrinseche nei processi Connect:Direct richiamati da Managed File Transfer

È possibile richiamare un processo Connect:Direct definito dall'utente da un trasferimento Managed File Transfer e trasmettere le informazioni dal trasferimento al processo Connect:Direct utilizzando variabili simboliche intrinseche nella definizione del processo.

Informazioni su questa attività

Questo esempio utilizza variabili simboliche intrinseche per passare le informazioni da un trasferimento Managed File Transfer a un processo definito dall'utente Connect:Direct . Per ulteriori informazioni relative alle variabili simboliche intrinseche utilizzate da Managed File Transfer, consultare Variabili di sostituzione da utilizzare con i processi Connect:Direct definiti dall'utente.

In questo esempio, il file viene trasferito da Managed File Transfer Agent a un nodo bridge Connect:Direct . La prima parte del trasferimento viene eseguita da Managed File Transfer. La seconda parte del trasferimento viene eseguita da un processo Connect:Direct definito dall'utente.

Procedura

1. Creare un processo Connect:Direct che utilizza variabili simboliche intrinseche.

```
%FTEPNAME PROCESS
    SNODE=%FTESNODE
    PNODEID=(%FTEPUSER,%FTEPPASS)
    SNODEID=(%FTESUSER,%FTESPASS)

COPY001 COPY
    FROM (
        FILE=%FTEFFILE
        DISP=%FTEFDISP
    )
    TO (
        FILE=%FTETFILE
        DISP=%FTETDISP
    )
PEND
```

2. Salvare questo processo in un file di testo nella seguente ubicazione: *MQ_DATA_PATH*/mqft/config/*coordination_queue_manager*/agents/*cd_bridge_agent*/Example.cdp
3. Modificare il file ConnectDirectProcessDefinition.xml per includere una regola che richiama il processo Connect:Direct creato nel Passo 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>
```

   In questo esempio, se una richiesta di trasferimento viene inoltrata all'agent bridge Connect:Direct che ha TOBERMORY come nodo Connect:Direct di origine o di destinazione, viene richiamato il processo Example.cdp Connect:Direct .
4. Inoltrare una richiesta di trasferimento file che soddisfi le condizioni definite nel file ConnectDirectProcessDefinition.xml nel Passo 3.

   Ad esempio:

   ```
   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
   ```

   In questo esempio, il nodo Connect:Direct di destinazione è TOBERMORY. Questo nodo è il nodo secondario nel trasferimento e il valore di %FTESNODE è impostato su TOBERMORY. Questo comando corrisponde alla condizione impostata nel file ConnectDirectProcessDefinition.xml .
5. Managed File Transfer trasferisce il file di origine in un'ubicazione temporanea sullo stesso sistema dell'agente bridge Connect:Direct .
6. L'agent bridge Connect:Direct imposta i valori delle variabili simboliche intrinseche dalle informazioni nella richiesta di trasferimento e nelle informazioni di configurazione.

   Le variabili simboliche intrinseche sono impostate sui seguenti valori:

   - %FTEPNAME=*nome_processo* - Questo valore è un nome processo di 8 caratteri generato dall'agente bridge Connect:Direct .

- %FTESNODE=TOBERMORY - Questo valore è impostato dal parametro **-df** del comando **fteCreateTransfer** .
- %FTEPUSER, =*primary_node_user* - Queste informazioni vengono prese dal file `ConnectDirectCredentials.xml` .
- %FTEPPASS=*primary_node_user_password* - Queste informazioni vengono prese dal file `ConnectDirectCredentials.xml` .
- %FTESUSER, =*utente_nodo_secondario* - Queste informazioni vengono prese dal file `ConnectDirectCredentials.xml` .
- %FTESPASS=*secondary_node_user_password* - Queste informazioni vengono prese dal file `ConnectDirectCredentials.xml` .
- %FTEFFILE =*ubicazione_temporanea* - Questo valore è l'ubicazione temporanea del file sullo stesso sistema dell'agente bridge Connect:Direct .
- %FTEFDISP=leave - Questo valore è impostato dal parametro **-sd** del comando **fteCreateTransfer** .
- %FTETFILE=`/home/bulgaria/destination.txt` - Questo valore è impostato dal parametro **-df** del comando **fteCreateTransfer** .
- %FTETDISP=overwrite - questo valore è impostato dal parametro **-de** del comando **fteCreateTransfer** .

7. Il processo Connect:Direct viene avviato sul nodo bridge Connect:Direct . Connect:Direct trasferisce il file dall'ubicazione temporanea sul sistema bridge Connect:Direct alla destinazione `/home/bulgaria/destination.txt` sul sistema su cui è in esecuzione il nodo Connect:Direct TOBERMORY.

Concetti correlati

“Inoltro di un processo Connect:Direct definito dall'utente da una richiesta di trasferimento file” a pagina 324

È possibile inviare una richiesta di trasferimento per un trasferimento che passa attraverso l'agent bridge Connect:Direct che richiama un processo Connect:Direct definito dall'utente come parte del trasferimento file.

Riferimenti correlati

Variabili di sostituzione da utilizzare con processi Connect:Direct definiti dall'utente

Utilizzo dei processi Connect:Direct per inoltrare richieste di trasferimento Managed File Transfer

È possibile inviare una richiesta di trasferimento all'agent bridge Connect:Direct da un processo Connect:Direct . Managed File Transfer fornisce comandi che possono essere richiamati da un'istruzione **RUN TASK** in un processo Connect:Direct .

Managed File Transfer fornisce i seguenti comandi da utilizzare con i processi Connect:Direct :

ftetag
Specificare questo comando in un passo che precede il comando **ftebxfer** o **ftecxfer** per creare le informazioni di verifica richieste per il trasferimento. Questo comando prende la specifica di origine del trasferimento come parametro. Per informazioni sul formato della specifica di origine, consultare **fteCreateTransfer**: avvio di un nuovo trasferimento file.

ftebxfer
Specificare questo comando per creare una richiesta di trasferimento file quando il gestore code a cui viene inviata la richiesta di trasferimento si trova sullo stesso sistema del nodo Connect:Direct che inoltra il comando. Questo comando utilizza gli stessi parametri del comando **fteCreateTransfer** . Per informazioni su questi parametri, vedere **fteCreateTransfer**: avvio di un nuovo trasferimento file. Questo comando ha anche un parametro aggiuntivo:

-qmgrname
Obbligatorio. Il nome del gestore code a cui inoltrare il comando.

ftecxfer

Specificare questo comando per creare una richiesta di trasferimento file quando il gestore code a cui è inoltrata la richiesta di trasferimento si trova su un sistema differente sul nodo Connect:Direct che inoltra il comando. Questo comando utilizza gli stessi parametri del comando **fteCreateTransfer** . Per informazioni sui parametri, consultare **fteCreateTransfer**: avviare un nuovo trasferimento file. Questo comando ha anche tre parametri aggiuntivi:

-qmgrname

Obbligatorio. Il nome del gestore code a cui inoltrare il comando.

-nomeconnessione

Obbligatorio. L'host e la porta del gestore code a cui inoltrare il comando, specificati in formato CONNAME IBM MQ . Ad esempio, `host.example.com(1337)`.

-nomecanale

Facoltativo. Il nome del canale da utilizzare per connettersi al gestore code a cui inoltrare il comando. Se non viene specificato, viene utilizzato il valore predefinito SYSTEM.DEF.SVRCONN .

Attività correlate

“Creazione e inoltro di un processo Connect:Direct che richiama Managed File Transfer utilizzando il richiedente Connect:Direct .” a pagina 329

Il richiedente Connect:Direct è una GUI (graphical user interface) che è possibile utilizzare per creare e inoltrare un processo Connect:Direct che richiama Managed File Transfer.

Riferimenti correlati

Esempio: un file di elaborazione Connect:Direct che richiama i comandi MFT

Creazione e inoltro di un processo Connect:Direct che richiama Managed File Transfer utilizzando il richiedente Connect:Direct .

Il richiedente Connect:Direct è una GUI (graphical user interface) che è possibile utilizzare per creare e inoltrare un processo Connect:Direct che richiama Managed File Transfer.

Informazioni su questa attività

Questa attività descrive come creare un processo Connect:Direct che richiama il comando Managed File Transfer **ftecxfer** o il comando **ftebxfer** . Utilizzare il comando **ftecxfer** quando il gestore code a cui viene inoltrata la richiesta di trasferimento si trova su un sistema differente rispetto al nodo Connect:Direct che inoltra il comando. Utilizzare il comando **ftebxfer** quando il gestore code a cui viene inoltrata la richiesta di trasferimento si trova sullo stesso sistema del nodo Connect:Direct che inoltra il comando. Il comando **ftecxfer** effettua una connessione client al gestore code agent dell'agent di origine del trasferimento. Prima di richiamare un comando **ftecxfer** , è necessario richiamare il comando **ftetag** e trasmettergli le informazioni sulla specifica di origine. Ciò consente al processo di essere registrato e controllato allo stesso modo dei trasferimenti avviati da Managed File Transfer.

Procedura

1. Avviare il richiedente Connect:Direct .
2. Nella scheda **Nodi** del pannello, selezionare il nodo Connect:Direct utilizzato come nodo primario del processo.
3. Selezionare **File** > **Nuovo** > **Processo**. Viene visualizzata la finestra **Proprietà del processo** .
4. Nel campo **Nome:** , immettere il nome del processo.
5. Selezionare il nodo secondario dall'elenco **Snode** > **Nome** .
6. Scegliere il sistema operativo del nodo secondario dall'elenco **Snode** > **Sistema operativo** .
7. Opzionale: Completare tutte le ulteriori informazioni richieste in questa finestra.
8. Fare clic su **OK**. La finestra **Proprietà processo** viene chiusa.
9. Creare un'istruzione che esegue il comando Managed File Transfer **ftetag** .
   a) Fare clic con il tasto destro del mouse nella finestra **Processo** sull'istruzione **End** .

b) Selezionare **Inserisci** > **Esegui attività**. Viene visualizzata la finestra **Esegui istruzione attività** .

c) Nel campo **Etichetta:** , immettere `Tag`.

d) Nel campo **Parametri o comandi facoltativi** , immettere `pgm(`*MQ_INSTALLATION_PATH*`/bin/ftetag) args(`*source_specification*`)`. Per ulteriori informazioni sul formato di *source_specification*, consultare **fteCreateTransfer**: start a new file transfer.

e) Fare clic su **OK**. La finestra **Esegui istruzione attività** viene chiusa.

10. Creare un'istruzione che esegue il comando Managed File Transfer **ftecxfer** o **ftebxfer** .

a) Fare clic con il tasto destro del mouse nella finestra **Processo** sull'istruzione **End** .

b) Selezionare **Inserisci** > **Esegui attività**. Viene visualizzata la finestra **Esegui istruzione attività** .

c) Nel campo **Etichetta:** , immettere `Transfer`.

d) Nel campo **Parametri o comandi facoltativi** , immettere `pgm(`*MQ_INSTALLATION_PATH*`/bin/ftecxfer) args(`*parameters*`)` o `pgm(`*MQ_INSTALLATION_PATH*`/bin/ftebxfer) args(`*parameters*`)` in base al comando scelto. I parametri utilizzati dai comandi **ftecxfer** e **ftebxfer** sono gli stessi utilizzati dal comando **fteCreateTransfer** , oltre ad alcuni parametri aggiuntivi specifici per **ftecxfer** e **ftebxfer**. Per ulteriori informazioni, consultare **fteCreateTransfer**: avviare un nuovo trasferimento file e “Utilizzo dei processi Connect:Direct per inoltrare richieste di trasferimento Managed File Transfer” a pagina 328.

e) Fare clic su **OK**. La finestra **Esegui istruzione attività** viene chiusa.

11. Opzionale: Creare eventuali istruzioni aggiuntive richieste.
12. Inoltrare il processo.

a) Fare clic con il tasto destro del mouse nella finestra **Processo** .

b) Selezionare **Inoltra**. Si apre la finestra **Connect:Direct Allega** .

c) Immettere il nome utente e la password da utilizzare per eseguire il processo.

d) Fare clic su **OK**.

Concetti correlati

“Utilizzo dei processi Connect:Direct per inoltrare richieste di trasferimento Managed File Transfer” a pagina 328
È possibile inviare una richiesta di trasferimento all'agent bridge Connect:Direct da un processo Connect:Direct . Managed File Transfer fornisce comandi che possono essere richiamati da un'istruzione **RUN TASK** in un processo Connect:Direct .

Utilizzo di MFT da IBM Integration Bus

È possibile utilizzare Managed File Transfer da IBM Integration Bus utilizzando i nodi FTEOutput e FTEInput.

- Utilizzare il nodo FTEInput per trasferire un file sulla rete utilizzando Managed File Transfer ed elaborare tale file come parte di un flusso Integration Bus .
- Utilizzare il nodo FTEOutput per trasferire un file che è stato emesso da un flusso Integration Bus in un'altra ubicazione nella rete.

Gli agent che trasferiscono i file da o verso l'agent del broker possono essere a qualsiasi livello di Managed File Transfer.

Per ulteriori informazioni, fare riferimento alla documentazione del prodottoIBM Integration Bus.

Ripristino e riavvio di MFT

Se l'agent o il gestore code non sono disponibili per qualsiasi motivo, ad esempio a causa di un errore di alimentazione o di rete, Managed File Transfer esegue il ripristino come riportato di seguito in questi scenari:

- In genere, se si verifica un problema durante il trasferimento di un file, Managed File Transfer recupera e riavvia il trasferimento file dopo che il problema è stato risolto.

- Se un file che era in fase di trasferimento viene eliminato o modificato mentre l'agent o il gestore code non sono disponibili, il trasferimento ha esito negativo e si riceve un messaggio nel log di trasferimento che fornisce i dettagli sull'errore.
- Se un processo agent non riesce durante un trasferimento file, il trasferimento continua quando si riavvia l'agent.
- Se un agent perde la connessione al gestore code dell'agent, l'agent attende durante il tentativo di riconnettersi al gestore code. Quando l'agent si riconnette correttamente al gestore code, il trasferimento corrente continua.
- Se l'agent viene arrestato per qualsiasi motivo, tutti i monitoraggi delle risorse associati ad un agent arrestano il polling. Quando l'agent viene ripristinato, vengono riavviati anche i monitoraggi e riprende il polling delle risorse.
- Per un trasferimento file con una disposizione di origine `delete`, se si verifica un ripristino dopo che tutti i dati vengono inviati da un agent di origine a un agent di destinazione, il file di origine viene sbloccato prima dell'eliminazione. Questo sblocco indica che il file di origine potrebbe essere modificato prima che il file venga eliminato. Pertanto, è considerato non sicuro eliminare il file di origine e viene visualizzata la seguente avvertenza:

  ```
  BFGTR0075W: The source file has not been deleted because it is possible that the source file
  was modified after the source file was transferred.
  ```

  In questo caso, verificare che il contenuto del file di origine non sia stato modificato e quindi eliminare manualmente il file di origine.

Puoi verificare lo stato dei tuoi trasferimenti in IBM MQ Explorer. Se i trasferimenti vengono visualizzati come `Stalled`, potrebbe essere necessario intraprendere un'azione correttiva perché lo stato di stallo denota un problema con l'agent o tra i due agent coinvolti nel trasferimento.

Attività correlate

“Impostazione di una scadenza per il recupero di trasferimenti in stallo” a pagina 331
È possibile impostare un timeout di ripristino del trasferimento per i trasferimenti di file in stallo che si applica a tutti i trasferimenti per un agent di origine. È anche possibile impostare un timeout di ripristino trasferimento per un singolo trasferimento. Se si imposta un periodo di tempo specifico, in secondi, durante il quale un agent di origine tenta di ripristinare un trasferimento file in stallo e il trasferimento non riesce quando l'agent raggiunge il timeout, il trasferimento non riesce.

Impostazione di una scadenza per il recupero di trasferimenti in stallo

È possibile impostare un timeout di ripristino del trasferimento per i trasferimenti di file in stallo che si applica a tutti i trasferimenti per un agent di origine. È anche possibile impostare un timeout di ripristino trasferimento per un singolo trasferimento. Se si imposta un periodo di tempo specifico, in secondi, durante il quale un agent di origine tenta di ripristinare un trasferimento file in stallo e il trasferimento non riesce quando l'agent raggiunge il timeout, il trasferimento non riesce.

Informazioni su questa attività

È possibile impostare un timeout di ripristino trasferimento che si applica a tutti i trasferimenti per un agent di origine aggiungendo un parametro di timeout di ripristino trasferimento al file `agent.properties` dell'agent. È anche possibile impostare un timeout di ripristino del trasferimento per un singolo trasferimento dalla riga comandi o con IBM MQ Exploreroppure utilizzando le attività Apache Ant . Se è presente un valore di timeout di ripristino del trasferimento impostato nel file `agent.properties` , l'impostazione del timeout di ripristino del trasferimento per un singolo trasferimento sovrascrive il valore nel file `agent.properties` .

Esistono tre opzioni per il timeout del ripristino del trasferimento:

- L'agent continua a tentare di recuperare il trasferimento bloccato fino a quando non viene completato correttamente. È uguale al comportamento predefinito dell'agent se il timeout di ripristino del trasferimento non è impostato.

- L'agent contrassegna il trasferimento come non riuscito immediatamente dopo l'avvio del ripristino.
- L'agent continua a ripetere il trasferimento in stallo per un periodo di tempo specificato prima che il trasferimento venga contrassegnato come non riuscito.

L'impostazione del timeout di ripristino del trasferimento file è facoltativa. Se non lo si imposta, i trasferimenti seguono il comportamento predefinito, in cui l'agent continua a tentare di ripristinare un trasferimento bloccato fino a quando non viene eseguito correttamente.

Concetti correlati

"Ripristino e riavvio di MFT" a pagina 330
Se l'agent o il gestore code non sono disponibili per qualsiasi motivo, ad esempio a causa di un errore di alimentazione o di rete, Managed File Transfer esegue il ripristino come riportato di seguito in questi scenari:

Concetti di timeout di ripristino del trasferimento

È possibile impostare la quantità di tempo, in secondi, durante la quale un agent di origine continua a tentare di recuperare un trasferimento file bloccato. Se il trasferimento non ha esito positivo quando l'agent raggiunge il timeout per l'intervallo di tentativi, il trasferimento ha esito negativo.

Precedenza timeout di recupero

Un valore di superotempo di ripristino del trasferimento per un singolo trasferimento specificato tramite i comandi **fteCreateTransfer**, **fteCreateTemplate**o **fteCreateMonitor** , oppure utilizzando IBM MQ Explorer, o specificato nell'elemento nidificato **fte:filespec** , ha la precedenza sul valore specificato per il parametro **transferRecoveryTimeout** nel file `agent.properties` per l'agent di origine.

Ad esempio, se il comando **fteCreateTransfer** viene avviato senza la coppia parametro - valore **-rt** , l'agent di origine AGENT1 controlla il file `agent.properties` per un valore **transferRecoveryTimeout** per determinare il funzionamento del timeout di recupero:

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -df C:\import\transferredfile.txt
C:\export\originalfile.txt
```

Se il parametro **transferRecoveryTimeout** nel file `agent.properties` non è impostato o è impostato su `-1`, l'agent segue il comportamento predefinito e tenta di ripristinare il trasferimento fino a quando non viene eseguito correttamente.

Tuttavia, se il comando **fteCreateTransfer** include il parametro **-rt** , il valore di questo parametro ha la precedenza sul valore nel file `agent.properties` e viene utilizzato come impostazione di timeout di ripristino per il trasferimento:

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -rt 21600 -df C:\import\transferredfile.txt
C:\export\originalfile.txt
```

Contatore timeout di ripristino

Il contatore del timeout di ripristino viene avviato quando il trasferimento entra nello stato di recupero. Viene pubblicato un messaggio di log di trasferimento nel SISTEMA SYSTEM.FTE con la stringa di argomenti `Log/agent_name/transfer_ID` per indicare che lo stato del trasferimento è stato modificato in ripristino e l'ora dell'orologio dell'agent di origine in cui è stato modificato lo stato. Se il trasferimento viene ripreso entro l'intervallo di tentativi impostato e non raggiunge il timeout di recupero (contatore < =timeout di ripristino), il contatore viene reimpostato su 0, pronto per iniziare di nuovo se il trasferimento entra nel recupero.

Se il contatore raggiunge il valore massimo impostato per il timeout di ripristino (contatore == timeout di ripristino), il ripristino del trasferimento si arresta e l'agent di origine riporta il trasferimento come non riuscito. Questo tipo di errore di trasferimento, causato dal fatto che il trasferimento ha raggiunto il timeout di ripristino, è indicato dal codice messaggio, `RECOVERY TIMEOUT (69)`. Un altro messaggio del log di trasferimento viene pubblicato nel SISTEMA SYSTEM.FTE FTE, con una stringa di argomento `Log/`

agent_name/transfer_ID, per indicare che il trasferimento non è riuscito e include un messaggio, il codice di ritorno e il log eventi dell'agent di origine. Il log degli eventi dell'agente di origine viene aggiornato con un messaggio quando si verifica uno dei seguenti eventi durante il recupero:

- Quando il parametro di timeout del ripristino è impostato su un valore maggiore di -1, il trasferimento avvia il ripristino. Il log eventi dell'agente viene aggiornato per indicare l'inizio del timer di recupero per il **TransferId** e la quantità di tempo che l'agente di origine attende prima di avviare l'elaborazione del timeout di recupero.
- Quando il trasferimento di recupero viene ripristinato, il log eventi dell'agent di origine viene aggiornato con un nuovo messaggio per indicare che il **TransferId** che era in ripristino viene ripristinato.
- Quando un trasferimento di ripristino è scaduto, il log eventi dell'agent di origine viene aggiornato per indicare il **TransferId** che non è riuscito durante il recupero a causa del timeout di ripristino.

Questi messaggi di log consentono agli utenti (sottoscrittori e logger) di identificare i trasferimenti non riusciti a causa del supero tempo di recupero del trasferimento.

Il contatore per il timeout di ripristino si trova sempre sull'agent di origine. Tuttavia, se l'agent di destinazione non riesce a ricevere informazioni dall'agent di origine in modo tempestivo, può inviare una richiesta all'agent di origine per inserire il trasferimento nel ripristino. Per un trasferimento in cui è impostata l'opzione di timeout di ripristino, l'agent di origine avvia il contatore di timeout di recupero quando riceve la richiesta dall'agent di destinazione.

La gestione manuale è ancora richiesta per i trasferimenti che non utilizzano l'opzione di timeout di ripristino, i trasferimenti non riusciti e parzialmente completi.

Per le serie di trasferimento, dove una singola richiesta di trasferimento viene emessa per più file e alcuni dei file sono stati completati correttamente ma solo parzialmente, il trasferimento è ancora contrassegnato come non riuscito in quanto non è stato completato come previsto. L'agent di origine potrebbe essere scaduto durante il trasferimento del file parzialmente completato.

Verificare che l'agent di destinazione e il server file siano pronti e in uno stato per accettare i trasferimenti file.

È necessario emettere nuovamente la richiesta di trasferimento per l'intera serie, ma per evitare problemi poiché alcuni file rimangono sulla destinazione dal tentativo di trasferimento iniziale, è possibile emettere la nuova richiesta con l'opzione `sovrascrivi se è specificata l'opzione` esistente. Ciò garantisce che la serie incompleta di file dal tentativo di trasferimento precedente venga ripulita come parte del trasferimento nuovo, prima che i file vengano scritti nuovamente nella destinazione.

Da IBM MQ 9.1.5, non è più necessario rimuovere manualmente i file parte lasciati su una destinazione dopo un tentativo di trasferimento iniziale non riuscito. Se un timeout di ripristino del trasferimento è impostato per un trasferimento, l'agent di origine sposta il trasferimento nello stato RecoveryTimedOut se il ripristino del trasferimento va in timeout. Dopo che il trasferimento è stato risincronizzato, l'agent di destinazione rimuove tutti i file parte creati durante il trasferimento e invia un messaggio di completamento all'agent di origine.

Tracce e messaggi

I punti di traccia sono inclusi a scopo diagnostico. Vengono registrati il valore di timeout di ripristino, l'inizio dell'intervallo di nuovi tentativi, l'avvio del periodo di ripresa e la reimpostazione del contatore e se il trasferimento è andato in timeout e non è riuscito. In caso di un problema o di un comportamento imprevisto, è possibile raccogliere i file di traccia e di log di output dell'agent di origine e fornirli quando richiesto dal supporto IBM , per facilitare la risoluzione dei problemi.

I messaggi notificano quando:

- Un trasferimento avvia il ripristino (BFGTR0081I)
- Un trasferimento è terminato perché si è verificato un timeout dal ripristino (BFGSS0081E)
- L'operazione di trasferimento riprende dopo il ripristino (BFGTR0082I)

Concetti correlati

“Ripristino e riavvio di MFT” a pagina 330

Se l'agent o il gestore code non sono disponibili per qualsiasi motivo, ad esempio a causa di un errore di alimentazione o di rete, Managed File Transfer esegue il ripristino come riportato di seguito in questi scenari:

Impostazione del timeout di ripristino del trasferimento per tutti i trasferimenti per un agent di origine

È possibile impostare un timeout di ripristino del trasferimento che si applica a tutti i trasferimenti per un agent di origine aggiungendo il parametro **transferRecoveryTimeout** al file `agent.properties` .

Informazioni su questa attività

Per impostare un timeout di ripristino treansfer che si applica a tutti i trasferimenti per un agent di origine, aggiungere la coppia parametro e valore per **transferRecoveryTimeout** al file agent.properties .

Esistono tre opzioni per il parametro **transferRecoveryTimeout** :

-1
L'agent continua a tentare di recuperare il trasferimento bloccato fino a quando il trasferimento non viene eseguito correttamente. L'uso di questa opzione equivale al comportamento predefinito dell'agente quando la proprietà non è impostata.

0
L'agent arresta il trasferimento file non appena avvia il ripristino.

>0
L'agent continua a tentare di recuperare il trasferimento in stallo per il periodo di tempo in secondi come impostato dal valore intero positivo specificato.

Tutte le modifiche apportate al file `agent.properties` diventano effettive solo dopo il riavvio dell'agent.

Se necessario, è possibile sovrascrivere il valore di supero tempo di ripristino del trasferimento nel file `agent.properties` per un singolo trasferimento. Per ulteriori informazioni, consultare “Impostazione del timeout di ripristino del trasferimento per singoli trasferimenti” a pagina 335.

Procedura

- Per specificare che l'agent continua a tentare di ripristinare il trasferimento in stallo fino a quando non viene completato correttamente, impostare un valore di timeout di ripristino del trasferimento `-1` come mostrato nel seguente esempio:

  ```
  transferRecoveryTimeout=-1
  ```

- Per specificare che l'agent contrassegna il trasferimento come non riuscito immediatamente dopo l'immissione del ripristino, impostare il valore di timeout del ripristino del trasferimento `0` come mostrato nel seguente esempio:

  ```
  transferRecoveryTimeout=0
  ```

- Per specificare che l'agent continua a ritentare un trasferimento in stallo per un determinato periodo di tempo prima che il trasferimento venga contrassegnato come non riuscito, impostare un valore di timeout di recupero trasferimento per il periodo di tempo, in secondi, in cui si desidera che l'agent continui a ritentare.

  Ad esempio, l'impostazione di un valore di timeout del ripristino del trasferimento `21600` significa che l'agent continua a tentare di ripristinare il trasferimento per sei ore da quando entra nel ripristino:

  ```
  transferRecoveryTimeout=21600
  ```

  Il valore massimo per questo parametro è 999999999.

Impostazione del timeout di ripristino del trasferimento per singoli trasferimenti

È possibile impostare il timeout di ripristino del trasferimento per un singolo trasferimento dalla riga comandi o con IBM MQ Exploreroppure utilizzando le attività Apache Ant . Se è impostato un valore di timeout di ripristino trasferimento nel file `agent.properties` , l'impostazione del timeout di ripristino trasferimento per un singolo trasferimento sovrascrive il valore impostato nel file `agent.properties` .

Informazioni su questa attività

È possibile impostare il parametro di timeout del ripristino del trasferimento per un singolo trasferimento quando:

- Creazione di un trasferimento utilizzando il comando **fteCreateTransfer** o utilizzando IBM MQ Explorer.
- Creazione di un modello di trasferimento utilizzando il comando **fteCreateTemplate** o utilizzando IBM MQ Explorer.
- Creazione di un monitoraggio delle risorse utilizzando il comando **fteCreateMonitor** o utilizzando IBM MQ Explorer.
- Copia o spostamento di file utilizzando le attività fte: filecopy o fte: filemove Ant .

Se si imposta un valore di timeout di ripristino del trasferimento per un singolo trasferimento, questo valore sovrascrive il valore di timeout del ripristino del trasferimento impostato nel file `agent.properties` (consultare“Impostazione del timeout di ripristino del trasferimento per tutti i trasferimenti per un agent di origine” a pagina 334).

Procedura

- Per utilizzare il comando **fteCreateTransfer** o **fteCreateTemplate** per impostare il timeout di ripristino del trasferimento, specificare l'opzione appropriata per il parametro **-rt** :

  -1
  L'agent continua a tentare di recuperare il trasferimento bloccato fino a quando il trasferimento non viene eseguito correttamente. L'uso di questa opzione equivale al comportamento predefinito dell'agente quando la proprietà non è impostata.

  0
  L'agent arresta il trasferimento file non appena avvia il ripristino.

  >0
  L'agent continua a tentare di ripristinare il trasferimento bloccato per il periodo di tempo specificato in secondi.

  Esempi per il comando **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
```

  Esempi per il comando **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
```

Non esiste alcun parametro **-rt** per il comando **fteCreateMonitor** . Se si imposta il parametro **-rt** con il comando **fteCreateTransfer** e anche il parametro **-gt** , il parametro di timeout di recupero viene incluso nel documento XML con la definizione di trasferimento generata quando si esegue il comando **fteCreateTransfer** . Il controllo risorse utilizza questo documento XML quando si esegue il comando **fteCreateMonitor** . Nel seguente esempio, i dettagli del timeout di ripristino del trasferimento vengono inclusi nel file `task.xml` :

```
fteCreateMonitor -ma AgentName -md C:\mqmft\monitors -mn Monitor_Name -mt task.xml -tr
"fileSize>=5MB,*.zip"
```

- Per utilizzare la pagina della procedura guidata IBM MQ Explorer Nuovo trasferimento, Nuovo monitoraggio o Nuovo template per impostare il timeout del ripristino del trasferimento, selezionare l'opzione richiesta nel campo **Timeout ripristino trasferimento** (secondi):

  Come agent di origine
  Se si seleziona **Come agent di origine**, il valore del parametro **transferRecoveryTimeout** dal file `agent.properties` viene utilizzato se è impostato, altrimenti viene applicato il comportamento predefinito per il timeout di ripristino del trasferimento.

  Casella di elenco numerica
  Se si immette un tempo in secondi nella casella di elenco numerica, l'agent continua a tentare di ripristinare il trasferimento in stallo per il periodo di tempo specificato.

  Nessuna
  Se si seleziona **Nessuno**, non viene impostato alcun timeout di ripristino del trasferimento e l'agent continua a tentare di ripristinare il trasferimento bloccato fino a quando il trasferimento non viene eseguito correttamente.

- Per impostare il timeout di ripristino utilizzando le attività Ant . includere l'opzione e il valore **transferRecoveryTimeout** , con gli elementi **fte:filecopy** o **fte:filemove** per spostare o copiare i file, ad esempio:

  Esempio per **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>
```

  Esempio per **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>
```

Concetti correlati

“Ripristino e riavvio di MFT” a pagina 330
Se l'agent o il gestore code non sono disponibili per qualsiasi motivo, ad esempio a causa di un errore di alimentazione o di rete, Managed File Transfer esegue il ripristino come riportato di seguito in questi scenari:

Attività correlate

“Avvio di un nuovo trasferimento file” a pagina 218
È possibile avviare un nuovo trasferimento file da IBM MQ Explorer o dalla riga comandi ed è possibile scegliere di trasferire un singolo file o più file in un gruppo.

“Creazione di un modello di trasferimento file utilizzando IBM MQ Explorer” a pagina 263
È possibile creare un modello di trasferimento file da IBM MQ Explorer o dalla riga comandi. È quindi possibile utilizzare tale modello per creare nuovi trasferimenti file utilizzando i relativi dettagli oppure inoltrare il modello per avviare il trasferimento file.

“Monitoraggio delle risorse MFT” a pagina 229
È possibile monitorare le risorse Managed File Transfer ; ad esempio, una coda o una directory. Quando viene soddisfatta una condizione su questa risorsa, il monitoraggio risorse avvia un'attività, ad esempio un trasferimento file. È possibile creare un monitoraggio delle risorse utilizzando il comando **fteCreateMonitor** o la vista **Monitor** nel plug-in Managed File Transfer per IBM MQ Explorer.

“Visualizzazione dello stato dei trasferimenti file nel log di trasferimento” a pagina 227
È possibile visualizzare i dettagli dei trasferimenti file utilizzando il **Log trasferimenti** in IBM MQ Explorer. Questi possono essere trasferimenti avviati dalla riga comandi o da IBM MQ Explorer. È anche possibile personalizzare quanto visualizzato nel **Log di trasferimento**.

Riferimenti correlati

Il file MFT `agent.properties`
fteCreateTransfer: avviare un nuovo trasferimento file
fteCreateTemplate: crea un nuovo modello di trasferimento file
fteCreateMonitor: crea un monitoraggio risorse MFT
fte: attività Ant filecopy
fte: attività Ant filemove

Linux Windows AIX

AmministrazioneMQ Telemetry

MQ Telemetry viene gestito utilizzando IBM MQ Explorer o su una riga comandi. Utilizzare explorer per configurare i canali di telemetria, controllare il servizio di telemetria e monitorare i client MQTT connessi a IBM MQ. Configurare la sicurezza di MQ Telemetry utilizzando JAAS, TLS e l'object authority manager IBM MQ .

Amministrazione mediante IBM MQ Explorer

Utilizzare explorer per configurare i canali di telemetria, controllare il servizio di telemetria e monitorare i client MQTT connessi a IBM MQ. Configurare la sicurezza di MQ Telemetry utilizzando JAAS, TLS e l'object authority manager IBM MQ .

Gestione mediante la riga comandi

MQ Telemetry può essere completamente gestito dalla riga di comando utilizzando comandi MQSC.

La documentazione MQ Telemetry contiene anche script di esempio che dimostrano l'utilizzo di base dell'applicazione client IBM MQ Telemetry Transport v3 .

Prima di utilizzarli, leggere e comprendere gli esempi in IBM MQ Telemetry Transport programmi di esempio .

Concetti correlati

MQ Telemetry

Riferimenti correlati

Proprietà MQXR

Linux AIX Configurazione di un gestore code per la telemetria su Linux e AIX

Seguire questa procedura per configurare MQ Telemetry manualmente. Se è necessaria solo una semplice configurazione che utilizza l'ID utente guest, è possibile invece eseguire la procedura guidata di supporto MQ Telemetry in IBM MQ Explorer.

Prima di iniziare

Se hai bisogno solo di una semplice configurazione, considera l'utilizzo del supporto MQ Telemetry in IBM MQ Explorer. Questo supporto include una procedura guidata e una procedura di comando di esempio `sampleMQM`. Queste risorse impostano una configurazione iniziale utilizzando l'ID utente guest. Consultare Verifica dell'installazione di MQ Telemetry utilizzando i programmi di esempio IBM MQ Explorer e IBM MQ Telemetry Transport.

Se è necessaria una configurazione più complessa che utilizza un metodo di autenticazione diverso, utilizzare la procedura in questa attività. Iniziare con i seguenti passi iniziali:

1. Consultare Considerazioni sull'installazione per MQ Telemetry per informazioni su come installare IBM MQe la funzione MQ Telemetry .
2. Creare e avviare un gestore code. Il gestore code viene indicato come *qMgr* in questa attività.
3. Come parte di questa attività, configurare il servizio di telemetria (MQXR). Le impostazioni della proprietà MQXR sono memorizzate in un file delle proprietà specifico della piattaforma: `mqxr_win.properties`. Normalmente non è necessario modificare direttamente il file delle proprietà MQXR, poiché quasi tutte le impostazioni possono essere configurate tramite i comandi di gestione MQSC o IBM MQ Explorer. Se si decide di modificare direttamente il file, arrestare il gestore code prima di apportare le modifiche. Vedere Proprietà MQXR.

Informazioni su questa attività

Seguire i passi di questa attività per configurare MQ Telemetry manualmente, utilizzando schemi di autorizzazione diversi.

Procedura

1. Aprire una finestra comandi nella directory degli esempi di telemetria.

   La directory degli esempi di telemetria è `/opt/mqm/mqxr/samples`.
2. Creare la coda di trasmissione di telemetria.

   Se `SYSTEM.MQTT.TRANSMIT.QUEUE` non esiste, viene creato automaticamente quando il servizio di telemetria (MQXR) viene avviato per la prima volta e impostato per utilizzare l'ID utente guest. Tuttavia, questa attività configura MQ Telemetry per utilizzare uno schema di autorizzazione differente. Per questa attività, creare `SYSTEM.MQTT.TRANSMIT.QUEUE`e configurare l'accesso ad esso prima di avviare il servizio di telemetria (MQXR).

   Esegui il seguente comando:

   ```
   echo "DEFINE QLOCAL('SYSTEM.MQTT.TRANSMIT.QUEUE') USAGE(XMITQ) MAXDEPTH(100000)" | runmqsc
   qMgr
   ```
3. Impostare la coda di trasmissione predefinita.

   È più semplice inviare messaggi direttamente ai client MQTT se `SYSTEM.MQTT.TRANSMIT.QUEUE` è la coda di trasmissione predefinita. In caso contrario, è necessario aggiungere una definizione di coda remota per ogni client che riceve messaggi IBM MQ ; consultare “Invio diretto di un messaggio ad un cliente” a pagina 343. Si noti che la modifica della coda di trasmissione predefinita potrebbe interferire con la propria configurazione esistente.

Quando il servizio di telemetria (MQXR) viene avviato per la prima volta, non imposta SYSTEM.MQTT.TRANSMIT.QUEUE come coda di trasmissione predefinita per il gestore code. Per configurare questa impostazione, modificare la proprietà della coda di trasmissione predefinita. A tale scopo, utilizzare IBM MQ Exploreroppure eseguire il seguente comando:

```
echo "ALTER QMGR DEFXMITQ('SYSTEM.MQTT.TRANSMIT.QUEUE')" | runmqsc qMgr
```

4. Seguire una procedura in "Autorizzazione dei client MQTT ad accedere agli oggetti di IBM MQ" a pagina 345 per creare uno o più ID utente. Gli ID utente hanno l'autorità di pubblicare, sottoscrivere e inviare pubblicazioni ai client MQTT .
5. Modificare il file installMQXRService_unix.mqsc per configurare il file di chiavi utilizzato per codificare la passphrase per i canali MQTT TLS:
   a) Aprire il file *WMQ program installation directory*/mqxr/samples/installMQXRService_unix.mqsc.
   b) Individuare la riga che include il parametro **STARTARG**e modificare l'opzione **-sf** per specificare l'ubicazione del file di chiavi delle credenziali.

      Per default, il file installMQXRService_unix.mqsc utilizza un file di chiavi predefinito denominato [DEFAULT]. Il file di chiavi predefinito è lo stesso per tutte le installazioni IBM MQ , quindi è necessario fornire un file di chiavi univoco per l'installazione quando si codificano le passphrase.

      Vedere anche il codice di esempio in "Creazione del SYSTEM.MQXR.SERVICE" a pagina 339.
6. Installare il servizio di telemetria (MQXR) eseguendo il seguente comando:

```
cat /opt/<install_dir>/mqxr/samples/installMQXRService_unix.mqsc | runmqsc queue_manager
```

7. Avviare il servizio.

```
echo "START SERVICE(SYSTEM.MQXR.SERVICE)" | runmqsc qMgr
```

   Il servizio di telemetria (MQXR) viene avviato automaticamente all'avvio del gestore code. Viene avviato manualmente in questa attività perché il gestore code è già in esecuzione.
8. Utilizzando IBM MQ Explorer, configurare i canali di telemetria per accettare le connessioni dai client MQTT .

   I canali di telemetria devono essere configurati in modo che le loro identità siano uno degli ID utente definiti nel passo "4" a pagina 339.

   Vedere anche DEFINE CHANNEL (MQTT).
9. Verificare la configurazione eseguendo il client di esempio.

   Per consentire al client di esempio di utilizzare il proprio canale di telemetria, il canale deve autorizzare il client a pubblicare, sottoscrivere e ricevere pubblicazioni. Il client di esempio si collega al canale di telemetria sulla porta 1883 per impostazione predefinita. Vedere inoltre IBM MQ Telemetry Transport programmi di esempio.

Creazione del SYSTEM.MQXR.SERVICE

Utilizzare il comando **runMQXRService** per creare il 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 Configurazione di un gestore code per la telemetria su Windows

Seguire questa procedura per configurare MQ Telemetry manualmente. Se è necessaria solo una semplice configurazione che utilizza l'ID utente guest, è possibile invece eseguire la procedura guidata di supporto MQ Telemetry in IBM MQ Explorer.

Prima di iniziare

Se hai bisogno solo di una semplice configurazione, considera l'utilizzo del supporto MQ Telemetry in IBM MQ Explorer. Questo supporto include una procedura guidata e una procedura di comando di esempio `sampleMQM`. Queste risorse impostano una configurazione iniziale utilizzando l'ID utente guest. Consultare Verifica dell'installazione di MQ Telemetry utilizzando i programmi di esempio IBM MQ Explorer e IBM MQ Telemetry Transport.

Se è necessaria una configurazione più complessa che utilizza un metodo di autenticazione diverso, utilizzare la procedura in questa attività. Iniziare con i seguenti passi iniziali:

1. Consultare Considerazioni sull'installazione per MQ Telemetry per informazioni su come installare IBM MQe la funzione MQ Telemetry .
2. Creare e avviare un gestore code. Il gestore code viene indicato come *qMgr* in questa attività.
3. Come parte di questa attività, configurare il servizio di telemetria (MQXR). Le impostazioni della proprietà MQXR sono memorizzate in un file delle proprietà specifico della piattaforma: `mqxr_win.properties`. Normalmente non è necessario modificare direttamente il file delle proprietà MQXR, poiché quasi tutte le impostazioni possono essere configurate tramite i comandi di gestione MQSC o IBM MQ Explorer. Se si decide di modificare direttamente il file, arrestare il gestore code prima di apportare le modifiche. Vedere Proprietà MQXR.

Informazioni su questa attività

Seguire i passi di questa attività per configurare MQ Telemetry manualmente, utilizzando schemi di autorizzazione diversi.

Procedura

1. Aprire una finestra comandi nella directory degli esempi di telemetria.

   La directory degli esempi di telemetria è *WMQ program installation directory*`\mqxr\samples`.
2. Creare la coda di trasmissione di telemetria.

   Se `SYSTEM.MQTT.TRANSMIT.QUEUE` non esiste, viene creato automaticamente quando il servizio di telemetria (MQXR) viene avviato per la prima volta e impostato per utilizzare l'ID utente guest. Tuttavia, questa attività configura MQ Telemetry per utilizzare uno schema di autorizzazione differente. Per questa attività, creare `SYSTEM.MQTT.TRANSMIT.QUEUE`e configurare l'accesso ad esso prima di avviare il servizio di telemetria (MQXR).

   Esegui il seguente comando:

   ```
   echo DEFINE QLOCAL('SYSTEM.MQTT.TRANSMIT.QUEUE') USAGE(XMITQ) MAXDEPTH(100000) | runmqsc qMgr
   ```
3. Impostare la coda di trasmissione predefinita.

   È più semplice inviare messaggi direttamente ai client MQTT se `SYSTEM.MQTT.TRANSMIT.QUEUE` è la coda di trasmissione predefinita. In caso contrario, è necessario aggiungere una definizione di coda remota per ogni client che riceve messaggi IBM MQ ; consultare “Invio diretto di un messaggio ad un cliente” a pagina 343. Si noti che la modifica della coda di trasmissione predefinita potrebbe interferire con la propria configurazione esistente.

Quando il servizio di telemetria (MQXR) viene avviato per la prima volta, non imposta SYSTEM.MQTT.TRANSMIT.QUEUE come coda di trasmissione predefinita per il gestore code. Per configurare questa impostazione, modificare la proprietà della coda di trasmissione predefinita. A tale scopo, utilizzare IBM MQ Exploreroppure eseguire il seguente comando:

```
echo ALTER QMGR DEFXMITQ('SYSTEM.MQTT.TRANSMIT.QUEUE') | runmqsc qMgr
```

4. Seguire una procedura in "Autorizzazione dei client MQTT ad accedere agli oggetti di IBM MQ" a pagina 345 per creare uno o più ID utente. Gli ID utente hanno l'autorità di pubblicare, sottoscrivere e inviare pubblicazioni ai client MQTT .
5. Modificare il file `installMQXRService_win.mqsc` per configurare il file di chiavi utilizzato per codificare la passphrase per i canali MQTT TLS:
   a) Aprire il file *WMQ program installation directory*`\mqxr\samples\installMQXRService_win.mqsc`.
   b) Individuare la riga che include il parametro **STARTARG**e modificare l'opzione **-sf** per specificare l'ubicazione del file di chiavi delle credenziali.

      Per default, il file `installMQXRService_win.mqsc` utilizza un file di chiavi predefinito denominato `[DEFAULT]`. Il file di chiavi predefinito è lo stesso per tutte le installazioni IBM MQ , quindi è necessario fornire un file di chiavi univoco per l'installazione quando si codificano le passphrase.

      Vedere anche il codice di esempio in "Creazione SYSTEM.MQXR.SERVICE" a pagina 341.
6. Installare il servizio di telemetria (MQXR) eseguendo il seguente comando:

```
type installMQXRService_win.mqsc | runmqsc qMgr
```

7. Avviare il servizio.

```
echo START SERVICE(SYSTEM.MQXR.SERVICE) | runmqsc qMgr
```

   Il servizio di telemetria (MQXR) viene avviato automaticamente all'avvio del gestore code. Viene avviato manualmente in questa attività perché il gestore code è già in esecuzione.
8. Utilizzando IBM MQ Explorer, configurare i canali di telemetria per accettare le connessioni dai client MQTT .

   I canali di telemetria devono essere configurati in modo che le loro identità siano uno degli ID utente definiti nel passo "4" a pagina 341.

   Vedere anche DEFINE CHANNEL (MQTT).
9. Verificare la configurazione eseguendo il client di esempio.

   Per consentire al client di esempio di utilizzare il proprio canale di telemetria, il canale deve autorizzare il client a pubblicare, sottoscrivere e ricevere pubblicazioni. Il client di esempio si collega al canale di telemetria sulla porta 1883 per impostazione predefinita. Vedere inoltre IBM MQ Telemetry Transport programmi di esempio.

Creazione SYSTEM.MQXR.SERVICE

Utilizzare il comando **runMQXRService** per creare il 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')
```

Linux Windows AIX Configurazione dell'accodamento distribuito per inviare messaggi ai client MQTT

Le applicazioni IBM MQ possono inviare i messaggi dei client MQTT v3 pubblicando su una sottoscrizione creata da un client o inviando direttamente un messaggio. Indipendentemente dal metodo utilizzato, il messaggio viene inserito in SYSTEM.MQTT.TRANSMIT.QUEUEe inviato al client dal servizio di telemetria (MQXR). Esistono diversi modi per inserire un messaggio su SYSTEM.MQTT.TRANSMIT.QUEUE.

Pubblicazione di un messaggio in risposta a una sottoscrizione client MQTT

Il servizio di telemetria (MQXR) crea una sottoscrizione per conto del client MQTT . Il client è la destinazione per tutte le pubblicazioni che corrispondono alla sottoscrizione inviata dal client. I servizi di telemetria inoltrano le pubblicazioni corrispondenti al client.

Un client MQTT è connesso a IBM MQ come gestore code, con il proprio nome gestore code impostato su `ClientIdentifier`. La destinazione delle pubblicazioni da inviare al client è una coda di trasmissione, SYSTEM.MQTT.TRANSMIT.QUEUE. Il servizio di telemetria inoltra i messaggi su SYSTEM.MQTT.TRANSMIT.QUEUE ai clienti MQTT , utilizzando il nome del gestore code di destinazione come chiave per un client specifico.

Il servizio di telemetria (MQXR) apre la coda di trasmissione utilizzando `ClientIdentifier` come nome gestore code. Il servizio di telemetria (MQXR) passa l'handle dell'oggetto della coda alla chiamata MQSUB per inoltrare le pubblicazioni che corrispondono alla sottoscrizione client. Nella risoluzione del nome oggetto, il `ClientIdentifier` viene creato come nome del gestore code remoto e la coda di trasmissione deve risolversi in SYSTEM.MQTT.TRANSMIT.QUEUE. Utilizzando la risoluzione del nome oggetto IBM MQ standard, *`ClientIdentifier`* viene risolto come segue; consultare Tabella 16 a pagina 343.

1. *`ClientIdentifier`* non corrisponde a nulla.

   `ClientIdentifier` è un nome gestore code remoto. Non corrisponde al nome del gestore code locale, a un alias del gestore code o a un nome della coda di trasmissione.

   Il nome della coda non è definito. Attualmente, il servizio di telemetria (MQXR) imposta SYSTEM.MQTT.PUBLICATION.QUEUE come nome della coda. Un client MQTT v3 non supporta le code, pertanto il nome della coda risolta viene ignorato dal client.

   La proprietà del gestore code locale, `Coda di trasmissione predefinita`, deve essere impostata su SYSTEM.MQTT.TRANSMIT.QUEUE, in modo che la pubblicazione venga inserita in SYSTEM.MQTT.TRANSMIT.QUEUE per essere inviata al client.

2. *`ClientIdentifier`* corrisponde a un alias del gestore code denominato *`ClientIdentifier`*.

   `ClientIdentifier` è un nome gestore code remoto. Corrisponde al nome di un alias del gestore code.

   L'alias del gestore code deve essere definito con *`ClientIdentifier`* come nome del gestore code remoto.

   Impostando il nome della coda di trasmissione nella definizione alias del gestore code, non è necessario che la trasmissione predefinita sia impostata su SYSTEM.MQTT.TRANSMIT.QUEUE.

Tabella 16. Risoluzione del nome di un alias del gestore code MQTT					
	Immissione		**Output**		
ClientIdentifier	**Nome del gestore code**	**Nome coda**	**Nome del gestore code**	**Nome coda**	**Coda di trasmissione**
Non corrisponde a nulla	*ClientIdentifier*	*non definito*	*ClientIdentifier*	*non definito*	Coda di trasmissione predefinita. SYSTEM.MQTT.TRANSMIT.QUEUE
Corrisponde a un alias del gestore code denominato *ClientIdentifier*	*ClientIdentifier*	*non definito*	*ClientIdentifier*	*non definito*	SYSTEM.MQTT.TRANSMIT.QUEUE

Per ulteriori informazioni sulla risoluzione dei nomi, consultare Risoluzione dei nomi.

Qualsiasi programma IBM MQ può eseguire la pubblicazione nello stesso argomento. La pubblicazione viene inviata ai relativi sottoscrittori, inclusi MQTT v3 client che hanno una sottoscrizione all'argomento.

Se un argomento di gestione viene creato in un cluster, con l'attributo CLUSTER(*clusterName*), qualsiasi applicazione nel cluster può pubblicare sul client; ad esempio:

```
echo DEFINE TOPIC('MQTTExamples') TOPICSTR('MQTT Examples') CLUSTER(MQTT) REPLACE | runmqsc qMgr
```

Nota: Non fornire a SYSTEM.MQTT.TRANSMIT.QUEUE un attributo cluster.

I publisher e i sottoscrittori del client MQTT possono connettersi a diversi gestori code. I sottoscrittori e i publisher possono far parte dello stesso cluster o essere collegati da una gerarchia di pubblicazione / sottoscrizione. La pubblicazione viene consegnata dal publisher al sottoscrittore utilizzando IBM MQ.

Invio diretto di un messaggio ad un cliente

Un'alternativa a un client che crea una sottoscrizione e riceve una pubblicazione che corrisponde all'argomento della sottoscrizione, invia un messaggio direttamente a un client MQTT v3 . Le applicazioni client MQTT V3 non possono inviare messaggi direttamente, ma altre applicazioni, come le applicazioni IBM MQ , possono farlo.

L'applicazione IBM MQ deve conoscere il ClientIdentifier del client MQTT v3 . Poiché i client MQTT v3 non hanno code, il nome della coda di destinazione viene passato al metodo MQTT v3 application client messageArrived come un nome argomento. Ad esempio, in un programma MQI, creare un descrittore oggetto con il client come ObjectQmgrName:

```
MQOD.ObjectQmgrName = ClientIdentifier ;
MQOD.ObjectName = name ;
```

Se l'applicazione viene scritta utilizzando JMS, creare una destinazione point-to-point; ad esempio:

```
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");
```

Per inviare un messaggio non richiesto a un client MQTT utilizzare una definizione di coda remota. Il nome del gestore code remoto deve essere risolto nel `ClientIdentifier` del client. La coda di trasmissione deve essere risolta in `SYSTEM.MQTT.TRANSMIT.QUEUE` ; consultare Tabella 17 a pagina 344. Il nome della coda remota può essere qualsiasi cosa. Il client lo riceve come stringa di argomenti.

Tabella 17. Risoluzione del nome di una definizione di coda remota del client MQTT

Immissione		Output		
Nome coda	**Nome del gestore code**	**Nome coda**	**Nome del gestore code**	**Coda di trasmissione**
Nome della definizione della coda remota	Nome gestore code locale o vuoto	Nome coda remota utilizzato come stringa argomento	*ClientIdentifier*	`SYSTEM.MQTT.TRANSMIT.QUEUE`

Se il client è connesso, il messaggio viene inviato direttamente al client MQTT , che richiama il metodo `messageArrived` ; consultare `messageArrived` method.

Se il client si è scollegato da una sessione persistente, il messaggio viene memorizzato in `SYSTEM.MQTT.TRANSMIT.QUEUE` ; vedere MQTT sessioni stateless e stateful. Viene inoltrato al cliente quando il client si riconnette di nuovo alla sessione.

Se si invia un messaggio non persistente, questo viene inviato al client con `al massimo` QoS (quality of service) `QoS=0`. Se si invia un messaggio persistente direttamente ad un client, per impostazione predefinita, viene inviato con `esattamente una volta` QoS (quality of service), `QoS=2`. Poiché il client potrebbe non avere un meccanismo di persistenza, il client può ridurre la qualità del servizio che accetta per i messaggi inviati direttamente. Per ridurre la qualità del servizio per i messaggi inviati direttamente a un client, sottoscrivere l'argomento `DEFAULT.QoS`. Specificare la qualità massima del servizio che il client può supportare.

Linux Windows AIX Autenticazione, autorizzazione e identificazione del client MQTT

Il servizio di telemetria (MQXR) pubblica o sottoscrive gli argomenti IBM MQ per conto dei client MQTT , utilizzando i canali MQTT . L'amministratore IBM MQ configura l'identità del canale MQTT utilizzata per l'autorizzazione IBM MQ . L'amministratore può definire un'identità comune per il canale oppure utilizzare il `Nome utente` o `ClientIdentifier` di un client connesso al canale.

Il servizio di telemetria (MQXR) può autenticare il client utilizzando il `Nome utente` fornito dal client o utilizzando un certificato client. Il `Nome utente` viene autenticato utilizzando una password fornita dal client.

Per riassumere: l'identificazione del client è la selezione dell'identità del client. In base al contesto, il client viene identificato da `ClientIdentifier`, `Username`, un'identità client comune creata dall'amministratore o un certificato client. L'identificativo client utilizzato per il controllo di autenticità non deve essere lo stesso identificativo utilizzato per l'autorizzazione.

I programmi client MQTT impostano `Nome utente` e `Password` che vengono inviati al server utilizzando un canale MQTT . Possono anche impostare le proprietà TLS richieste per codificare e autenticare la connessione. L'amministratore decide se autenticare il canale MQTT e come autenticare il canale.

Per autorizzare un client MQTT ad accedere agli oggetti IBM MQ , autorizzare `ClientIdentifier`o `Username` del client oppure autorizzare un'identità client comune. Per consentire a un client di connettersi a IBM MQ, autenticare il `Nome utente`o utilizzare un certificato client. Configurare JAAS per autenticare il `Nome utente`e configurare TLS per autenticare un certificato client.

Se imposti una Password sul client, crittografa la connessione utilizzando VPN o configura il canale MQTT per utilizzare TLS, per mantenere la password privata.

È difficile gestire i certificati client. Per questo motivo, se i rischi associati all'autenticazione della password sono accettabili, l'autenticazione della password viene spesso utilizzata per autenticare i client.

Se esiste un modo sicuro per gestire e memorizzare il certificato client, è possibile fare affidamento sull'autenticazione del certificato. Tuttavia, raramente i certificati possono essere gestiti in modo sicuro nei tipi di ambienti in cui viene utilizzata la telemetria. Invece, l'autenticazione delle unità che utilizzano i certificati client è completata dall'autenticazione delle password client sul server. A causa della complessità aggiuntiva, l'utilizzo dei certificati client è limitato alle applicazioni altamente sensibili. L'uso di due forme di autenticazione è chiamato autenticazione a due fattori. È necessario conoscere uno dei fattori, ad esempio una password, e l'altro, ad esempio un certificato.

In un'applicazione altamente sensibile, ad esempio un dispositivo con chip e pin, il dispositivo viene bloccato durante la fabbricazione per evitare la manomissione dell'hardware e del software interni. Un certificato client sicuro, limitato nel tempo, viene copiato sul dispositivo. Il dispositivo viene distribuito nell'ubicazione in cui deve essere utilizzato. Un'ulteriore autenticazione viene eseguita ogni volta che il dispositivo viene utilizzato, utilizzando una parola d'ordine o un altro certificato da una smart card.

Linux Windows AIX Identità e autorizzazione del client MQTT

Utilizzare l'ID client, Nome utente, o un'identità client comune per l'autorizzazione ad accedere agli oggetti IBM MQ .

L'amministratore IBM MQ ha tre opzioni per selezionare l'identità del canale MQTT . L'amministratore effettua la scelta quando definisce o modifica il canale MQTT utilizzato dal client. L'identità viene utilizzata per autorizzare l'accesso agli argomenti IBM MQ . La scelta viene effettuata nel seguente ordine:

1. L'ID client (vedere USECLNTID).
2. Un'identità fornita dall'amministratore per il canale (MCAUSER del canale. Vedere MCAUSER).
3. Se nessuna delle scelte precedenti si applica, il Nome utente passato dal client MQTT (Nome utente è un attributo della classe MqttConnectOptions . Deve essere impostato prima che il client si colleghi al servizio. Il valore predefinito è null).

Prevenzione dei problemi: L'identità scelta da questo processo viene successivamente indicata, ad esempio dal comando DISPLAY CHSTATUS (MQTT), come MCAUSER del client. Tenere presente che questa non è necessariamente la stessa identità del MCAUSER del canale a cui si fa riferimento nella scelta (2).

Utilizzare il comando IBM MQ **setmqaut** per selezionare quali oggetti e quali azioni sono autorizzati ad essere utilizzati dall'identit. associata al canale MQTT . Ad esempio, il seguente codice autorizza un'identità del canale MQTTClient, fornita dall'amministratore del gestore code 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
```

Linux Windows AIX *Autorizzazione dei client MQTT ad accedere agli oggetti di IBM MQ*

Attenersi alla seguente procedura per autorizzare i client di MQTT a pubblicare e sottoscrivere oggetti IBM MQ . La procedura segue quattro modelli di controllo accessi alternativi.

Prima di iniziare

I client MQTT sono autorizzati ad accedere agli oggetti in IBM MQ mediante l'assegnazione di un'identità quando si collegano ad un canale di telemetria. L'amministratore IBM MQ configura il canale di telemetria utilizzando Esplora risorse di IBM MQ per fornire a un client uno dei tre tipi di identità:

1. ClientIdentifier
2. Nome utente

3. Un nome che l'amministratore assegna al canale.

Indipendentemente dal tipo utilizzato, l'identità deve essere definita in IBM MQ come principal dal servizio di autorizzazione installato. Il servizio di autorizzazione predefinito su Windows o Linux è denominato OAM (Object Authority Manager). Se si utilizza OAM, l'identità deve essere definita come ID utente.

Utilizzare l'identità per fornire a un client, o a una raccolta di client, l'autorizzazione a pubblicare o sottoscrivere gli argomenti definiti in IBM MQ. Se un client MQTT ha sottoscritto un argomento, utilizzare l'identità per fornire l'autorizzazione a ricevere le pubblicazioni risultanti.

È difficile gestire un sistema con decine di migliaia di client MQTT , ognuno dei quali richiede autorizzazioni di accesso individuali. Una soluzione è definire identità comuni e associare singoli client MQTT a una delle identità comuni. Definire tutte le identità comuni necessarie per definire diverse combinazioni di autorizzazioni. Un'altra soluzione è scrivere il proprio servizio di autorizzazione che può trattare più facilmente con migliaia di utenti rispetto al sistema operativo.

È possibile combinare i client MQTT in identità comuni in due modi, utilizzando OAM:

1. Definire più canali di telemetria, ciascuno con un diverso ID utente assegnato dall'amministratore tramite Esplora risorse di IBM MQ . I client che si collegano utilizzando numeri di porta TCP/IP differenti sono associati a canali di telemetria differenti e sono assegnati a identità differenti.
2. Definire un singolo canale di telemetria, ma ciascun client deve selezionare un `Nome utente` da una piccola serie di ID utente. L'amministratore configura il canale di telemetria per selezionare il client `Nome utente` come sua identità.

In questa attività, l'identità del canale di telemetria viene denominata *`mqttUser`*, indipendentemente da come è impostata. Se le raccolte di client utilizzano identità differenti, utilizzare più *`mqttUsers`*, uno per ogni raccolta di client. Poiché l'attività utilizza OAM, ogni *`mqttUser`* deve essere un ID utente.

Informazioni su questa attività

In questa attività, è possibile scegliere tra quattro modelli di controllo accessi che è possibile personalizzare in base a requisiti specifici. I modelli differiscono nella loro granularità del controllo accessi.

- “Nessun controllo accessi” a pagina 346
- “Controllo degli accessi generici” a pagina 346
- “Controllo accessi a grana media” a pagina 347
- “Controllo dell'accesso dettagliato” a pagina 347

Il risultato dei modelli è quello di assegnare *`mqttUsers`* serie di autorizzazioni per pubblicare e sottoscrivere IBM MQe ricevere pubblicazioni da IBM MQ.

Nessun controllo accessi
Ai client MQTT viene fornita l'autorizzazione di gestione IBM MQ e può eseguire qualsiasi azione su qualsiasi oggetto.

Procedura

1. Creare un ID utente *`mqttUser`* per agire come identità di tutti i client MQTT .
2. Aggiungere *`mqttUser`* al gruppo `mqm` ; consultare Aggiunta di un utente a un gruppo su Windows o Creazione e gestione di gruppi su Linux

Controllo degli accessi generici
I client MQTT hanno l'autorità per pubblicare e sottoscrivere e per inviare messaggi ai client MQTT . Non dispongono dell'autorizzazione per eseguire altre azioni o per accedere ad altri oggetti.

Procedura

1. Creare un ID utente *mqttUser* per agire come identità di tutti i client MQTT .
2. Autorizzare *mqttUser* a pubblicare e sottoscrivere tutti gli argomenti e a inviare pubblicazioni ai client 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
```

Controllo accessi a grana media
I client MQTT sono divisi in gruppi differenti per pubblicare e sottoscrivere diverse serie di argomenti e per inviare messaggi a client MQTT .

Procedura

1. Creare più ID utente, *mqttUsers*e più argomenti di gestione nella struttura ad albero degli argomenti di pubblicazione / sottoscrizione.
2. Autorizzare *mqttUsers* differenti per argomenti differenti.

```
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. Creare un gruppo *mqtt*e aggiungere tutti i *mqttUsers* al gruppo.
4. Autorizzare *mqtt* a inviare argomenti ai client MQTT .

```
setmqaut -m qMgr -t q -n SYSTEM.MQTT.TRANSMIT.QUEUE -p mqtt -all +put
```

Controllo dell'accesso dettagliato
I client MQTT sono incorporati in un sistema di controllo degli accessi esistente, che autorizza i gruppi ad eseguire azioni sugli oggetti.

Informazioni su questa attività

Un ID utente viene assegnato ad uno o più gruppi del sistema operativo in base alle autorizzazioni richieste. Se le applicazioni IBM MQ stanno eseguendo la pubblicazione e la sottoscrizione allo stesso spazio argomento dei client MQTT , utilizzare questo modello. I gruppi vengono indicati come `Publish` *X*, `Subscribe` *Y*e `mqtt`

Publish *X*
I membri dei gruppi `Publish` *X* possono pubblicare in *topicX*.

Subscribe *Y*
I membri dei gruppi `Subscribe` *Y* possono sottoscrivere *topicY*.

mqtt
I membri del gruppo *mqtt* possono inviare pubblicazioni ai client MQTT .

Procedura

1. Creare più gruppi `Publish` *X* e `Subscribe` *Y* assegnati a più argomenti di gestione nella struttura ad albero degli argomenti di pubblicazione / sottoscrizione.
2. Creare un gruppo `mqtt`.
3. Creare più ID utente, *mqttUsers*, e aggiungere gli utenti a uno qualsiasi dei gruppi, a seconda di ciò che sono autorizzati a fare.
4. Autorizzare diversi gruppi `Publish` *X* e `Subscribe` *X* a diversi argomenti e autorizzare il gruppo *mqtt* a inviare messaggi ai client 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
```

Linux Windows AIX Autenticazione client MQTT utilizzando una parola d'ordine

Autenticare il Nome utente utilizzando la password client. È possibile autenticare il client utilizzando un'identità diversa rispetto all'identità utilizzata per autorizzare il client a pubblicare e sottoscrivere argomenti.

Il servizio di telemetria (MQXR) utilizza JAAS per autenticare il client Username. JAAS utilizza Password fornita dal client MQTT .

L'amministratore IBM MQ decide se autenticare il Nome utenteo non autenticarlo affatto, configurando il canale MQTT a cui si connette un client. I client possono essere assegnati a canali diversi e ciascun canale può essere configurato per autenticare i propri client in modi differenti. Utilizzando JAAS, è possibile configurare quali metodi devono autenticare il client e quali possono facoltativamente autenticare il client.

La scelta dell'identità per l'autenticazione non influisce sulla selezione dell'identità per l'autorizzazione. È possibile impostare un'identità comune per l'autorizzazione per comodità di gestione, ma autenticare ciascun utente per utilizzare tale identità. La seguente procedura descrive la procedura per autenticare i singoli utenti per utilizzare un'identità comune:

1. L'amministratore IBM MQ imposta l'identità del canale MQTT su qualsiasi nome, ad esempio MQTTClientUser, utilizzando Esplora risorse di IBM MQ .
2. L'amministratore IBM MQ autorizza MQTTClient a pubblicare e sottoscrivere qualsiasi argomento:

```
 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. Lo sviluppatore dell'applicazione client MQTT crea un oggetto MqttConnectOptions e imposta Nome utente e Password prima di connettersi al server.
4. Lo sviluppatore della sicurezza crea un JAAS LoginModule per autenticare il nome utente con Password e lo include nel file di configurazione JAAS .
5. L'amministratore IBM MQ configura il canale MQTT per autenticare il UserName del client utilizzando JAAS.

Linux Windows AIX Autenticazione client MQTT tramite TLS

Le connessioni tra il client MQTT e il gestore code vengono sempre avviate dal client MQTT. Il client MQTT è sempre il client SSL. L'autenticazione client del server e l'autenticazione server del client MQTT sono entrambe facoltative.

Fornire al client un certificato digitale firmato privato, è possibile autenticare il client MQTT su IBM MQ. L'amministratore IBM MQ può forzare i client MQTT ad autenticarsi sul gestore code utilizzando TLS. È possibile richiedere l'autenticazione client solo come parte dell'autenticazione reciproca.

Come alternativa all'utilizzo di SSL, alcuni tipi di VPN (Virtual Private Network), ad esempio IPsec, autenticano gli endpoint di una connessione TCP/IP. VPN codifica ogni pacchetto IP che passa attraverso la rete. Una volta che tale connessione VPN è stabilita, è stata creata una rete attendibile. È possibile connettere i client MQTT ai canali di telemetria utilizzando TCP/IP sulla rete VPN.

L'autenticazione client tramite TLS si basa sulla presenza di un segreto nel client. Il segreto è la chiave privata del client nel caso di un certificato autofirmato o una chiave fornita da un'autorità di certificazione. La chiave viene utilizzata per firmare il certificato digitale del client. Chiunque sia in possesso della corrispondente chiave pubblica può verificare il certificato digitale. I certificati possono essere attendibili o, se sono concatenati, è possibile risalire attraverso una catena di certificati a un certificato root

attendibile. La verifica del client invia al server tutti i certificati presenti nella catena di certificati fornita dal client. Il server controlla la catena di certificati finché non trova un certificato attendibile. Il certificato attendibile è il certificato pubblico generato da un certificato autofirmato oppure un certificato root in genere emesso da un'autorità di certificazione. Come ultimo passo facoltativo, il certificato attendibile può essere confrontato con un elenco di revoche di certificati in tempo reale.

Il certificato attendibile potrebbe essere emesso da un'autorità di certificazione ed essere già incluso nell'archivio certificati JRE. Potrebbe essere un certificato autofirmato oppure un certificato che è stato aggiunto al keystore del canale di telemetria come certificato attendibile.

Nota: Il canale di telemetria include un keystore/truststore combinato che contiene sia le chiavi private per uno o più canali di telemetria, sia eventuali certificati pubblici necessari per autenticare i client. Poiché un canale SSL deve disporre di un keystore ed è lo stesso file del truststore del canale, non viene mai fatto riferimento all'archivio certificati JRE. L'implicazione è che se l'autenticazione di un client richiede un certificato root CA, è necessario posizionare il certificato root nel keystore per il canale, anche se il certificato root CA si trova già nell'archivio certificati JRE. All'archivio certificati JRE non viene mai fatto riferimento.

Occorre considerare le minacce che l'autenticazione client deve contrastare e i ruoli che il client e il server ricoprono nel contrastare le minacce. L'autenticazione del certificato client da sola non è sufficiente a impedire l'accesso non autorizzato a un sistema. Se qualcun altro ha accesso al dispositivo client, tale dispositivo non effettua necessariamente azioni autorizzate dal titolare del certificato. Non fare mai affidamento su una singola difesa contro attacchi indesiderati. Utilizzare almeno un approccio di autenticazione a due fattori e integrare il possesso di un certificato con la conoscenza di informazioni private. Ad esempio, utilizzare JAAS e autenticare il client utilizzando una password emessa dal server.

La minaccia principale al certificato client è che esso cada nelle mani sbagliate. Il certificato è contenuto in un keystore protetto da password nel client. Come viene collocato nel keystore? In che modo il client MQTT ottiene la password sul keystore? Quanto è sicura la protezione con password? I dispositivi di telemetria sono spesso facili da rimuovere e possono essere oggetto di attacchi in privato. Il dispositivo deve essere a prova di manomissione? La distribuzione e la protezione dei certificati sul lato client è notoriamente difficile: si chiama problema di gestione chiavi.

Una minaccia secondaria è che il dispositivo venga utilizzato erroneamente per accedere ai server in modi non intenzionali. Ad esempio, se l'applicazione MQTT viene manomessa, potrebbe essere possibile utilizzare un punto debole della configurazione del server utilizzando l'identità del client autenticato.

Per autenticare un client MQTT tramite SSL, configurare il canale di telemetria e il client.

Concetti correlati

“Configurazione del canale di telemetria per autenticazione client MQTT mediante TLS” a pagina 349
L'amministratore IBM MQ configura i canali di telemetria sul server. Ogni canale è configurato per accettare una connessione TCP/IP su un numero di porta diverso. I canali TLS sono configurati con l'accesso protetto da passphrase ai file chiave. Se un canale TLS è definito senza passphrase o file chiave, il canale non accetta le connessioni TLS.

Configurazione client MQTT per l'autenticazione client utilizzando TLS

Linux Windows AIX *Configurazione del canale di telemetria per autenticazione client MQTT mediante TLS*

L'amministratore IBM MQ configura i canali di telemetria sul server. Ogni canale è configurato per accettare una connessione TCP/IP su un numero di porta diverso. I canali TLS sono configurati con l'accesso protetto da passphrase ai file chiave. Se un canale TLS è definito senza passphrase o file chiave, il canale non accetta le connessioni TLS.

Impostare la proprietà, `com.ibm.mq.MQTT.ClientAuth` di un canale di telemetria TLS su `REQUIRED` per forzare tutti i client che si connettono su tale canale a fornire la prova di aver verificato i certificati digitali. I certificati client vengono autenticati utilizzando i certificati dalle autorità di certificazione, che portano a un certificato root attendibile. Se il certificato client è autofirmato o è firmato da un certificato che proviene da un'autorità di certificazione, i certificati firmati pubblicamente del client o dell'autorità di certificazione devono essere memorizzati in modo sicuro sul server.

Inserire il certificato client firmato pubblicamente o il certificato dall'autorità di certificazione nel keystore del canale di telemetria. Sul server, i certificati firmati pubblicamente vengono memorizzati nello stesso file chiave dei certificati firmati privatamente, piuttosto che in un truststore separato.

Il server verifica la firma di tutti i certificati client che vengono inviati utilizzando tutti i certificati pubblici e le suite di cifratura di cui dispone. Il server verifica la catena di chiavi. Il gestore code può essere configurato per verificare il certificato rispetto all'elenco di revoca certificati. La proprietà dell'elenco nomi di revoca del gestore code è SSLCRLNL.

Se uno qualsiasi dei certificati inviati da un client viene verificato da un certificato nel keystore del server, il client viene autenticato.

L'amministratore IBM MQ può configurare lo stesso canale di telemetria per utilizzare JAAS per controllare UserName o ClientIdentifier del client con la Passworddel client.

È possibile utilizzare lo stesso keystore per più canali di telemetria.

La verifica di almeno un certificato digitale nel keystore del client protetto da password sulla periferica autentica il client sul server. Il certificato digitale viene utilizzato solo per l'autenticazione da IBM MQ. Non viene utilizzato per verificare l'indirizzo TCP/IP del client o per impostare l'identità del client per l'autorizzazione o l'account. L'identità del client adottato dal server è il Nome utente o ClientIdentifier del client oppure un'identità creata dall'amministratore IBM MQ .

Puoi anche utilizzare le suite di cifratura TLS per l'autenticazione client. Se si prevede di utilizzare le suite di cifratura SHA-2 , consultare “Requisiti di sistema per l'utilizzo delle suite di crittografia SHA-2 con i canali MQTT” a pagina 354.

Concetti correlati

“Configurazione del canale di telemetria per l'autenticazione di canale utilizzando TLS” a pagina 351
L'amministratore IBM MQ configura i canali di telemetria sul server. Ogni canale è configurato per accettare una connessione TCP/IP su un numero di porta diverso. I canali TLS sono configurati con l'accesso protetto da passphrase ai file chiave. Se un canale TLS è definito senza passphrase o file chiave, il canale non accetta le connessioni TLS.

CipherSpecs e CipherSuites

Riferimenti correlati

DEFINE CHANNEL (MQTT)

MODIFICA CANALE (MQTT)

Linux Windows AIX Autenticazione del canale di telemetria mediante TLS

Le connessioni tra il client MQTT e il gestore code vengono sempre avviate dal client MQTT. Il client MQTT è sempre il client SSL. L'autenticazione client del server e l'autenticazione server del client MQTT sono entrambe facoltative.

Il client tenta sempre di autenticare il server, a meno che il client non sia configurato in modo da utilizzare una CipherSpec che supporta la connessione anonima. Se l'autenticazione non riesce, la connessione non viene stabilita.

Come alternativa all'utilizzo di SSL, alcuni tipi di VPN (Virtual Private Network), ad esempio IPsec, autenticano gli endpoint di una connessione TCP/IP. VPN codifica ogni pacchetto IP che passa attraverso la rete. Una volta che tale connessione VPN è stabilita, è stata creata una rete attendibile. È possibile connettere i client MQTT ai canali di telemetria utilizzando TCP/IP sulla rete VPN.

L'autenticazione server tramite SSL autentica il server al quale si sta per inviare informazioni riservate. Il client esegue i controlli che corrispondono ai certificati inviati dal server, rispetto ai certificati collocati nel relativo truststore o nel relativo cacerts archivio JRE.

L'archivio certificati JRE è un file JKS, cacerts. Si trova in JRE InstallPath\lib\security\. È installato con la password predefinita changeit. È possibile memorizzare certificati attendibili nell'archivio certificati JRE o nel truststore del client. Non è possibile utilizzare entrambi gli archivi.

Utilizzare il truststore client se si desidera mantenere i certificati pubblici che il client considera separati dai certificati utilizzati da altre applicazioni Java . Utilizzare l'archivio certificati JRE se si desidera utilizzare un archivio certificati comune per tutte le applicazioni Java in esecuzione sul client. Se si decide di utilizzare l'archivio certificati JRE, riesaminare i certificati in esso contenuti per assicurarsi che siano attendibili.

È possibile modificare la configurazione JSSE fornendo un diverso provider di trust. È possibile personalizzare un provider di trust per eseguire diversi controlli su un certificato. In alcuni ambienti OGSi che hanno utilizzato il client MQTT, l'ambiente fornisce un diverso fornitore di fiducia.

Per autenticare il canale di telemetria tramite TLS, configurare il server e il client.

Linux Windows AIX Configurazione del canale di telemetria per l'autenticazione di canale utilizzando TLS

L'amministratore IBM MQ configura i canali di telemetria sul server. Ogni canale è configurato per accettare una connessione TCP/IP su un numero di porta diverso. I canali TLS sono configurati con l'accesso protetto da passphrase ai file chiave. Se un canale TLS è definito senza passphrase o file chiave, il canale non accetta le connessioni TLS.

Memorizzare il certificato digitale del server, firmato con la chiave privata, nel keystore che il canale di telemetria utilizzerà sul server. Memorizzare i certificati nella relativa catena di chiavi nel keystore, se si desidera trasmettere la catena di chiavi al client. Configurare il canale di telemetria utilizzando IBM MQ explorer per utilizzare TLS. Fornire il percorso al keystore e la passphrase per accedere al keystore. Se non si imposta il numero di porta TCP/IP del canale, il numero di porta del canale di telemetria TLS assume il valore predefinito di 8883.

Puoi anche utilizzare le suite di cifratura TLS per l'autenticazione del canale. Se si prevede di utilizzare le suite di cifratura SHA-2 , consultare “Requisiti di sistema per l'utilizzo delle suite di crittografia SHA-2 con i canali MQTT” a pagina 354.

Importante: V 9.4.0 V 9.4.0 Da IBM MQ 9.4.0, i file stash e i repository delle chiavi CMS non vengono supportati con i canali AMQP e MQTT che utilizzano SSL/TLS. Utilizzare i repository di chiavi PKCS #12 e proteggere le password del repository di chiavi utilizzando invece il sistema di protezione password IBM MQ . È possibile creare un archivio di chiavi PKCS #12 utilizzando il seguente comando:

```
runmqakm -keydb -create -db filename.p12 -pw password -type pkcs12
```

Questo comando crea un file di repository chiavi PKCS #12 denominato *filename*.p12 protetto con la password specificata.

Concetti correlati

“Configurazione del canale di telemetria per autenticazione client MQTT mediante TLS” a pagina 349
L'amministratore IBM MQ configura i canali di telemetria sul server. Ogni canale è configurato per accettare una connessione TCP/IP su un numero di porta diverso. I canali TLS sono configurati con l'accesso protetto da passphrase ai file chiave. Se un canale TLS è definito senza passphrase o file chiave, il canale non accetta le connessioni TLS.

CipherSpecs e CipherSuites

Riferimenti correlati

DEFINISCI CANALE (MQTT)

MODIFICA CANALE (MQTT)

Esempio di configurazione del canale MQTT utilizzando autenticazione TLS

Questo esempio ti guida attraverso un esempio di configurazione di un canale MQTT che utilizza l'autenticazione TLS.

L'esempio configura un canale tra MQTT e Mosquitto.

L'esempio utilizza un contenitore Docker per IBM MQ su Red Hat Enterprise Linux e Mosquitto su CentOS, ma si applica a qualsiasi tipo di server. (CentOS è stato utilizzato per Mosquitto a causa delle titolarità del registro.)

Configurare il keystore e il canale IBM MQ per TLS unidirezionale

Importante: V 9.4.0 V 9.4.0 Da IBM MQ 9.4.0, i file stash e i repository delle chiavi CMS non vengono supportati con i canali AMQP e MQTT che utilizzano SSL/TLS. Utilizzare i repository di chiavi PKCS #12 e proteggere le password del repository di chiavi utilizzando invece il sistema di protezione password IBM MQ .

Completare i seguenti passi:

1. V 9.4.0 V 9.4.0 Creare il keystore IBM MQ :

   ```
   runmqakm -keydb -create -db mqtt.p12 -pw "passw0rd" -type p12
   ```

2. V 9.4.0 V 9.4.0 Creare un certificato personale:

   ```
   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
   ```

   È possibile utilizzare il seguente comando per confermare la creazione del certificato:

   ```
   runmqakm -cert -list -v -db mqtt.p12 -pw "passw0rd"
   ```

3. V 9.4.0 V 9.4.0 Creare il canale MQTT immettendo il comando seguente al prompt 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)
   ```

   Si noti che il canale utilizza le associazioni di cifratura Java , consultare TLS CipherSpecs e CipherSuites in IBM MQ classes for JMS.

4. Estrarre il certificato:

   ```
   runmqakm -cert -extract -db mqtt.kdb -stashed -label ibmwebspheremqmqm -target serverCert.pem
   ```

Installa Mosquitto on CentOS in un container Docker

Completa la seguente procedura per creare il contenitore Docker con Mosquitto in esecuzione in CentOS:

1. `docker pull centos`
2. `docker run -it centos /bin/bash`
3. `yum -y install epel-release`
4. `yum -y install mosquitto`

Sposta certificato firmatario in Mosquitto

Completa la seguente procedura per spostare il certificato che hai creato in IBM MQ a Mosquitto. Questi passaggi vengono eseguiti sulla macchina host Docker .

1. Visualizza gli ID contenitore in Docker:

   ```
   docker container ls
   ```

2. Copiare il file dal contenitore docker al docker del sistema locale

   ```
   cp MQ_Container_ID:/var/mqm/mqtt/serverCert.pem serverCert.pem
   ```

3. Copia il file dalla tua macchina locale nella directory root sulla macchina centOS :

```
docker cp serverCert.pem CentOS_ContainerID:/serverCert.pem
```

Pubblica con Mosquitto

Pubblicare un messaggio di test su Mosquitto utilizzando il seguente comando:

```
mosquitto_pub -h 172.17.0.2 --cafile serverCert.pem --insecure -p 8883 -i mosquittoClient -t
test -m 'test message' -d
```

Gli argomenti del comando hanno i seguenti significati:

-h
L'indirizzo IP dell'host Red Hat Enterprise Linux (può essere trovato utilizzando **nslookup**).

-- cafile
Il file contenente il certificato del firmatario.

-- non sicuro
Questa opzione viene specificata perché l'esempio utilizza un certificato autofirmato. Non utilizzare questa opzione quando si utilizzano certificati CA reali.

-p
Numero di porta.

-i
ID client.

-t
L'argomento in fase di pubblicazione.

-m
Il messaggio in fase di pubblicazione.

-d
Abilitare i messaggi di debug.

Configurare il canale MQTT per l'autenticazione TLS reciproca

Immettere il seguente comando per riconfigurare il canale di MQTT come SSLCAUTH (REQUIRED).

```
ALTER CHANNEL(MQTTDEMO) CHLTYPE(MQTT) SSLCAUTH(REQUIRED)
```

Crea una coppia chiave / certificato sul server Mosquitto e aggiungi a IBM MQ

Immettere i comandi seguenti per creare la coppia chiave / certificato su Mosquitto:

1. Utilizzare **openSSL** per creare la coppia chiave / certificato per Mosquitto:

```
openssl req -x509 -newkey rsa:4096 -keyout mosquittoKey.pem -out mosquittoCert.pem -subj "/
CN=Mosquitto"
```

2. Elenca gli ID contenitore per i contenitori:

```
docker container ls
```

3. Copiare il certificato Mosquitto nel docker del sistema locale:

```
docker cp CentOS_ContainerID:mosquittoCert.pem .
```

4. Copiare il certificato di Mosquitto in IBM MQ:

```
docker cp mosquittoCert.pem MQ_Container_ID:/var/mqm/mqtt
```

5. Aggiungere il certificato al keystore IBM MQ :

```
runmqakm -cert -add -db mqtt.kdb -stashed -file mosquittoCert.pem
```

6. Riavviare il canale MQTT .

Pubblica con Mosquitto e autenticazione reciproca

Completa la seguente procedura per pubblicare con Mosquitto utilizzando l'autenticazione reciproca.

1. Il seguente comando deve pubblicare correttamente un messaggio di test:

```
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. Il seguente comando non è in grado di pubblicare un messaggio di test e generare un messaggio di errore perché non invia un certificato personale da 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
```

Informazioni correlate

Gestione di chiavi e certificati

Linux Windows AIX Requisiti di sistema per l'utilizzo delle suite di crittografia SHA-2 con i canali MQTT

Se si utilizza una versione di Java che supporta le suite di cifratura SHA-2 , è possibile utilizzare queste suite per proteggere i canali MQTT (telemetria) e le app client.

Per IBM MQ 8.0 , che comprende il servizio di telemetria (MQXR), la versione minima di Java è Java 7 da IBM , SR6. Le suite di cifratura SHA-2 sono supportate per impostazione predefinita in Java 7 da IBM, SR4 in poi. È quindi possibile utilizzare le suite di crittografia SHA-2 con il servizio di telemetria (MQXR) per proteggere i canali MQTT (telemetria).

Se si sta eseguendo un client MQTT con un JRE differente, è necessario assicurarsi che supporti anche le suite di crittografia SHA-2 .

Concetti correlati

Servizio di telemetria (MQXR)

“Configurazione del canale di telemetria per l'autenticazione di canale utilizzando TLS” a pagina 351
L'amministratore IBM MQ configura i canali di telemetria sul server. Ogni canale è configurato per accettare una connessione TCP/IP su un numero di porta diverso. I canali TLS sono configurati con l'accesso protetto da passphrase ai file chiave. Se un canale TLS è definito senza passphrase o file chiave, il canale non accetta le connessioni TLS.

Riferimenti correlati

DEFINE CHANNEL (MQTT)

MODIFICA CANALE (MQTT)

Linux Windows AIX Privacy di pubblicazione sui canali di telemetria

La riservatezza delle pubblicazioni MQTT inviate in entrambe le direzioni attraverso i canali di telemetria è protetta utilizzando TLS per codificare le trasmissioni sulla connessione.

I client MQTT che si collegano ai canali di telemetria utilizzano TLS per proteggere la privacy delle pubblicazioni trasmesse sul canale utilizzando la codifica a chiave simmetrica. Poiché gli endpoint non sono autenticati, non puoi considerare attendibile la sola crittografia del canale. Combina la protezione della privacy con il server o l'autenticazione reciproca.

Come alternativa all'utilizzo di SSL, alcuni tipi di VPN (Virtual Private Network), ad esempio IPsec, autenticano gli endpoint di una connessione TCP/IP. VPN codifica ogni pacchetto IP che passa attraverso

la rete. Una volta che tale connessione VPN è stabilita, è stata creata una rete attendibile. È possibile connettere i client MQTT ai canali di telemetria utilizzando TCP/IP sulla rete VPN.

Per una configurazione tipica, che codifica il canale e autentica il server, consultare “Autenticazione del canale di telemetria mediante TLS” a pagina 350.

La crittografazione delle connessioni TLS senza autenticare il server espone la connessione agli attacchi man - in - the - middle. Anche se le informazioni scambiate sono protette contro le intercettazioni, non sai con chi le stai scambiando. A meno che tu non controlli la rete, sei esposto a qualcuno che intercetta le tue trasmissioni IP e si maschera come l'endpoint.

Puoi creare una connessione TLS crittografata, senza autenticare il server, utilizzando uno scambio di chiavi Diffie - Hellman CipherSpec che supporti TLS anonimo. Il segreto principale, condiviso tra client e server e utilizzato per codificare le trasmissioni TLS, viene stabilito senza scambiare un certificato server firmato privatamente.

Poiché le connessioni anonime non sono sicure, la maggior parte delle implementazioni TLS non utilizzano per impostazione predefinita CipherSpecsanonime. Se una richiesta client per la connessione TLS viene accettata da un canale di telemetria, il canale deve avere un keystore protetto da una passphrase. Per impostazione predefinita, poiché le implementazioni TLS non utilizzano CipherSpecsanonime, il keystore deve contenere un certificato firmato privatamente che il client può autenticare.

Se si utilizza CipherSpecsanonimo, il keystore del server deve esistere, ma non deve contenere alcun certificato firmato privatamente.

Un altro modo per stabilire una connessione crittografata consiste nel sostituire il provider di attendibilità sul client con la propria implementazione. Il provider di attendibilità non autentica il certificato del server, ma la connessione viene codificata.

Attenzione: Quando si utilizza TLS con MQTT è possibile utilizzare messaggi di grandi dimensioni, tuttavia, potrebbe verificarsi un possibile impatto sulle prestazioni. MQTT è ottimizzato per l'elaborazione di piccoli messaggi (in genere tra 1KB e 1MB di dimensione).

Linux Windows AIX Configurazione TLS di canali di telemetria e client MQTT Java

Configurare TLS per autenticare il canale di telemetria e il client MQTT Java e codificare il trasferimento dei messaggi tra di essi. MQTT I client Java utilizzano JSSE (Java Secure Socket Extension) per connettere canali di telemetria utilizzando TLS. Come alternativa all'utilizzo di SSL, alcuni tipi di VPN (Virtual Private Network), ad esempio IPsec, autenticano gli endpoint di una connessione TCP/IP. VPN codifica ogni pacchetto IP che passa attraverso la rete. Una volta che tale connessione VPN è stabilita, è stata creata una rete attendibile. È possibile connettere i client MQTT ai canali di telemetria utilizzando TCP/IP sulla rete VPN.

È possibile configurare la connessione tra un client Java MQTT e un canale di telemetria per utilizzare il protocollo TLS su TCP/IP. Ciò che è protetto dipende dal modo in cui si configura TLS per utilizzare JSSE. A partire dalla configurazione più sicura, è possibile configurare tre diversi livelli di sicurezza:

1. Consentire la connessione solo ai client MQTT attendibili. Connetti un client MQTT solo a un canale di telemetria attendibile. Crittografare i messaggi tra il client e il gestore code; consultare “Autenticazione client MQTT tramite TLS” a pagina 348
2. Connetti un client MQTT solo a un canale di telemetria attendibile. Crittografare i messaggi tra il client e il gestore code; consultare “Autenticazione del canale di telemetria mediante TLS” a pagina 350.
3. Crittografare i messaggi tra il client e il gestore code; consultare “Privacy di pubblicazione sui canali di telemetria” a pagina 354.

Parametri di configurazione JSSE

Modificare i parametri JSSE per modificare il modo in cui è configurata una connessione TLS. I parametri di configurazione JSSE sono disposti in tre serie:

1. MQ Telemetry canale
2. Client MQTT Java
3. JRE

Configurare i parametri del canale di telemetria utilizzando IBM MQ Explorer. Impostare i parametri MQTT Java Client nell'attributo `MqttConnectionOptions.SSLProperties` . Modificare i parametri di sicurezza JRE modificando i file nella directory di sicurezza JRE sul client e sul server.

MQ Telemetry canale

Impostare tutti i parametri TLS del canale di telemetria utilizzando IBM MQ Explorer.

`ChannelName`

`ChannelName` è un parametro obbligatorio su tutti i canali.

Il nome del canale identifica il canale associato ad un numero di porta particolare. Denominare i canali per gestire le serie di client MQTT .

`PortNumber`

`PortNumber` è un parametro facoltativo su tutti i canali. Il valore predefinito è 1883 per i canali TCP e 8883 per i canali TLS.

Il numero di porta TCP/IP associato a questo canale. I client MQTT sono connessi ad un canale specificando la porta definita per il canale. Se il canale ha proprietà TLS, il client deve connettersi utilizzando il protocollo TLS; ad esempio:

```
MQTTClient mqttClient = new MqttClient( "ssl://www.example.org:8884", "clientId1");
mqttClient.connect();
```

`NomeKeyFile`

`KeyFileName` è un parametro obbligatorio per i canali TLS. Deve essere omesso per i canali TCP.

`KeyFileNome` è il percorso del keystore Java contenente certificati digitali forniti. Utilizzare JKS, JCEKS o PKCS12 come tipo di keystore sul server.

Identificare il tipo di keystore utilizzando una delle seguenti estensioni file:

```
.jks
.jceks
.p12
.pkcs12
```

Si presume che un keystore con qualsiasi altra estensione file sia un keystore JKS.

È possibile combinare un tipo di keystore sul server con altri tipi di keystore sul client.

Inserire il certificato privato del server nel keystore. Il certificato è noto come certificato server. Il certificato può essere autofirmato o parte di una concatenazione di certificati firmata da un'autorità di firma.

Se si sta utilizzando una catena di certificati, inserire i certificati associati nel keystore del server.

Il certificato del server e tutti i certificati nella sua catena di certificati vengono inviati ai client per autenticare l'identità del server.

Se `ClientAuth` è stato impostato su `Required`, il keystore deve contenere i certificati necessari per autenticare il client. Il client invia un certificato autofirmato, o una catena di certificati, e il client viene autenticato dalla prima verifica di questo materiale rispetto a un certificato nel keystore. Utilizzando una catena di certificati, un certificato può verificare molti client, anche se vengono emessi con certificati client differenti.

`PassPhrase`

`PassPhrase` è un parametro obbligatorio per i canali TLS. Deve essere omesso per i canali TCP.

La passphrase viene utilizzata per proteggere il keystore.

ClientAuth

`ClientAuth` è un parametro TLS facoltativo. L'impostazione predefinita è nessuna autenticazione client. Deve essere omesso per i canali TCP.

Impostare `ClientAuth` se si desidera che il servizio di telemetria (MQXR) autentichi il client, prima di consentire al client di connettersi al canale di telemetria.

Se si imposta `ClientAuth`, il client deve connettersi al server utilizzando TLS e autenticare il server. In risposta all'impostazione di `ClientAuth`, il client invia il proprio certificato digitale al server e qualsiasi altro certificato nel relativo keystore. Il suo certificato digitale è noto come certificato client. Questi certificati vengono autenticati rispetto a quelli contenuti nel keystore del canale e nell'archivio JRE `cacerts` .

CipherSuite

`CipherSuite` è un parametro TLS facoltativo. Viene utilizzato il valore predefinito per provare tutti i CipherSpecsabilitati. Deve essere omesso per i canali TCP.

Se si desidera utilizzare una particolare CipherSpec, impostare `CipherSuite` sul nome della CipherSpec che deve essere utilizzata per stabilire la connessione TLS.

Il servizio di telemetria e il client MQTT negoziano un CipherSpec comune da tutti i CipherSpecs abilitati ad ogni estremità. Se una specifica CipherSpec viene specificata in una o in entrambe le estremità della connessione, deve corrispondere alla CipherSpec nell'altra estremità.

Installare ulteriori cifrature aggiungendo ulteriori fornitori a JSSE.

FIPS (Federal Information Processing Standards)

FIPS è un'impostazione facoltativa. Per impostazione predefinita non è impostato.

Nel pannello delle proprietà del gestore code o utilizzando **runmqsc**, impostare SSLFIPS. SSLFIPS specifica se devono essere utilizzati solo algoritmi certificati FIPS.

Elenco nomi revoche

L'elenco nomi di revoca è un'impostazione facoltativa. Per impostazione predefinita non è impostato.

Nel pannello delle proprietà del gestore code o utilizzando **runmqsc**, impostare SSLCRLNL. SSLCRLNL specifica un elenco nomi degli oggetti delle informazioni di autenticazione utilizzati per fornire le posizioni di revoca dei certificati.

Non viene utilizzato nessun altro parametro del gestore code che imposta le proprietà TLS.

Client MQTT Java

Impostare le proprietà TLS per il client Java in `MqttConnectionOptions.SSLProperties` ; ad esempio:

```
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);
```

I nomi e i valori di specifiche proprietà sono descritti nella classe `MqttConnectOptions` . Per i collegamenti alla documentazione API client per le librerie client MQTT, consultare MQTT client programming reference.

Protocollo

`Protocollo` è facoltativo.

Il protocollo viene selezionato in negoziazione con il server di telemetria. Se si richiede un protocollo specifico, è possibile selezionarne uno. Se il server di telemetria non supporta il protocollo, la connessione ha esito negativo.

ContextProvider

`ContextProvider` è facoltativo.

KeyStore

`KeyStore` è facoltativo. Configurarlo se `ClientAuth` è impostato sul server per forzare l'autenticazione del client.

Inserire il certificato digitale del client, firmato utilizzando la chiave privata, nel keystore. Specificare il percorso e la password del keystore. Il tipo e il fornitore sono facoltativi. JKS è il tipo predefinito e IBMJCE è il fornitore predefinito.

Specificare un provider keystore differente per fare riferimento a una classe che aggiunge un nuovo provider keystore. Inoltrare il nome dell'algoritmo utilizzato dal provider keystore per istanziare il `KeyManagerFactory` impostando il nome del gestore chiavi.

TrustStore

`TrustStore` è facoltativo. È possibile inserire tutti i certificati attendibili nell'archivio JRE `cacerts` .

Configurare il truststore se si desidera avere un truststore diverso per il client. Potrebbe non essere necessario configurare il truststore se il server utilizza un certificato emesso da una CA nota che ha già il proprio certificato root memorizzato in `cacerts`.

Aggiungere il certificato firmato pubblicamente del server o il certificato root al truststore e specificare il percorso del truststore e la password. JKS è il tipo predefinito e IBMJCE è il fornitore predefinito.

Specificare un provider truststore differente per fare riferimento a una classe che aggiunge un nuovo provider truststore. Inoltrare il nome dell'algoritmo utilizzato dal provider truststore per creare un'istanza di `TrustManagerFactory` impostando il nome del gestore sicuro.

JRE

Altri aspetti della sicurezza Java che influiscono sul comportamento di TLS sia sul server che sul client sono configurati nel JRE. I file di configurazione su Windows si trovano in *Java Installation Directory*`\jre\lib\security`. Se si sta utilizzando il JRE fornito con IBM MQ , il percorso è come mostrato nella seguente tabella:

Tabella 18. Percorsi file per piattaforma per file di configurazione TLS JRE

Piattaforma	**Percorso file**
Windows	*WMQ Installation Directory*`\java\jre\lib\security`
AIX and Linux piattaforme	*WMQ Installation Directory*`/java/jre64/jre/lib/security`

Autorità di certificazione note

Il file `cacerts` contiene i certificati root delle autorità di certificazione note. `cacerts` viene utilizzato per impostazione predefinita, a meno che non si specifichi un truststore. Se si utilizza l'archivio `cacerts` o non si fornisce un truststore, è necessario esaminare e modificare l'elenco di firmatari in `cacerts` per soddisfare i propri requisiti di sicurezza.

V 9.4.0 V 9.4.0 È possibile utilizzare il comando **runmqktool** per gestire il file dei certificati `cacerts` . `cacerts` è un file JKS. Specificare il parametro **-storetype jks** quando il comando **runmqktool** viene utilizzato per gestire il file dei certificati.

V 9.4.0 V 9.4.0 La password predefinita per il file `cacerts` è `changeit`. Modificare la password utilizzando il comando **runmqktool -storepasswd** per proteggere il file.

Configurazione delle classi di sicurezza

Utilizzare il file `java.security` per registrare ulteriori provider di sicurezza e altre proprietà di sicurezza predefinite.

Autorizzazioni

Utilizzare il file `java.policy` per modificare le autorizzazioni concesse alle risorse. `javaws.policy` concede le autorizzazioni a `javaws.jar`

Livello di crittografia

Alcuni JRE vengono forniti con una crittografia di potenza ridotta. Se non è possibile importare le chiavi nei keystore, la causa potrebbe essere una codifica di livello ridotto. Se necessario, scaricare i file con giurisdizione limitata da IBM developer kits, Security information.

Importante: Il tuo paese di origine potrebbe avere restrizioni sull'importazione, il possesso, l'uso o la riesportazione in un altro paese, del software di crittografia. Prima di scaricare o utilizzare i file delle politiche illimitate, è necessario controllare le leggi del proprio Paese. Controllare le relative normative e le relative politiche relative all'importazione, al possesso, all'utilizzo e alla riesportazione del software di crittografia, per determinare se è consentito.

Modificare il provider di attendibilità per consentire al client di connettersi a qualsiasi server

L'esempio illustra come aggiungere un provider sicuro e farvi riferimento dal codice client MQTT . L'esempio non esegue alcuna autenticazione del client o del server. La connessione TLS risultante è codificata senza essere autenticata.

Il frammento di codice in Figura 16 a pagina 359 imposta il provider e il gestore sicuro `AcceptAllProviders` per il client 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. Frammento di codice MQTT Client

```
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

Linux Windows AIX Configurazione del canale di telemetria JAAS

Configurare JAAS per autenticare il Nome `utente` inviato dal client.

L'amministratore IBM MQ configura quali canali MQTT richiedono l'autenticazione client utilizzando JAAS. Specificare il nome di una configurazione JAAS per ogni canale che deve eseguire l'autenticazione JAAS . I canali possono utilizzare la stessa configurazione JAAS oppure possono utilizzare configurazioni JAAS differenti. Le configurazioni sono definite in *WMQData directory*`\qmgrs\`*qMgrName*`\mqxr\jaas.config`.

Il file `jaas.config` è organizzato per nome di configurazione JAAS . Sotto ogni nome di configurazione c'è un elenco di configurazioni di login; consultare “File jaas.config di esempio” a pagina 361.

JAAS fornisce quattro moduli di login standard. I moduli NT standard e UNIX Login hanno un valore limitato.

Modulo JndiLogin
Esegue l'autenticazione rispetto a un servizio directory configurato in JNDI (Java Naming and Directory Interface).

Krb5LoginModule
Autentica utilizzando i protocolli Kerberos .

NTLoginModule
Autentica utilizzando le informazioni di sicurezza NT per l'utente corrente.

Modulo UnixLogin
Autentica utilizzando le informazioni di sicurezza UNIX per l'utente corrente.

Il problema con l'utilizzo di `NTLoginModule` o `UnixLoginModule` è che il servizio di telemetria (MQXR) viene eseguito con l'identità `mqm` e non con l'identità del canale MQTT . `mqm` è l'identità passata a `NTLoginModule` o `UnixLoginModule` per l'autenticazione e non l'identità del client.

Per risolvere questo problema, scrivere il proprio modulo di login o utilizzare gli altri moduli di login standard. Un esempio di `JAASLoginModule.java` viene fornito con MQ Telemetry. È un'implementazione dell'interfaccia `javax.security.auth.spi.LoginModule` . Utilizzarla per sviluppare il proprio metodo di autenticazione.

Tutte le nuove classi LoginModule fornite devono trovarsi nel percorso classi del servizio di telemetria (MQXR). Non inserire le classi nelle directory IBM MQ che si trovano nel percorso di classe. Creare le proprie directory e definire l'intero percorso classe per il servizio di telemetria (MQXR).

È possibile aumentare il percorso classe utilizzato dal servizio di telemetria (MQXR) impostando il percorso classe nel file `service.env` . CLASSPATH deve essere in maiuscolo e l'istruzione del percorso classe può contenere solo valori letterali. Non è possibile utilizzare variabili in CLASSPATH; ad esempio, CLASSPATH=%CLASSPATH% non è corretto. Il servizio di telemetria (MQXR) imposta il proprio percorso classi. Il CLASSPATH definito in `service.env` viene aggiunto ad esso.

Il servizio di telemetria (MQXR) fornisce due callback che restituiscono il `Nome utente` e la `Password` per un client connesso al canale MQTT . `Nome utente` e `Password` sono impostati nell'oggetto `MqttConnectOptions` . Consultare “Metodo JAASLoginModule.Login() di esempio” a pagina 361 per un esempio di come accedere a `Nome utente` e `Password`.

File `jaas.config` di esempio

Un esempio di file di configurazione JAAS con una configurazione denominata, `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";
};
```

Metodo `JAASLoginModule.Login()` di esempio

Un esempio di modulo di accesso JAAS codificato per ricevere il `Nome utente` e la `Password` forniti da un client 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;
}
```

Attività correlate

Risoluzione del problema: modulo di login di JAAS non richiamato dal servizio di telemetria

Riferimenti correlati

Classe MQXR AuthCallback

Gestione di un client AMQP

È possibile amministrare un client AMQP utilizzando IBM MQ Explorer o su una riga comandi. Utilizzare Explorer per configurare canali e monitorare client AMQP connessi a IBM MQ. Configurare la sicurezza dei client AMQP utilizzando TLS e JAAS.

Prima di iniziare

Per informazioni sull'installazione di AMQP sulla tua piattaforma, vedi Scelta di cosa installare.

Amministrazione mediante IBM MQ Explorer

Utilizzare Explorer per configurare i canali AMQP e monitorare i client AMQP connessi a IBM MQ. È possibile configurare la sicurezza dei client AMQP utilizzando TLS e JAAS.

Gestione mediante la riga comandi

È possibile gestire un client AMQP dalla riga comandi utilizzando i comandi MQSC.

V 9.4.0 Linux Windows V 9.4.0 AIX Il servizio AMQP non si avvia automaticamente all'avvio del gestore code

Da IBM MQ 9.4.0, il funzionamento predefinito dell'impostazione dell'attributo **CONTROL** per l'avvio del servizio AMQP è cambiato. Quando si crea e si avvia un nuovo gestore code, il servizio AMQP non si avvia automaticamente come parte del processo di avvio del gestore code.

Tra IBM MQ 9.0.4 e IBM MQ 9.4.0, il comportamento predefinito dell'impostazione dell'attributo **CONTROL** per l'avvio del servizio AMQP è QMGR.

Se il componente AMQP è stato installato, il servizio AMQP è stato avviato automaticamente, anche se non utilizzato. Per evitare l'avvio predefinito della JVM (AMQP Java Virtual Machine), erano disponibili due opzioni:

- Non si sta installando il componente AMQP oppure
- Modifica dell'attributo **CONTROL** del servizio AMQP in MANUAL dopo l'avvio del gestore code.

Da IBM MQ 9.4.0, i gestori code appena creati hanno ripristinato l'impostazione dell'attributo **CONTROL** di SYSTEM.AMQP.SERVICE a MANUAL, che era l'impostazione predefinita prima di IBM MQ 9.0.4.

I gestori code migrati, se utilizzano AMQP, continuano ad avviare automaticamente il servizio durante l'avvio del gestore code. Per determinare se AMQP è stato utilizzato, vengono controllati:

- Canali AMQP esistenti
- Il canale ha avviato i messaggi nel log degli errori AMQP.

Attenzione:

- Ciò si verifica solo una volta; la prima volta che il gestore code viene avviato dopo un aggiornamento.
- Durante la migrazione, se l'attributo **CONTROL** viene modificato da QMGR a MANUAL, viene registrato un messaggio informativo nella registrazione errori IBM MQ per indicare la modifica. Per ulteriori informazioni, consultare Ubicazione dei log AMQP, dei log degli errori e dei file di configurazione .

Se si desidera avviare automaticamente il servizio AMQP, modificare l'attributo **CONTROL** del servizio in QMGR e riavviare il gestore code. I successivi riavvii del gestore code avviano il servizio AMQP.

Visualizzazione degli oggetti IBM MQ utilizzati dai clienti AMQP

È possibile visualizzare le diverse risorse IBM MQ utilizzate dai client AMQP, ad esempio le connessioni e le sottoscrizioni.

Connessioni

Quando il servizio AMQP viene avviato, vengono create nuove Hconn e connesse al gestore code. Questo pool di Hconns viene utilizzato quando i client AMQP pubblicano i messaggi. È possibile visualizzare le Hconn utilizzando il comando **DISPLAY CONN** . Ad esempio:

```
DISPLAY CONN(*) TYPE(CONN) WHERE (APPLDESC LK 'IBM MQ Advanced Message Queuing Protocol*')
```

Questo comando mostra anche tutte le Hconn specifiche del client. Le Hconn che dispongono di un attributo ID client vuoto sono le Hconn utilizzate nel pool

Quando un client AMQP si connette a un canale AMQP, al gestore code viene connesso un nuovo Hconn. Questo Hconn viene utilizzato per utilizzare i messaggi in maniera asincrona per le sottoscrizioni create dal client AMQP. È possibile visualizzare l'Hconn utilizzato da un particolare client AMQP utilizzando il comando **DISPLAY CONN** . Ad esempio:

```
DISPLAY CONN(*) TYPE(CONN) WHERE (CLIENTID EQ 'recv_abcd1234')
```

Sottoscrizioni create dai client

Quando un client AMQP sottoscrive un argomento, viene creata una nuova sottoscrizione IBM MQ . Il nome della sottoscrizione contiene le seguenti informazioni:

- Il nome del cliente. Se il client si è unito a una sottoscrizione condivisa, viene utilizzato il nome della condivisione
- Il modello di argomento a cui il client ha effettuato la sottoscrizione
- Un prefisso. Il prefisso è `private` se il client ha creato una sottoscrizione non condivisa oppure `share` se il client si è unito a una sottoscrizione condivisa

Per visualizzare le sottoscrizioni utilizzate da un particolare client AMQP, eseguire il comando **DISPLAY SUB** e filtrare il prefisso `private` :

```
DISPLAY SUB(':private:*')
```

Per visualizzare le sottoscrizioni condivise utilizzate da più client, eseguire il comando **DISPLAY SUB** e filtrare in base al prefisso `share` :

```
DISPLAY SUB(':share:*')
```

Poiché le sottoscrizioni condivise possono essere utilizzate da più client AMQP, è possibile visualizzare i client che attualmente utilizzano i messaggi dalla sottoscrizione condivisa. Puoi farlo elencando le Hconn che attualmente hanno un handle aperto sulla coda di sottoscrizione. Per visualizzare i client che attualmente utilizzano una condivisione, completare la seguente procedura:

1. Trovare il nome della coda che la sottoscrizione condivisa utilizza come destinazione. Ad esempio:

   ```
   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. Eseguire il comando **DISPLAY CONN** per trovare gli handle aperti su tale coda:

   ```
   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. Per ogni handle, visualizzare l'ID del client AMQP con l'handle aperto:

```
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)
```

Identificazione, autorizzazione e autenticazione del client AMQP

Come altre applicazioni client IBM MQ , è possibile proteggere le connessioni AMQP in diversi modi.

È possibile utilizzare le seguenti funzioni di protezione per proteggere le connessioni AMQP a IBM MQ:

- Record di autenticazione di canale
- Autenticazione connessione
- Configurazione utente MCA canale
- Definizioni di autorizzazione IBM MQ
- Connettività TLS

Da una prospettiva di sicurezza, stabilire una connessione consiste nelle seguenti due fasi:

- Decidere se la connessione deve continuare
- Determinazione dell'identità IBM MQ che l'applicazione assume per i successivi controlli di autorizzazione

Le seguenti informazioni illustrano le diverse configurazioni IBM MQ e i passi che vengono eseguiti quando un client AMQP tenta di stabilire una connessione. Non tutte le configurazioni IBM MQ utilizzano tutte le operazioni descritte. Ad esempio, alcune configurazioni non utilizzano TLS per le connessioni all'interno del firewall aziendale e alcune configurazioni utilizzano TLS ma non utilizzano i certificati client per l'autenticazione. Molti ambienti non utilizzano moduli JAAS personalizzati o personalizzati.

Stabilire una connessione

La seguente procedura descrive cosa accade quando una connessione viene stabilita da un client AMQP. La procedura determina se la connessione continua e quale identità IBM MQ assume l'applicazione per i controlli di autorizzazione:

1. Se il client apre una connessione TLS a IBM MQ e fornisce un certificato, il gestore code tenta di convalidarlo.
2. Se il client fornisce credenziali nome utente e password, il gestore code riceve un frame SASL AMQP e viene controllata la configurazione CONNAUTH di MQ .
3. Le regole di autenticazione del canale MQ vengono controllate (ad esempio, se l'indirizzo IP e il DN del certificato TLS sono validi)

4. Il canale MCAUSER viene asserito, a meno che le regole di autenticazione del canale non determinino diversamente.
5. Se è stato configurato un modulo JAAS , viene richiamato
6. MQ Controllo autorizzazione CONNECT applicato all'ID utente MQ risultante.
7. Connessione stabilita con un'identità IBM MQ assunta.

Pubblicazione di un messaggio

La seguente procedura descrive cosa accade quando un messaggio viene pubblicato da un client AMQP. La procedura determina se la connessione continua e quale identità IBM MQ assume l'applicazione per i controlli di autorizzazione:

1. Il frame di collegamento AMQP arriva al gestore code. L'autorizzazione di pubblicazione IBM MQ per la stringa di argomenti specificata viene controllata per l'identità utente MQ stabilita durante la connessione.
2. Il messaggio viene pubblicato nella stringa argomento specificata.

Sottoscrizione a un pattern di argomento

La seguente procedura descrive cosa accade quando un client AMQP sottoscrive un pattern di argomento. La procedura determina se la connessione continua e quale identità IBM MQ assume l'applicazione per i controlli di autorizzazione:

1. Il frame di collegamento AMQP arriva al gestore code. L'autorizzazione di sottoscrizione IBM MQ per il modello di argomento specificato viene controllata per l'identit ... utente MQ stabilita durante la connessione.
2. La sottoscrizione è stata creata.

Identità e autorizzazione del client AMQP

Utilizzare l'ID client AMQP, il nome utente AMQP o un'identità client comune definita sul canale o in una regola di autenticazione del canale, per l'autorizzazione ad accedere agli oggetti IBM MQ .

L'amministratore effettua la scelta quando definisce o modifica il canale AMQP, configurando l'impostazione CONNAUTH del gestore code o definendo le regole di autenticazione del canale. L'identità viene utilizzata per autorizzare l'accesso agli argomenti IBM MQ . La scelta viene effettuata in base a quanto segue:

1. L'attributo USECLNTID del canale.
2. L'attributo ADOPTCTX della regola CONNAUTH del gestore code.
3. L'attributo MCAUSER definito nel canale.
4. L'attributo USERSRC di una regola di autenticazione di canale corrispondente.

Prevenzione dei problemi: L'identità scelta da questo processo viene successivamente indicata, ad esempio dal comando AMQP (DISPLAY CHSTATUS), come MCAUSER del client. Tenere presente che questa non è necessariamente la stessa identità del MCAUSER del canale a cui si fa riferimento nella scelta (2).

Utilizzare il comando IBM MQ **setmqaut** per selezionare quali oggetti e quali azioni sono autorizzati ad essere utilizzati dall'identit ... associata al canale AMQP. Ad esempio, i seguenti comandi autorizzano un'identità del canale `AMQPClient`, fornita dall'amministratore del gestore code `QM1`:

```
setmqaut -m QM1 -t topic -n SYSTEM.BASE.TOPIC -p AMQPClient -all +pub +sub
```

e

```
setmqaut -m QM1 -t qmgr -p AMQPClient -all +connect
```

Autenticazione client AMQP utilizzando una password

Autenticare il nome utente del client AMQP utilizzando la password del client. È possibile autenticare il client utilizzando un'identità diversa da quella utilizzata per autorizzare il client a pubblicare e sottoscrivere argomenti.

Il servizio AMQP può utilizzare MQ CONNAUTH o JAAS per autenticare il nome utente del client. Se uno di questi è configurato, la password fornita dal client viene verificata dalla configurazione CONNAUTH di MQ o dal modulo JAAS .

La seguente procedura descrive i passi di esempio per autenticare i singoli utenti rispetto agli utenti e alle password del sistema operativo locale e, in caso di esito positivo, adottare l'identità comune `AMQPUser`:

1. L'amministratore di IBM MQ imposta l'identità MCAUSER del canale AMQP su qualsiasi nome, ad esempio `AMQPUser`, utilizzando Esplora risorse di IBM MQ .
2. L'amministratore IBM MQ autorizza `AMQPUser` a pubblicare e sottoscrivere qualsiasi argomento:

   ```
   setmqaut -m QM1 -t topic -n SYSTEM.BASE.TOPIC -p AMQPUser -all +pub +sub +connect
   ```

3. L'amministratore IBM MQ configura una regola IDPWOS CONNAUTH per controllare il nome utente e password presentati dal client. La regola CONNAUTH deve impostare CHCKCLNT (REQUIRED) e ADOPTCTX (NO).

Nota: Si consiglia di utilizzare le regole di autenticazione di canale e di impostare l'attributo del canale MCAUSER su un utente che non dispone di privilegi, per consentire un maggiore controllo sulle connessioni al gestore code.

Privacy delle pubblicazioni sui canali

La privacy delle pubblicazioni AMQP inviate in entrambe le direzioni attraverso i canali AMQP è protetta utilizzando TLS per codificare le trasmissioni sulla connessione.

I client AMQP che si collegano ai canali AMQP utilizzano TLS per proteggere la privacy delle pubblicazioni trasmesse sul canale utilizzando la crittografia a chiave simmetrica. Poiché gli endpoint non sono autenticati, non puoi considerare attendibile la sola crittografia del canale. Combina la protezione della privacy con il server o l'autenticazione reciproca.

Come alternativa all'utilizzo di TLS, alcuni tipi di VPN (Virtual Private Network), come IPsec, autenticano gli endpoint di una connessione TCP/IP. VPN codifica ogni pacchetto IP che passa attraverso la rete. Una volta che tale connessione VPN è stabilita, è stata creata una rete attendibile. È possibile connettere i client AMQP ai canali AMQP utilizzando TCP/IP sulla rete VPN.

La crittografia delle connessioni TLS senza autenticare il server espone la connessione agli attacchi man - in - the - middle. Anche se le informazioni scambiate sono protette contro le intercettazioni, non sai con chi le stai scambiando. A meno che tu non controlli la rete, sei esposto a qualcuno che intercetta le tue trasmissioni IP e si maschera come l'endpoint.

Puoi creare una connessione TLS crittografata, senza autenticare il server, utilizzando uno scambio di chiavi Diffie - Hellman CipherSpec che supporti TLS anonimo. Il segreto master, condiviso tra il client e il server e utilizzato per codificare le trasmissioni TLS, viene stabilito senza scambiare un certificato server firmato privatamente.

Poiché le connessioni anonime non sono sicure, la maggior parte delle implementazioni TLS non utilizzano per impostazione predefinita CipherSpecsanonime. Se una richiesta client per la connessione TLS viene accettata da un canale AMQP, il canale deve avere un keystore protetto da una passphrase. Per impostazione predefinita, poiché le implementazioni TLS non utilizzano CipherSpecsanonime, il keystore deve contenere un certificato firmato privatamente che il client può autenticare.

Se si utilizza CipherSpecsanonimo, il keystore del server deve esistere, ma non deve contenere alcun certificato firmato privatamente.

Un altro modo per stabilire una connessione crittografata consiste nel sostituire il provider di attendibilità sul client con la propria implementazione. Il provider di attendibilità non autentica il certificato del server, ma la connessione viene codificata.

Configurazione dei client AMQP con TLS

È possibile configurare i client AMQP per utilizzare TLS per proteggere i dati in transito nella rete e per autenticare l'identità del gestore code a cui si connette il client.

Per utilizzare TLS per la connessione da un client AMQP a un canale AMQP, è necessario assicurarsi che il gestore code sia stato configurato per TLS. Configurazione di TLS sui gestori code descrive come configurare il keystore da cui un gestore code legge i certificati TLS.

Quando il gestore code è stato configurato con un keystore, è necessario configurare gli attributi TLS sul canale AMQP a cui si connetteranno i client. I canali AMQP hanno quattro attributi correlati alla configurazione TLS come segue:

SSLCAUTH
L'attributo SSLCAUTH viene utilizzato per specificare se il gestore code deve richiedere un client AMQP per presentare un certificato client per verificarne l'identità.

SSLCIPH
L'attributo SSLCIPH specifica la codifica che il canale deve utilizzare per codificare i dati nel flusso TLS.

V 9.4.0 Da IBM MQ 9.4.0, i canali AMQP supportano CipherSpecsgenerici ANY*. Per ulteriori informazioni su CipherSpecs, consultare Abilitazione di CipherSpecs.

SSLPEER
L'attributo SSLPEER viene utilizzato per specificare il DN (distinguished name) a cui deve corrispondere un certificato client se deve essere consentita una connessione.

CERTLABL
CERTLABL specifica il certificato che il gestore code deve presentare al client. Il keystore del gestore code può contenere più certificati. Questo attributo consente di specificare il certificato da utilizzare per le connessioni a questo canale. Se non è specificato alcun CERTLABL, viene utilizzato il certificato nel repository delle chiavi del gestore code con l'etichetta corrispondente all'attributo CERTLABL del gestore code.

Una volta configurato il tuo canale AMQP con gli attributi TLS, devi riavviare il servizio AMQP utilizzando il seguente comando:

```
STOP SERVICE(SYSTEM.AMQP.SERVICE) START SERVICE(SYSTEM.AMQP.SERVICE)
```

Quando un client AMQP si connette a un canale AMQP protetto da TLS, il client verifica l'identità del certificato presentato dal gestore code. Per fare ciò, è necessario configurare il proprio client AMQP con un truststore contenente il certificato del gestore code. I passi per eseguire questa operazione variano a seconda del client AMQP che si sta utilizzando. Per informazioni sui vari client e API AMQP, consultare la rispettiva documentazione del client AMQP.

Riferimenti correlati
DEFINE CHANNEL (definire un nuovo canale)
STOP SERVICE (arrestare un servizio) su Multiplatforms
START SERVICE (avvio di un servizio) su Multiplatforms

Disconnessione dei clienti AMQP dal gestore code

Se si desidera disconnettere i client AMQP dal gestore code, eseguire il comando PURGE CHANNEL o arrestare la connessione al client AMQP.

- Eseguire il comando **PURGE CHANNEL** . Ad esempio:

```
PURGE CHANNEL(MYAMQP) CLIENTID('recv_28dbb7e')
```

- In alternativa, arrestare la connessione utilizzata dal client AMQP per disconnettere il client completando la seguente procedura:

  1. Individuare la connessione utilizzata dal client eseguendo il comando **DISPLAY CONN** . Ad esempio:

```
DISPLAY CONN(*) TYPE(CONN) WHERE (CLIENTID EQ 'recv_28dbb7e')
```

     L'output del comando è il seguente:

```
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. Arrestare la connessione. Ad esempio:

```
STOP CONN(707E0A565F2D0020)
```

Gestione multicast

Utilizzare queste informazioni per informazioni sulle attività di gestione di IBM MQ Multicast, come la riduzione della dimensione dei messaggi multicast e l'abilitazione della conversione dei dati.

Introduzione a multicast

Utilizzare queste informazioni per iniziare a utilizzare gli argomenti IBM MQ Multicast e gli oggetti delle informazioni di comunicazione.

Informazioni su questa attività

IBM MQ La messaggistica multicast utilizza la rete per consegnare i messaggi associando gli argomenti agli indirizzi di gruppo. Le seguenti attività sono un modo rapido per verificare se la porta e l'indirizzo IP richiesti sono configurati correttamente per la messaggistica multicast.

Creazione di un oggetto COMMINFO per multicast

L'oggetto informazioni di comunicazione (COMMINFO) contiene gli attributi associati alla trasmissione multicast. Per ulteriori informazioni sui parametri oggetto COMMINFO, consultare DEFINE COMMINFO.

Utilizzare il seguente esempio di riga comandi per definire un oggetto COMMINFO per multicast:

```
DEFINE COMMINFO(MC1) GRPADDR(group address) PORT(port number)
```

dove *MC1* è il nome dell'oggetto COMMINFO, *indirizzo gruppo* è l'indirizzo IP o il nome DNS multicast del gruppo e *numero porta* è la porta su cui trasmettere (il valore predefinito è 1414).

Viene creato un nuovo oggetto COMMINFO denominato *MC1* ; questo nome è il nome che è necessario specificare quando si definisce un oggetto TOPIC nell'esempio successivo.

Creazione di un oggetto TOPIC per multicast

Un argomento è l'oggetto delle informazioni pubblicate in un messaggio di pubblicazione / sottoscrizione e un argomento viene definito creando un oggetto TOPIC. Gli oggetti TOPIC hanno

due parametri che definiscono se possono essere utilizzati o meno con multicast. Questi parametri sono **COMMINFO** e **MCAST**.

- **COMMINFO** Questo parametro specifica il nome dell'oggetto delle informazioni di comunicazione multicast. Per ulteriori informazioni sui parametri oggetto COMMINFO, consultare DEFINE COMMINFO.
- **MCAST** Questo parametro specifica se il multicast è consentito in questa posizione nella struttura ad albero degli argomenti.

Utilizzare il seguente esempio di riga comandi per definire un oggetto TOPIC per multicast:

```
DEFINE TOPIC(ALLSPORTS) TOPICSTR('Sports') COMMINFO(MC1) MCAST(ENABLED)
```

Viene creato un nuovo oggetto TOPIC denominato *ALLSPORTS* . Ha una stringa di argomenti *Sports*, il relativo oggetto delle informazioni di comunicazione è denominato *MC1* (che è il nome specificato quando si definisce un oggetto COMMINFO nell'esempio precedente) e multicast è abilitato.

Verifica della pubblicazione / sottoscrizione multicast

Una volta creati gli oggetti TOPIC e COMMINFO, è possibile verificarli utilizzando l'esempio `amqspubc` e `amqssubc` . Per ulteriori informazioni su questi esempi, consultare Programmi di esempio di pubblicazione / sottoscrizione.

1. Aprire due finestre della riga comandi; la prima riga comandi è per l'esempio di pubblicazione `amqspubc` e la seconda riga comandi è per l'esempio di sottoscrizione `amqssubc` .
2. Immettere il seguente comando nella riga comandi 1:

   ```
   amqspubc Sports QM1
   ```

   dove *Sports* è la stringa argomento dell'oggetto TOPIC definito in un esempio precedente e *QM1* è il nome del gestore code.
3. Immettere il seguente comando nella riga comandi 2:

   ```
   amqssubc Sports QM1
   ```

   dove *Sports* e *QM1* sono gli stessi utilizzati nel passaggio “2” a pagina 369.
4. Immettere `Hello world` alla riga comandi 1. Se la porta e l'indirizzo IP specificati nell'oggetto COMMINFO sono configurati correttamente; l'esempio `amqssubc` , che è in ascolto sulla porta per le pubblicazioni dall'indirizzo specificato, emette `Hello world` sulla riga comandi 2.

Topologia argomento IBM MQ Multicast

Utilizzare questo esempio per comprendere la topologia dell'argomento IBM MQ Multicast.

IBM MQ Il supporto multicast richiede che ogni struttura ad albero secondaria abbia il proprio gruppo multicast e flusso di dati all'interno della gerarchia totale.

Lo schema di reindirizzamento IP *rete classful* ha uno spazio degli indirizzi designato per l'indirizzo multicast. L'intervallo multicast completo dell'indirizzo IP è compreso tra `224.0.0.0` e `239.255.255.255` ma alcuni di questi indirizzi sono riservati. Per un elenco di indirizzi riservati, contattare l'amministratore del sistema oppure visualizzare https://www.iana.org/assignments/multicast-addresses per ulteriori informazioni. Si consiglia di utilizzare l'indirizzo multicast di ambito locale nell'intervallo compreso tra `239.0.0.0` e `239.255.255.255`.

Nel seguente diagramma, sono disponibili due possibili flussi di dati multicast:

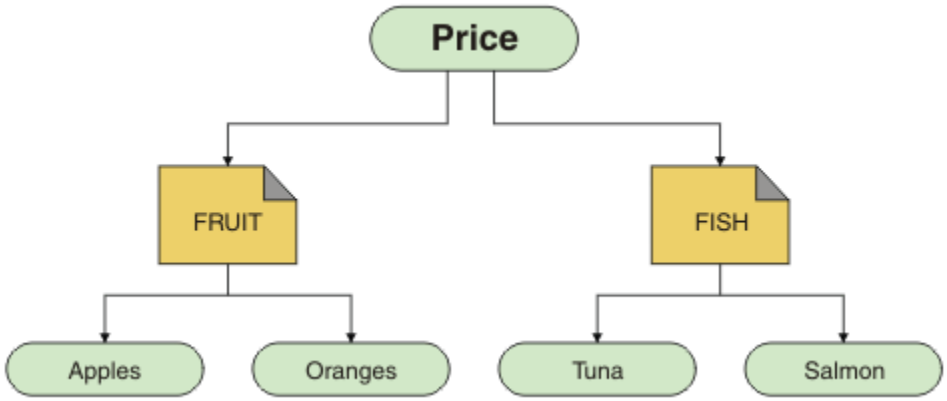
```
DEF COMMINFO(MC1) GRPADDR(239.XXX.XXX.XXX
)

DEF COMMINFO(MC2) GRPADDR(239.YYY.YYY.YYY)
```

dove *239.XXX.XXX.XXX* e *239.YYY.YYY.YYY* sono indirizzi multicast validi.

Queste definizioni degli argomenti vengono utilizzate per creare una struttura ad albero degli argomenti come mostrato nel seguente diagramma:

```
DEFINE TOPIC(FRUIT) TOPICSTRING('Price/FRUIT') MCAST(ENABLED) COMMINFO(MC1)
DEFINE TOPIC(FISH) TOPICSTRING('Price/FISH') MCAST(ENABLED) COMMINFO(MC2)
```



Ciascun oggetto informazioni di comunicazione multicast (COMMINFO) rappresenta un flusso di dati differente poiché i rispettivi indirizzi di gruppo sono diversi. In questo esempio, l'argomento FRUIT è definito per utilizzare l'oggetto COMMINFO MC1 , l'argomento FISH è definito per utilizzare l'oggetto COMMINFO MC2 e il nodo Price non ha definizioni multicast.

IBM MQ Il multicast ha un limite di 255 caratteri per le stringhe argomento. Questa limitazione significa che è necessario prestare attenzione ai nomi dei nodi e dei nodi foglia all'interno della struttura ad albero; se i nomi dei nodi e dei nodi foglia sono troppo lunghi, la stringa dell'argomento potrebbe superare 255 caratteri e restituire il codice motivo 2425 (0979) (RC2425): MQRC_TOPIC_STRING_ERROR . Si consiglia di rendere le stringhe di argomenti il più brevi possibile perché le stringhe di argomenti più lunghe potrebbero avere un effetto negativo sulle prestazioni.

Controllo della dimensione dei messaggi multicast

Utilizzare queste informazioni per informazioni sul formato del messaggio IBM MQ e ridurre la dimensione dei messaggi IBM MQ .

I messaggi IBM MQ hanno un numero di attributi associati che sono contenuti nel descrittore del messaggio. Per i messaggi di piccole dimensioni, questi attributi potrebbero rappresentare la maggior parte del traffico dati e possono avere un effetto negativo significativo sulla velocità di trasmissione. IBM MQ Multicast consente all'utente di configurare quali, se presenti, di questi attributi vengono trasmessi insieme al messaggio.

La presenza di attributi del messaggio, diversi dalla stringa dell'argomento, dipende dal fatto che l'oggetto COMMINFO indichi che devono essere inviati o meno. Se un attributo non viene trasmesso, l'applicazione di ricezione applica un valore predefinito. I valori MQMD predefiniti non corrispondono necessariamente al valore MQMD_DEFAULT e sono descritti in Tabella 19 a pagina 371.

L'oggetto COMMINFO contiene l'attributo MCPROP che controlla il numero di campi MQMD e il flusso di proprietà utente con il messaggio. Impostando il valore di questo attributo su un livello appropriato, è possibile controllare la dimensione dei messaggi multicast IBM MQ :

MCPROP
Il controllo proprietà multicast verifica quante proprietà utente e MQMD vengono trasmesse insieme al messaggio.

TUTTO
Vengono trasmesse tutte le proprietà utente e tutti i campi di MQMD.

Rispondi
Solo le proprietà utente e i campi MQMD che si occupano delle risposte ai messaggi vengono trasmessi. Queste proprietà sono:

- MsgType
- MessageId
- CorrelId
- ReplyToQ
- ReplyToQmgr

UTENTE
Solo le proprietà utente vengono trasmesse.

Nessuna
Non vengono trasmessi né le proprietà utente, né i campi MQMD.

COMPAT
Questo valore fa in modo che la trasmissione del messaggio venga eseguita in modalità compatibile a RMM, il che consente alcune interoperazioni con le applicazioni XMS correnti e IBM Integration Bus RMM .

XMS .NET La messaggistica multicast (utilizzando RMM) è stata obsoleta da IBM MQ 9.2 e rimossa alle IBM MQ 9.3.

Attributi messaggio multicast

Gli attributi dei messaggi possono provenire da diverse posizioni, ad esempio MQMD, i campi in MQRFH2e le proprietà dei messaggi.

La seguente tabella mostra cosa succede quando i messaggi vengono inviati in base al valore di MCPROP (descritto in precedenza in questa sezione) e il valore predefinito utilizzato quando un attributo non viene inviato.

Tabella 19. Attributi di messaggistica e modalità di relazione con multicast

Attributo	Azione quando si utilizza multicast	Valore predefinito se non trasmesso
TopicString	Sempre incluso	Non applicabile
StrucId MQMQ	Non trasmesso	Non applicabile
Versione MQMD	Non trasmesso	Non applicabile
Prospetto	Incluso se non predefinito	0
MsgType	Incluso se non predefinito	MQMT_DATAGRAM
Scadenza	Incluso se non predefinito	0
Feedback	Incluso se non predefinito	0
Codifica	Incluso se non predefinito	MQENC_NORMAL (equivalente)
CodedCharSetId	Incluso se non predefinito	1208
Formato	Incluso se non predefinito	MQRFH2
Priorità	Incluso se non predefinito	4
Persistenza	Incluso se non predefinito	MQPER_NOT_PERSISTENT
MsgId	Incluso se non predefinito	Nullo
CorrelId	Incluso se non predefinito	Nullo

Tabella 19. Attributi di messaggistica e modalità di relazione con multicast (Continua)		
Attributo	**Azione quando si utilizza multicast**	**Valore predefinito se non trasmesso**
BackoutCount	Incluso se non predefinito	0
ReplyToQ	Incluso se non predefinito	Spazio
ReplyToQMgr	Incluso se non predefinito	Spazio
UserIdentifier	Incluso se non predefinito	Spazio
AccountingToken	Incluso se non predefinito	Nullo
Tipo IT PutApp	Incluso se non predefinito	JAVA MQAT
PutAppIName	Incluso se non predefinito	Spazio
PutDate	Incluso se non predefinito	Spazio
PutTime	Incluso se non predefinito	Spazio
ApplOriginData	Incluso se non predefinito	Spazio
GroupID	Esclusi	Non applicabile
MsgSeqNumber	Esclusi	Non applicabile
Offset	Esclusi	Non applicabile
MsgFlags	Esclusi	Non applicabile
OriginalLength	Esclusi	Non applicabile
UserProperties	Inclusi	Non applicabile

Riferimenti correlati

Multi COMMINFO ALTER

DEFINE COMMINFO

Abilitazione della conversione dati per la messaggistica multicast

Utilizzare queste informazioni per comprendere il funzionamento della conversione dei dati per la messaggistica IBM MQ Multicast.

IBM MQ Multicast è un protocollo condiviso, senza connessione e quindi non è possibile per ogni client effettuare richieste specifiche per la conversione dei dati. Ogni client sottoscritto allo stesso flusso multicast riceve gli stessi dati binari; pertanto, se è richiesta la conversione dei dati IBM MQ , la conversione viene eseguita localmente su ciascun client.

In un'installazione con piattaforma mista, è possibile che la maggior parte dei client richieda i dati in un formato che non sia il formato nativo dell'applicazione di trasmissione. In questa situazione, i valori **CCSID** e **ENCODING** dell'oggetto COMMINFO multicast possono essere utilizzati per definire la codifica della trasmissione del messaggio per una maggiore efficienza.

IBM MQ Multicast supporta la conversione dei dati del payload del messaggio per i seguenti formati integrati:

- MQADMIN
- MQEVENT
- MQPCF
- MQRFH
- MQRFH2

- MQSTR

Oltre a questi formati, è anche possibile definire i propri formati e utilizzare un'uscita di conversione dati MQDXP - parametro di conversione dati .

Per informazioni sulla programmazione delle conversione dei dati, consultare Conversione dei dati in MQI per la messaggistica multicast.

Per ulteriori informazioni sulla conversione dei dati, consultare Conversione dei dati.

Per ulteriori informazioni sulle uscite di conversione dati e `ClientExitPath`, consultare la sezione ClientExitPercorso del file di configurazione client.

Monitoraggio applicazione multicast

Utilizzare queste informazioni per informazioni sulla gestione e il monitoraggio di IBM MQ Multicast.

Lo stato dei publisher e dei sottoscrittori correnti per il traffico multicast (ad esempio, il numero di messaggi inviati e ricevuti o il numero di messaggi persi) viene periodicamente trasmesso al server dal client. Quando si riceve lo stato, l'attributo COMMEV dell'oggetto COMMINFO specifica se il gestore code inserisce o meno un messaggio di evento nel SISTEMA SYSTEM.ADMIN.PUBSUB.EVENT. Il messaggio dell'evento contiene le informazioni sullo stato ricevute. Queste informazioni sono un aiuto diagnostico prezioso per individuare l'origine di un problema.

Utilizzare il comando MQSC **DISPLAY CONN** per visualizzare le informazioni di connessione relative alle applicazioni connesse al gestore code. Per ulteriori informazioni sul comando **DISPLAY CONN** , consultare DISPLAY CONN.

Utilizzare il comando MQSC **DISPLAY TPSTATUS** per visualizzare lo stato dei publisher e dei sottoscrittori. Per ulteriori informazioni sul comando **DISPLAY TPSTATUS** , consultare DISPLAY TPSTATUS.

COMMEV e l'indicatore di affidabilità del messaggio multicast

L' *indicatore di affidabilità*, utilizzato insieme all'attributo **COMMEV** dell'oggetto COMMINFO, è un elemento chiave nel monitoraggio dei publisher e dei sottoscrittori IBM MQ Multicast. L'indicatore di affidabilità (il campo **MSGREL** che viene restituito nei comandi di stato di pubblicazione o sottoscrizione) è un indicatore IBM MQ che illustra la percentuale di trasmissioni che non hanno errori. A volte i messaggi devono essere ritrasmessi a causa di un errore di trasmissione, che si riflette nel valore di **MSGREL**. Le cause potenziali degli errori di trasmissione includono sottoscrittori lenti, reti occupate e interruzioni di rete. **COMMEV** controlla se i messaggi di eventi vengono generati per handle multicast creati utilizzando l'oggetto COMMINFO ed è impostato su uno dei tre valori possibili:

DISABILITATO
I messaggi di evento non vengono scritti.

Abilitato
I messaggi di evento vengono sempre scritti, con una frequenza definita nel parametro COMMINFO **MONINT** .

ECCEZIONE
I messaggi di evento vengono scritti se l'affidabilità del messaggio è al di sotto della soglia di affidabilità. Un livello di affidabilità del messaggio del 90% o inferiore indica che potrebbe esserci un problema con la configurazione di rete o che una o più applicazioni di pubblicazione / sottoscrizione sono in esecuzione troppo lentamente:

- Il valore **MSGREL(100,100)** indica che non si sono verificati problemi a breve termine o a lungo termine.
- Il valore **MSGREL(80,60)** indica che il 20% dei messaggi sta attualmente avendo problemi, ma che si tratta anche di un miglioramento rispetto al valore a lungo termine di 60.

I client potrebbero continuare a trasmettere e ricevere traffico multicast anche quando la connessione unicast al gestore code è interrotta, pertanto i dati potrebbero non essere aggiornati.

Affidabilità dei messaggi multicast

Utilizzare queste informazioni per informazioni su come impostare la sottoscrizione multicast IBM MQ e la cronologia dei messaggi.

Un elemento chiave per superare l'errore di trasmissione con multicast è il buffer dei dati trasmessi (una cronologia dei messaggi che devono essere conservati all'estremità di trasmissione del link) da IBM MQ. Questo processo significa che non è richiesta alcuna memorizzazione nel buffer dei messaggi nel processo di inserimento dell'applicazione poiché IBM MQ fornisce l'affidabilità. La dimensione di questa cronologia viene configurata mediante l'oggetto informazioni di comunicazione (COMMINFO), come descritto nelle seguenti informazioni. Un buffer di trasmissione più grande significa che c'è più cronologia di trasmissione da ritrasmettere se necessario, ma a causa della natura del multicast, la consegna garantita al 100% non può essere supportata.

La cronologia dei messaggi multicast IBM MQ è controllata nell'oggetto delle informazioni di comunicazione (COMMINFO) dall'attributo **MSGHIST** :

MSGHIST
Questo valore è la quantità di cronologia dei messaggi in kilobyte conservata dal sistema per gestire le ritrasmissioni nel caso di NACK (riconoscimenti negativi).

Il valore `0` fornisce il livello minimo di affidabilità. Il valore predefinito è `100` KB.

La cronologia della nuova sottoscrizione IBM MQ Multicast è controllata nell'oggetto informazioni di comunicazione (COMMINFO) dall'attributo **NSUBHIST** :

NSUBHIST
La cronologia nuovo sottoscrittore verifica se un sottoscrittore che si iscrive a un flusso di pubblicazioni riceve tutti dati attualmente disponibili o solo le pubblicazioni disponibili dal momento della sottoscrizione.

Nessuna
Un valore di `NONE` fa sì che il trasmettitore trasmetta solo la pubblicazione effettuata dal momento della sottoscrizione. `NONE` è il valore predefinito.

TUTTO
Un valore `ALL` fa sì che il trasmettitore ritrasmetta la quantità di cronologia dell'argomento nota. In alcune circostanze, questa situazione può fornire un comportamento simile alle pubblicazioni conservate.

Nota: L'utilizzo del valore di `ALL` potrebbe avere un effetto negativo sulle prestazioni se esiste una cronologia di argomenti di grandi dimensioni poiché tutta la cronologia degli argomenti viene ritrasmessa.

Riferimenti correlati
DEFINE COMMINFO
Multi COMMINFO ALTER

Attività multicast avanzate

Utilizzare queste informazioni per informazioni sulle attività di gestione avanzate di IBM MQ Multicast, come la configurazione dei file `.ini` e l'interoperabilità con IBM MQ LLM.

Per considerazioni sulla sicurezza in un'installazione multicast, consultare Sicurezza multicast.

Collegamento tra domini di pubblicazione / sottoscrizione multicast e non multicast

Utilizzare queste informazioni per comprendere cosa accade quando un publisher non multicast pubblica in un argomento abilitato per IBM MQ Multicast.

Se un publisher non multicast pubblica un argomento definito come **MCAST** abilitato e **BRIDGE** abilitato, il gestore code trasmette il messaggio tramite multicast direttamente a tutti i sottoscrittori che potrebbero

essere in ascolto. Un publisher multicast non può eseguire la pubblicazione su argomenti che non sono abilitati multicast.

Gli argomenti esistenti possono essere abilitati multicast impostando i parametri **MCAST** e **COMMINFO** di un oggetto argomento. Consultare Initial multicast concepts per ulteriori informazioni su questi parametri.

L'attributo oggetto COMMINFO **BRIDGE** controlla le pubblicazioni dalle applicazioni che non utilizzano multicast. Se **BRIDGE** è impostato su ENABLED e il parametro **MCAST** dell'argomento è impostato anche su ENABLED, le pubblicazioni delle applicazioni che non utilizzano il multicast vengono collegate tramite bridge alle applicazioni che lo utilizzano. Per ulteriori informazioni sul parametro **BRIDGE** , consultare DEFINE COMMINFO.

Configurazione dei file .ini per Multicast

Utilizzare queste informazioni per comprendere i campi IBM MQ Multicast nei file `.ini` .

È possibile eseguire un'ulteriore configurazione di IBM MQ Multicast in un file `ini` . Il file `ini` specifico che è necessario utilizzare dipende dal tipo di applicazioni:

- Client: configurare il file *MQ_DATA_PATH* `/mqclient.ini` .
- Gestore code: configurare il file *MQ_DATA_PATH* `/qmgrs/`*QMNAME*`/qm.ini` .

dove *MQ_DATA_PATH* è l'ubicazione della directory di dati IBM MQ (`/var/mqm/mqclient.ini`) e *QMNAME* è il nome del gestore code a cui si applica il file `.ini` .

Il file `.ini` contiene campi utilizzati per ottimizzare il comportamento di IBM MQ Multicast:

```
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
```

Protocollo

UDP
In questa modalità, i package vengono inviati utilizzando il protocollo UDP. Tuttavia, gli elementi di rete non possono fornire assistenza nella distribuzione multicast come avviene in modalità IP. Il formato del pacchetto rimane compatibile con PGM. Questo è il valore predefinito.

IP
In questa modalità, il trasmettitore invia pacchetti IP non elaborati. Gli elementi di rete con supporto PGM assistono nella distribuzione di pacchetti multicast affidabili. Questa modalità è completamente compatibile con lo standard PGM.

IPVERSION

IPv4
Comunicare utilizzando solo il protocollo IPv4 . Questo è il valore predefinito.

IPv6
Comunicare utilizzando solo il protocollo IPv6 .

ANY
Comunicare utilizzando IPv4, IPv6o entrambi, a seconda del protocollo disponibile.

ENTRAMBI
Supporta la comunicazione utilizzando IPv4 e IPv6.

Frequenza LimitTrans

DISABILITATO
Non esiste alcun controllo della velocità di trasmissione. Questo è il valore predefinito.

STATICO
Implementa il controllo della velocità di trasmissione statica. Il trasmettitore non trasmetterà ad una velocità superiore a quella specificata dal parametro TransRateLimit.

DINAMICO
Il trasmettitore adatta la sua velocità di trasmissione in base al feedback che riceve dai ricevitori. In questo caso, il limite della velocità di trasmissione non può essere superiore al valore specificato dal parametro TransRateLimit. Il trasmettitore cerca di raggiungere una velocità di trasmissione ottimale.

Limite TransRate
Il limite della velocità di trasmissione in Kbps.

SocketTTL
Il valore di SocketTTL determina se il traffico multicast può passare attraverso un router o il numero di router che può passare attraverso.

Batch
Controlla se i messaggi vengono batch o inviati immediatamente Ci sono 2 valori possibili:

- *NO* I messaggi non sono in batch, vengono inviati immediatamente.
- *Sì* I messaggi vengono sottoposti a batch.

Loop
Impostare il valore su 1 per abilitare il loop multicast. Il loop multicast definisce se i dati inviati vengono inviati in loop o meno all'host.

Interfaccia
L'indirizzo IP dell'interfaccia su cui fluisce il traffico multicast. Per ulteriori informazioni e la risoluzione dei problemi, consultare: Test delle applicazioni multicast su una rete non multicast e Impostazione della rete appropriata per il traffico multicast

FeedbackMode

NACK
Feedback da riconoscimenti negativi. Questo è il valore predefinito.

ACK
Feedback da riconoscimenti positivi.

WAIT1
Feedback da parte di riconoscimenti positivi in cui il trasmettitore attende solo 1 ACK da uno qualsiasi dei ricevitori.

HeartbeatTimeout
Il timeout heartbeat in millisecondi. Il valore 0 indica che gli eventi di timeout heartbeat non vengono generati dal destinatario o dai destinatari dell'argomento. Il valore predefinito è 20000.

HeartbeatInterval
L'intervallo di heartbeat in millisecondi. Il valore 0 indica che non viene inviato alcun heartbeat. L'intervallo di heartbeat deve essere notevolmente inferiore al valore **`HeartbeatTimeout`** per evitare falsi eventi di timeout di heartbeat. Il valore predefinito è 2000.

Interoperabilità multicast con IBM MQ Low Latency Messaging

Utilizzare queste informazioni per comprendere l'interoperabilità tra IBM MQ Multicast e IBM MQ LLM (Low Latency Messaging).

Il trasferimento payload di base è possibile per un'applicazione che utilizza LLM, con un'altra applicazione che utilizza multicast per scambiare messaggi in entrambe le direzioni. Anche se multicast utilizza la tecnologia LLM, il prodotto LLM stesso non è incorporato. Pertanto è possibile installare sia LLM che IBM MQ Multicast, e operare e servire i due prodotti separatamente.

Le applicazioni LLM che comunicano con multicast potrebbero dover inviare e ricevere le proprietà del messaggio. Le proprietà del messaggio IBM MQ e i campi MQMD vengono trasmessi come proprietà del messaggio LLM con codici proprietà del messaggio LLM specifici, come mostrato nella tabella seguente:

Tabella 20. Associazioni delle proprietà IBM MQ message to IBM MQ LLM

IBM MQ proprietà	Tipo di proprietà IBM MQ LLM	Tipo di proprietà LLM	Codice proprietà 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	Stringa LLM_PROP_KIND_	-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	Stringa LLM_PROP_KIND_	-1013
MQMD.ReplyToQMger	RMM_MSG_PROP_BYTES	Stringa LLM_PROP_KIND_	-1014
MQMD.PutDate	RMM_MSG_PROP_BYTES	Stringa LLM_PROP_KIND_	-1020
MQMD.PutTime	RMM_MSG_PROP_BYTES	Stringa LLM_PROP_KIND_	-1021
MQMD.ApplOriginData	RMM_MSG_PROP_BYTES	Stringa LLM_PROP_KIND_	-1022
MQPubOptions	RMM_MSG_PROP_INT32	LLM_PROP_KIND_int32	-1053

Per ulteriori informazioni su LLM, consultare la documentazione del prodotto LLM: IBM MQ Low Latency Messaging.

IBM i AmministrazioneIBM MQ for IBM i

I comandi CL sono il metodo preferito per amministrare IBM MQ su IBM i. È anche possibile utilizzare comandi MQSC, comandi PCF, comandi di controllo e gestione remota.

Informazioni su questa attività

Le attività di gestione includono la creazione, l'avvio, la modifica, la visualizzazione, l'arresto e la cancellazione di cluster, processi e oggetti di IBM MQ (gestori code, code, elenchi nomi, definizioni di processi, canali, canali di connessione client, listener, servizi e oggetti di informazioni di autenticazione).

Consultare i seguenti link per dettagli su come amministrare IBM MQ for IBM i.

Concetti correlati

Informazioni sui nomi delle librerie dei gestori code IBM MQ for IBM i

Servizi e componenti installabili su IBM i

Attività correlate

Modifica delle informazioni di configurazione IBM MQ su Multiplatforms

Impostazione della sicurezza su IBM i

“Richiamo del gestore code di messaggi non recapitabili su IBM i” a pagina 167
Su IBM MQ for IBM i, si richiama il gestore DLQ impostando il comando **STRMQMDLQ** .

Determinazione dei problemi con applicazioni IBM MQ for IBM i

Riferimenti correlati

Oggetti di sistema e predefiniti

IBM i Gestione di IBM MQ for IBM i utilizzando i comandi CL

Utilizzare queste informazioni per comprendere i comandi IBM MQ IBM i .

È possibile accedere alla maggior parte dei gruppi di comandi IBM MQ , inclusi quelli associati a gestori code, code, argomenti, canali, elenchi nomi, definizioni di processi e oggetti delle informazioni di autenticazione utilizzando il comando **WRK*** pertinente.

Il comando principale nella serie è **WRKMQM**. Questo comando consente, ad esempio, di visualizzare un elenco di tutti i gestori code sul sistema, insieme alle informazioni sullo stato. In alternativa, è possibile elaborare tutti i comandi specifici del gestore code utilizzando varie opzioni per ciascuna voce.

Dal comando **WRKMQM** è possibile selezionare aree specifiche di ciascun gestore code, ad esempio, gestione di canali, argomenti o code e da lì selezionare singoli oggetti.

Registrazione delle definizioni dell'applicazione IBM MQ

Quando si creano o personalizzano applicazioni IBM MQ , è utile conservare un record di tutte le definizioni IBM MQ create. Questo record può essere utilizzato per:

- Scopi di recupero
- Manutenzione
- Rollout delle applicazioni IBM MQ

È possibile registrare le definizioni di applicazione IBM MQ in 1 di 2 modi:

1. Creazione di programmi CL per generare le definizioni IBM MQ per il server.
2. Creazione di file di testo MQSC come membri SRC per generare le definizioni IBM MQ utilizzando il linguaggio di comandi IBM MQ multipiattaforma.

Per ulteriori dettagli sulla definizione degli oggetti coda, consultare “Amministrazione di IBM MQ utilizzando i comandi MQSC” a pagina 12 e “Utilizzo di IBM MQ Programmable Command Format” a pagina 26.

Riferimenti correlati

IBM MQ for IBM i Riferimento comandi CL

IBM i Prima di iniziare ad utilizzare IBM MQ for IBM i utilizzando i comandi CL

Utilizzare queste informazioni per avviare il sottosistema IBM MQ e creare un gestore code locale.

Prima di iniziare

Assicurarsi che il sottosistema IBM MQ sia in esecuzione (utilizzando il comando STRSBS QMQM/QMQM) e che la coda lavori associata a tale sottosistema non sia congelata. Per impostazione predefinita, il sottosistema IBM MQ e la coda lavori sono entrambi denominati QMQM nella libreria QMQM.

Informazioni su questa attività

Utilizzo della riga comandi IBM i per avviare un gestore code

Procedura

1. Creare un gestore code locale immettendo il comando CRTMQM da una riga comandi IBM i .

Quando si crea un gestore code, è possibile impostare tale gestore code come gestore code predefinito. Il gestore code predefinito (di cui può essere presente solo uno) è il gestore code a cui si applica un comando CL, se il parametro del nome del gestore code (MQMNAME) viene omesso.

2. Avviare il gestore code locale emettendo il comando STRMQM da una riga comandi IBM i .

   Se l'avvio del gestore code richiede più di pochi secondi, IBM MQ visualizzerà i messaggi di stato in modo intermittente che descrivono l'avanzamento dell'avvio. Per ulteriori informazioni su questi messaggi, consultare Messaggi e codici di errore.

Operazioni successive

È possibile arrestare un gestore code immettendo il comando ENDMQM dalla riga comandi IBM i e controllare un gestore code immettendo altri comandi IBM MQ da una riga comandi IBM i .

I gestori di code remoti non possono essere avviati in remoto, ma devono essere creati e avviati nei relativi sistemi da operatori locali. Un'eccezione è rappresentata dal caso in cui esistono funzioni operative remote (all'esterno di IBM MQ for IBM i) per abilitare tali operazioni.

L'amministratore della coda locale non può arrestare un gestore code remoto.

Nota: Come parte della disattivazione di un sistema IBM MQ , è necessario disattivare i gestori code attivi. Ciò è descritto in “In corso di quiesceIBM MQ for IBM i” a pagina 449.

IBM i Creazione di oggetti IBM MQ for IBM i

Utilizzare queste informazioni per comprendere i metodi per la creazione di oggetti IBM MQ per IBM i.

Prima di iniziare

Le seguenti attività suggeriscono vari modi in cui è possibile utilizzare IBM MQ for IBM i dalla riga comandi.

Informazioni su questa attività

Ci sono due metodi in linea per creare oggetti IBM MQ , che sono:

Procedura

1. Utilizzando un comando Create, ad esempio: il comando **Create MQM Queue** : **CRTMQMQ**
2. Utilizzo di un comando di gestione oggetto MQM, seguito da F6, ad esempio: il comando **Work with MQM Queues** : **WRKMQMQ**

Operazioni successive

Per un elenco di tutti i comandi consultare IBM MQ for IBM i Comandi CL.

Nota: Tutti i comandi MQM possono essere inoltrati dal menu Comandi gestore code messaggi. Per visualizzare questo menu, immettere GO CMDMQM sulla riga comandi e premere il tasto Enter .

Il sistema visualizza automaticamente il pannello di richiesta quando si seleziona un comando da questo menu. Per visualizzare il pannello di richieste per un comando immesso direttamente sulla riga comandi, premere F4 prima di premere il tasto Enter .

Creazione di una coda locale utilizzando il comando CRTMQMQ

Procedura

1. Immettere CHGMQM sulla riga comandi e premere il tasto F4 .
2. Nel pannello **Crea coda MQM**, immettere il nome della coda che si desidera creare nel campo Queue name . Per specificare un nome con maiuscole e minuscole, racchiudere il nome tra apici.
3. Immettere *LCL nel campo Queue type .

4. Specificare un nome gestore code, a meno che non si utilizzi il gestore code predefinito e premere il tasto `Enter` . È possibile sovrascrivere qualsiasi valore con un nuovo valore. Scorrere in avanti per visualizzare ulteriori campi. Le opzioni utilizzate per i cluster sono alla fine dell'elenco di opzioni.
5. Una volta modificati i valori, premere il pulsante `Enter` per creare la coda.

Creazione di una coda locale mediante il comando WRKMQMQ

Procedura

1. Immettere `WRKMQMQ` sulla riga comandi.
2. Immettere il nome di un Gestore code.
3. Se si desidera visualizzare il pannello di richiesta, premere F4. Il pannello di richiesta è utile per ridurre il numero di code visualizzate, specificando un nome coda generico o un tipo di coda.
4. Premere `Enter` per visualizzare il **pannello Gestione code MQM** . È possibile sovrascrivere qualsiasi valore con un nuovo valore. Scorrere in avanti per visualizzare ulteriori campi. Le opzioni utilizzate per i cluster sono alla fine dell'elenco di opzioni.
5. Premere F6 per creare una nuova coda; questo porta al pannello **CRTMQMQ** . Consultare “Creazione di una coda locale utilizzando il comando CRTMQMQ” a pagina 379 per istruzioni su come creare la coda. Una volta creata la coda, viene nuovamente visualizzato il pannello **Gestisci code MQM** . La nuova coda viene aggiunta all'elenco quando si preme F5=`Refresh`.

Modifica degli attributi del gestore code

Informazioni su questa attività

Per modificare gli attributi del gestore code specificato sul comando **CHGMQM** , specificando gli attributi e i valori che si desidera modificare. Ad esempio, utilizzare le seguenti opzioni per modificare gli attributi di `jupiter.queue.manager`:

Procedura

Immettere **CHGMQM** sulla riga comandi e premere il tasto F4 .

Risultati

Il comando modifica la coda di messaggi non instradabili utilizzata e abilita gli eventi di inibizione.

IBM i Gestione delle code locali su IBM i

Questa sezione contiene esempi di alcuni comandi che è possibile utilizzare per gestire le code locali. Tutti i comandi visualizzati sono disponibili anche utilizzando le opzioni dal **pannello dei comandi WRKMQMQ**.

Definizione di una coda locale

Per un'applicazione, il gestore code locale è il gestore code a cui è connessa l'applicazione. Le code gestite da un gestore code locale vengono dette locali per tale gestore code.

Utilizzare il comando **CRTMQMQ QTYPE *LCL** per creare una definizione di una coda locale e anche per creare la struttura dati denominata coda. È anche possibile modificare le caratteristiche della coda da quelle della coda locale predefinita.

In questo esempio, la coda definita, `orange.local.queue`, è specificata per avere le seguenti caratteristiche:

- È abilitato per le ricezioni, disabilitato per le inserimenti e opera su base FIFO (first - in - first - out).
- È una coda *ordinaria* , vale a dire, non è una coda di iniziazione o una coda di trasmissione e non genera messaggi trigger.

- La lunghezza massima della coda è 1000 messaggi; la lunghezza massima del messaggio è 2000 byte.

Il seguente comando esegue questa operazione sul gestore code predefinito:

```
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 che questa coda non è una coda di trasmissione.
2. Se si dispone già di una coda locale con il nome `orange.local.queue` sullo stesso gestore code, questo comando non riesce. Utilizzare l'attributo REPLACE *YES se si desidera sovrascrivere la definizione esistente di una coda, ma consultare anche "Modifica degli attributi della coda locale" a pagina 382.

Definizione di una coda di messaggi non recapitabili

Ogni gestore code deve avere una coda locale da utilizzare come coda di messaggi non recapitabili in modo che i messaggi che non possono essere consegnati alla destinazione corretta possano essere memorizzati per un successivo recupero. È necessario informare esplicitamente il gestore code della coda di messaggi non recapitabili. È possibile eseguire questa operazione specificando una coda di messaggi non recapitabili nel comando **CRTMQM** oppure è possibile utilizzare il comando **CHGMQM** per specificarne una in un secondo momento. È inoltre necessario definire la coda di messaggi non recapitabili prima che possa essere utilizzata.

Una coda di messaggi non recapitabili di esempio denominata `SYSTEM.DEAD.LETTER.QUEUE` viene fornita con il prodotto. Questa coda viene creata automaticamente quando si crea il gestore code. Se necessario, è possibile modificare questa definizione. Non c'è bisogno di rinominarlo, anche se si può, se ti piace.

Una coda di messaggi non recapitabili non ha requisiti speciali tranne che:

- Deve essere una coda locale.
- Il suo attributo MAXMSGL (lunghezza massima messaggio) deve consentire alla coda di contenere i messaggi più grandi che il gestore code deve gestire **più** la dimensione dell'intestazione dei messaggi non instradabili (MQDLH).

IBM MQ fornisce un gestore code di messaggi non recapitabili che consente di specificare il modo in cui i messaggi trovati in una coda di messaggi non recapitabili devono essere elaborati o rimossi. Per ulteriori informazioni, fare riferimento a "Richiamo del gestore code di messaggi non recapitabili su IBM i" a pagina 167.

Visualizzazione degli attributi dell'oggetto predefiniti

Quando si definisce un oggetto IBM MQ , vengono utilizzati tutti gli attributi non specificati dall'oggetto predefinito. Ad esempio, quando si definisce una coda locale, la coda eredita tutti gli attributi omessi nella definizione dalla coda locale predefinita, denominata `SYSTEM.DEFAULT.LOCAL.QUEUE`. Per vedere esattamente quali sono questi attributi, utilizzare il seguente comando:

```
DSPMQMQ QNAME(SYSTEM.DEFAULT.LOCAL.QUEUE) MQMNAME(MYQUEUEMANAGER)
```

Copia di una definizione di coda locale

È possibile copiare una definizione di coda utilizzando il comando CPYMQMQ . Ad esempio:

```
CPYMQMQ FROMQ('orange.local.queue') TOQ('magenta.queue') MQMNAME(MYQUEUEMANAGER)
```

Questo comando crea una coda con gli stessi attributi della coda originale `orange.local.queue`, piuttosto che quelli della coda locale predefinita del sistema.

È anche possibile utilizzare il comando **CPYMQMQ** per copiare una definizione di coda, ma sostituendo una o più modifiche agli attributi dell'originale. Ad esempio:

```
CPYMQMQ FROMQ('orange.local.queue') TOQ('third.queue') MQMNAME(MYQUEUEMANAGER)
MAXMSGLEN(1024)
```

Questo comando copia gli attributi della coda `orange.local.queue` nella coda `third.queue`, ma specifica che la lunghezza massima del messaggio sulla nuova coda deve essere di 1024 byte anziché 2000.

Nota: Quando si utilizza il comando **CPYMQMQ** , si copiano solo gli attributi della coda, non i messaggi sulla coda.

Modifica degli attributi della coda locale

È possibile modificare gli attributi della coda in due modi, utilizzando il comando **CHGMQMQ** o il comando **CPYMQMQ** con l'attributo REPLACE *YES. In “Definizione di una coda locale” a pagina 380, è stata definita la coda `orange.local.queue`. Se, ad esempio, è necessario aumentare la lunghezza massima del messaggio su questa coda a 10.000 byte.

- Utilizzando il comando **CHGMQMQ** :

  ```
  CHGMQMQ QNAME('orange.local.queue') MQMNAME(MYQUEUEMANAGER) MAXMSGLEN(10000)
  ```

  Questo comando modifica un singolo attributo, quello della lunghezza massima del messaggio; tutti gli altri attributi rimangono uguali.
- Utilizzando il comando **CRTMQMQ** con l'opzione REPLACE *YES, ad esempio:

  ```
  CRTMQMQ QNAME('orange.local.queue') QTYPE(*LCL) MQMNAME(MYQUEUEMANAGER)
  MAXMSGLEN(10000) REPLACE(*YES)
  ```

  Questo comando modifica non solo la lunghezza massima del messaggio, ma anche tutti gli altri attributi, ai quali vengono assegnati valori predefiniti. La coda è ora abilitata all'inserimento, mentre in precedenza era inibita. L'inserimento abilitato è quello predefinito, come specificato dalla coda SYSTEM.DEFAULT.LOCAL.QUEUE, a meno che non sia stato modificato.

  Se si ***riduce*** la lunghezza massima dei messaggi su una coda esistente, i messaggi esistenti non vengono influenzati. Qualsiasi nuovo messaggio, tuttavia, deve soddisfare i nuovi criteri.

Cancellazione di una coda locale

Per eliminare tutti i messaggi da una coda locale denominata `magenta.queue`, utilizzare il seguente comando:

```
CLRMQMQ QNAME('magenta.queue') MQMNAME(MYQUEUEMANAGER)
```

Non è possibile cancellare una coda se:

- Ci sono messaggi di cui non è stato eseguito il commit che sono stati inseriti nella coda nel punto di sincronizzazione.

- Un'applicazione ha attualmente la coda aperta.

Eliminazione di una coda locale

Utilizzare il comando **DLTMQMQ** per eliminare una coda locale.

Una coda non può essere eliminata se contiene messaggi di cui non è stato eseguito il commit o se è in uso.

Abilitazione di code grandi

IBM MQ supporta code più grandi di 2 GB. Consultare la documentazione del sistema operativo per informazioni su come abilitare IBM i per supportare file di grandi dimensioni.

Le informazioni sul prodotto IBM i sono disponibili in IBM Documentation.

Alcuni programmi di utilità potrebbero non essere in grado di gestire i file superiori a 2 GB. Prima di abilitare il supporto file di grandi dimensioni, consultare la documentazione del sistema operativo per informazioni sulle restrizioni su tale supporto.

IBM i Gestione delle code alias su IBM i

Questa sezione contiene esempi di alcuni comandi che è possibile utilizzare per gestire le code alias. Tutti i comandi visualizzati sono disponibili anche utilizzando le opzioni dal **pannello dei comandi WRKMQMQ**.

Una coda alias (a volte nota come alias della coda) fornisce un metodo di reindirizzamento delle chiamate MQI. Una coda alias non è una coda reale ma una definizione che si risolve in una coda reale. La definizione della coda alias contiene un nome coda di destinazione, specificato dall'attributo TGTQNAME .

Quando un'applicazione specifica una coda alias in una chiamata MQI, il gestore code risolve il nome della coda reale in fase di runtime.

Ad esempio, un'applicazione è stata sviluppata per inserire i messaggi su una coda denominata `my.alias.queue`. Specifica il nome di questa coda quando effettua una richiesta **MQOPEN** e, indirettamente, se inserisce un messaggio su questa coda. L'applicazione non è consapevole che la coda è una coda alias. Per ogni chiamata MQI che utilizza questo alias, il gestore code risolve il nome della coda reale, che potrebbe essere una coda locale o una coda remota definita su questo gestore code.

Modificando il valore dell'attributo TGTQNAME , è possibile reindirizzare chiamate MQI a un'altra coda, possibilmente su un altro gestore code. È utile per la manutenzione, la migrazione e il bilanciamento del carico.

Definizione di una coda alias

Il seguente comando crea una coda alias:

```
CRTMQMQ QNAME('my.alias.queue') QTYPE(*ALS) TGTQNAME('yellow.queue')
MQMNAME(MYQUEUEMANAGER)
```

Questo comando reindirizza le chiamate MQI che specificano `my.alias.queue` alla coda `yellow.queue`. Il comando non crea la coda di destinazione; le chiamate MQI hanno esito negativo se la coda `yellow.queue` non esiste al runtime.

Se si modifica la definizione dell'alias, è possibile reindirizzare le chiamate MQI a un'altra coda. Ad esempio:

```
CHGMQMQ QNAME('my.alias.queue') TGTQNAME('magenta.queue') MQMNAME(MYQUEUEMANAGER)
```

Questo comando reindirizza chiamate MQI a un altra coda, `magenta.queue`.

È inoltre possibile utilizzare le code alias per fare in modo che una singola coda (la coda di destinazione) sembri avere attributi differenti per applicazioni differenti. Questa operazione viene eseguita definendo due alias, uno per ogni applicazione. Si supponga che vi siano due applicazioni:

- L'applicazione ALPHA può inserire messaggi su `yellow.queue`, ma non è consentita la ricezione di messaggi da esso.
- L'applicazione BETA può ricevere messaggi da `yellow.queue`, ma non è consentito inserirlo.

È possibile eseguire questa operazione utilizzando i comandi riportati di seguito:

```
/* 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 utilizza il nome della coda `alphas.alias.queue` nelle relative chiamate MQI; BETA utilizza il nome della coda `betas.alias.queue`. Entrambi accedono alla stessa coda, ma in modi diversi.

È possibile utilizzare l'attributo REPLACE *YES quando si definiscono le code alias, nello stesso modo in cui si utilizzano questi attributi con le code locali.

Utilizzo di altri comandi con code alias

È possibile utilizzare i comandi appropriati per visualizzare o modificare gli attributi della coda alias. Ad esempio:

```
* 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 Utilizzo delle code modello su IBM i

Questa sezione contiene esempi di alcuni dei comandi che è possibile utilizzare per gestire le code modello. Tutti i comandi visualizzati sono disponibili anche utilizzando le opzioni dal **pannello dei comandi WRKMQMQ**.

Un gestore code crea una coda dinamica se riceve una chiamata MQI da un'applicazione che specifica un nome coda che è stato definito come una coda modello. Il nome della nuova coda dinamica viene generato dal gestore code quando viene creata la coda. Una coda modello è un modello che specifica gli attributi di qualsiasi coda dinamica da essa creata.

Le code modello forniscono un metodo conveniente per le applicazioni per creare le code come sono richieste.

Definizione di una coda modello

Si definisce una coda modello con un insieme di attributi nello stesso modo in cui si definisce una coda locale. Le code modello e le code locali hanno la stessa serie di attributi, tranne per il fatto che sulle code modello è possibile specificare se le code dinamiche create sono temporanee o permanenti. (Le code permanenti vengono conservate durante i riavvii del gestore code, mentre quelle temporanee non lo sono). Ad esempio:

```
CRTMQMQ QNAME('green.model.queue') QTYPE(*MDL) DFNTYPE(*PERMDYN)
```

Questo comando crea una definizione di coda modello. Dall'attributo DFNTYPE , le code effettive create da questo template sono code dinamiche permanenti. Gli attributi non specificati vengono copiati automaticamente dalla coda predefinita SYSYTEM.DEFAULT.MODEL.QUEUE .

È possibile utilizzare l'attributo REPLACE *YES quando si definiscono le code modello, nello stesso modo in cui si utilizzano le code locali.

Utilizzo di altri comandi con code modello

È possibile utilizzare i comandi appropriati per visualizzare o modificare gli attributi di una coda modello. Ad esempio:

```
/* 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 Utilizzo del trigger su IBM i

Utilizzare queste informazioni per informazioni sull'attivazione e sulle definizioni di processo.

IBM MQ fornisce una funzione per avviare automaticamente un'applicazione quando vengono soddisfatte determinate condizioni su una coda. Un esempio delle condizioni è quando il numero di messaggi su una coda raggiunge un numero specificato. Questa funzione è denominata *attivazione* ed è descritta in dettaglio in Trigger canali.

Cos' è l'attivazione?

Il gestore code definisce alcune condizioni come eventi trigger. Se il trigger è abilitato per una coda e si verifica un evento trigger, il gestore code invia un messaggio trigger a una coda denominata coda di iniziazione. La presenza del messaggio trigger sulla coda di iniziazione indica che si è verificato un evento trigger.

I messaggi trigger generati dal gestore code non sono persistenti. Ciò ha l'effetto di ridurre la registrazione (migliorando quindi le prestazioni) e di ridurre i duplicati durante il riavvio, in modo da migliorare il tempo di riavvio.

Cos' è il controllo trigger?

Il programma che elabora la coda di iniziazione viene chiamato un'applicazione di controllo dei trigger e la sua funzione è quella di leggere il messaggio del trigger e intraprendere l'azione appropriata, in base alle informazioni contenute nel messaggio del trigger. Di solito, questa azione consente di avviare un'altra applicazione per elaborare la coda che ha causato la generazione del messaggio trigger. Dal punto di vista del gestore code, non c'è nulla di speciale nell'applicazione di controllo dei trigger - è un'altra applicazione che legge i messaggi da una coda (la coda di iniziazione).

Modifica degli attributi di inoltro del lavoro del controllo trigger

Il controllo trigger fornito come comando **STRMQMTRM** inoltra un lavoro per ogni messaggio trigger utilizzando la descrizione lavoro predefinita di sistema, QDFTJOBD. Ciò ha delle limitazioni in quanto i lavori inoltrati vengono chiamati sempre QDFTJOBD e hanno gli attributi della descrizione lavoro predefinita, incluso l'elenco librerie, *SYSVAL. IBM MQ fornisce un metodo per sovrascrivere questi attributi. Ad esempio, è possibile personalizzare i lavori inoltrati in modo che abbiano nomi di lavoro più significativi come segue:

1. Nella descrizione del lavoro, specificare la descrizione desiderata, ad esempio i valori di registrazione.
2. Specificare i dati di ambiente della definizione di processo utilizzata nel processo di attivazione:

```
CHGMQMPRC PRCNAME(MY_PROCESS) MQMNAME(MHA3) ENVDATA ('JOBD(MYLIB/TRIGJOBD)')
```

   Il controllo trigger esegue un SBMJOB utilizzando la descrizione specificata.

È possibile sovrascrivere altri attributi di SBMJOB specificando la parola chiave e il valore appropriati nei dati di ambiente della definizione del processo. L'unica eccezione è la parola chiave CMD perché questo attributo è riempito dal controllo trigger. Segue un esempio del comando per specificare i dati di ambiente della definizione del processo in cui devono essere modificati sia il nome lavoro che la descrizione:

```
CHGMQMPRC PRCNAME(MY_PROCESS) MQMNAME(MHA3) ENVDATA ('JOBD(MYLIB/TRIGJOB)
JOB(TRIGGER)')
```

Definizione di una coda dell'applicazione per il trigger

Una coda dell'applicazione è una coda locale utilizzata dalle applicazioni per la messaggistica, tramite MQI. L'attivazione richiede la definizione di un certo numero di attributi della coda sulla coda dell'applicazione. L'attivazione è abilitata dall'attributo TRGENBL .

In questo esempio, un evento trigger deve essere generato quando ci sono 100 messaggi con priorità 5 o superiore sulla coda locale `motor.insurance.queue`, come segue:

```
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)
```

dove i parametri sono:

MQMNAME(MYQUEUEMANAGER)
Il nome del gestore code.

QNAME('motor.insurance.queue')
Il nome della coda dell'applicazione definita.

PRCNAME('motor.insurance.quote.process')
Il nome dell'applicazione che deve essere avviata da un programma di controllo trigger.

MAXMSGLEN(2000)
La lunghezza massima dei messaggi sulla coda.

DFTMSGPST(*YES)
I messaggi su questa coda sono persistenti per impostazione predefinita.

INITQNAME('motor.ins.init.queue')
Il nome della coda di iniziazione su cui il gestore code deve inserire il messaggio trigger.

TRGENBL(*YES)
Il valore dell'attributo trigger.

TRGTYPE(*DEPTH)
Un evento trigger viene generato quando il numero di messaggi con la priorità richiesta (**TRGMSGPTY**) raggiunge il numero specificato in **TRGDEPTH**.

TRGDEPTH(100)
Il numero di messaggi richiesti per generare un evento trigger.

TRGMSGPTY(5)
La priorità dei messaggi che devono essere conteggiati dal gestore code per decidere se generare un evento trigger. Vengono conteggiati solo i messaggi con priorità 5 o superiore.

Definizione di una coda di iniziazione

Quando si verifica un evento trigger, il gestore code inserisce un messaggio trigger nella coda di avvio specificata nella definizione della coda dell'applicazione. Le code di iniziazione non dispongono di impostazioni speciali, ma è possibile utilizzare la seguente definizione della coda locale `motor.ins.init.queue` come guida:

```
CRTMQMQ MQMNAME(MYQUEUEMANAGER) QNAME('motor.ins.init.queue') QTYPE(*LCL)
GETENBL(*YES) SHARE(*NO) TRGTYPE(*NONE)
MAXMSGL(2000)
MAXDEPTH(1000)
```

Creazione di una definizione di processo

Utilizzare il comando **CRTMQMPRC** per creare una definizione di processo. Una definizione di processo associa una coda dell'applicazione all'applicazione che deve elaborare i messaggi dalla coda. Ciò viene eseguito tramite l'attributo PRCNAME nella coda dell'applicazione `motor.insurance.queue`. Il seguente comando crea il processo richiesto, `motor.insurance.quote.process`, identificato in questo esempio:

```
CRTMQMPRC MQMNAME(MYQUEUEMANAGER) PRCNAME('motor.insurance.quote.process')
TEXT('Insurance request message processing')
APPTYPE(*OS400) APPID(MQTEST/TESTPROG)
USRDATA('open, close, 235')
```

dove i parametri sono:

MQMNAME(MYQUEUEMANAGER)
Il nome del gestore code.

PRCNAME('motor.insurance.quote.process')
Il nome della definizione del processo.

TEXT('Insurance request message processing')
Una descrizione del programma di applicazione a cui si riferisce questa definizione. Questo testo viene visualizzato quando si utilizza il comando **DSPMQMPRC** . Questo può aiutare a identificare cosa fa il processo. Se si utilizzano spazi nella stringa, è necessario racchiudere la stringa tra virgolette singole.

APPTYPE(*OS400)
Il tipo di applicazione da avviare.

APPID(MQTEST/TESTPROG)
Il nome del file eseguibile dell'applicazione, specificato come nome file completo.

USRDATA('open, close, 235')
Dati definiti dall'utente, che possono essere utilizzati dall'applicazione.

Visualizzazione della definizione del processo

Utilizzare il comando **DSPMQMPRC** per esaminare i risultati della definizione. Ad esempio:

```
MQMNAME(MYQUEUEMANAGER) DSPMQMPRC('motor.insurance.quote.process')
```

È anche possibile utilizzare il comando **CHGMQMPRC** per modificare una definizione di processo esistente e il comando **DLTMQMPRC** per eliminare una definizione di processo.

IBM i Comunicazione tra due sistemi IBM MQsu IBM i

Questo esempio di codifica illustra come impostare due sistemi IBM MQ for IBM i , utilizzando comandi CL, in modo che possano comunicare tra loro.

I sistemi sono denominati SYSTEMA e SYSTEMBe il protocollo di comunicazione utilizzato è TCP/IP.

Effettuare la seguente procedura:

1. Creare un gestore code su SYSTEMA, richiamandolo QMGRA1.

```
CRTMQM   MQMNAME(QMGRA1) TEXT('System A - Queue +
Manager 1') UDLMSGQ(SYSTEM.DEAD.LETTER.QUEUE)
```

2. Avviare questo gestore code.

```
STRMQM   MQMNAME(QMGRA1)
```

3. Definire gli IBM MQ oggetti su SYSTEMA necessari per l'invio di messaggi a un gestore code su 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. Creare un gestore code su SYSTEMB, richiamandolo QMGRB1.

```
CRTMQM   MQMNAME(QMGRB1) TEXT('System B - Queue +
Manager 1') UDLMSGQ(SYSTEM.DEAD.LETTER.QUEUE)
```

5. Avviare il gestore code su SYSTEMB.

```
STRMQM   MQMNAME(QMGRB1)
```

6. Definire gli oggetti IBM MQ necessari per ricevere i messaggi dal gestore code su 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. Infine, avviare un listener TCP/IP su SYSTEMB in modo che il canale possa essere avviato. Questo esempio utilizza la porta predefinita 1414.

```
STRMQMLSR MQMNAME(QMGRB1)
```

È ora possibile inviare i messaggi di verifica tra SYSTEMA e SYSTEMB. Utilizzando uno degli esempi forniti, inserire una serie di messaggi nella coda remota su SYSTEMA.

Avviare il canale su SYSTEMA, utilizzando il comando **STRMQMCHL**oppure utilizzando il comando **WRKMQMCHL** e immettendo una richiesta di avvio (Opzione 14) rispetto al canale mittente.

Il canale deve passare allo stato RUNNING e i messaggi vengono inviati alla coda TARGETB su SYSTEMB.

Controllare i messaggi immettendo il comando:

```
WRKMQMMSG QNAME(TARGETB) MQMNAME(QMGRB1).
```

IBM i Definizioni di risorse di esempio su IBM i

Questo esempio contiene il programma CL AMQSAMP4 sample IBM i .

```
/*********************************************************************/
/*                                                                   */
/* 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 Modi alternativi di amministrazione di IBM MQ for IBM i

L'uso dei comandi CL è il metodo preferito di gestione di IBM MQ for IBM i. Tuttavia, è possibile utilizzare vari altri metodi di gestione, inclusi i comandi MQSC, i comandi PCF, i comandi di controllo e la gestione remota.

Informazioni su questa attività

Di solito, si utilizzano i comandi CL IBM i per gestire IBM MQ for IBM i. Per una panoramica di questi comandi, consultare “Gestione di IBM MQ for IBM i utilizzando i comandi CL” a pagina 378.

È anche possibile utilizzare i comandi MQSC e i comandi PCF come descritto nei sottoargomenti ed è possibile utilizzare i comandi di controllo come descritto in “Amministrazione di IBM MQ for Multiplatforms utilizzando i comandi di controllo” a pagina 10.

È possibile utilizzare gli eventi di strumentazione IBM MQ per monitorare il funzionamento dei gestori code. Consultare Eventi di strumentazione per informazioni sugli eventi di strumentazione IBM MQ e su come utilizzarli.

Utilizzare uno qualsiasi dei metodi di gestione descritti nei seguenti argomenti secondari come alternativa all'utilizzo dei comandi CL IBM i :

IBM i Amministrazione locale e remota su IBM i

Gli oggetti IBM MQ for IBM i vengono gestiti localmente o in remoto.

Informazioni su questa attività

Per *amministrazione locale* si intende l'esecuzione di attività di amministrazione su qualsiasi gestore code definito sul sistema locale. In IBM MQ, è possibile considerarlo come amministrazione locale poiché non sono coinvolti canali IBM MQ , ossia la comunicazione è gestita dal sistema operativo. Per eseguire questo tipo di attività, è necessario accedere al sistema remoto e immettere i comandi da lì oppure creare un processo che possa emettere i comandi per l'utente.

IBM MQ supporta la gestione da un singolo punto tramite la *amministrazione remota*. La gestione remota consiste nell'inviare messaggi di controllo PCF (Programmable Command Format) a `SYSTEM.ADMIN.COMMAND.QUEUE` sul gestore code di destinazione.

Esistono diversi modi per generare messaggi PCF. Questi sono descritti nei seguenti passi.

Procedura

- Scrivere un programma utilizzando i messaggi PCF. Consultare “Gestione utilizzando comandi PCF su IBM i” a pagina 394.
- Scrivere un programma utilizzando MQAI, che invia messaggi PCF. Consultare “Utilizzo di MQAI per semplificare l'utilizzo dei PCF” a pagina 38.
- Utilizzare IBM MQ Explorer, disponibile con IBM MQ for Windows, che consente di utilizzare una GUI (graphical user interface) e genera i messaggi PCF corretti. Consultare “Utilizzo di IBM MQ Explorer con IBM MQ for IBM i” a pagina 394.
- Utilizzare **STRMQMMQSC** per inviare comandi indirettamente a un gestore code remoto. Consultare “Amministrazione utilizzando i comandi MQSC su IBM i” a pagina 392.

  Ad esempio, è possibile immettere un comando remoto per modificare una definizione di coda su un gestore code remoto.

  Alcuni comandi non possono essere emessi in questo modo, in particolare, creando o avviando i gestori code e avviando i server dei comandi. Per eseguire questo tipo di attività, è necessario collegarsi al sistema remoto e immettere i comandi da tale ubicazione oppure creare un processo che possa emettere i comandi per conto dell'utente.

IBM i Amministrazione utilizzando i comandi MQSC su IBM i

Su IBM i, si crea un elenco di comandi in un file di script, quindi si esegue il file utilizzando il comando **STRMQMMQSC** . I comandi MQSC vengono utilizzati per gestire gli oggetti del gestore code, inclusi il gestore code stesso, le code, le definizioni dei processi, gli elenchi nomi, i canali, i canali di connessione client, i listener, i servizi, gli argomenti e gli oggetti delle informazioni di autenticazione.

Informazioni su questa attività

I comandi di script IBM MQ (MQSC) sono scritti in formato leggibile, in testo EBCDIC. I comandi MQSC vengono immessi in un gestore code utilizzando il comando CL **STRMQMMQSC** IBM MQ . Questo metodo

è solo un metodo batch, che prende il suo input da un file fisico di origine nel sistema della libreria del server. Il nome predefinito per questo file fisico origine è QMQSC.

Attenzione: Non utilizzare la libreria QTEMP come libreria di origine per STRMQMMQSC, poiché l'utilizzo della libreria QTEMP è limitato. È necessario utilizzare un'altra libreria come file di input per il comando.

Per la portabilità tra ambienti IBM MQ , limitare la lunghezza della riga nei file di comandi MQSC a 72 caratteri. Utilizzare il segno più per indicare che il comando continua sulla riga successiva.

Gli attributi oggetto specificati in MQSC vengono mostrati in maiuscolo in questo argomento (ad esempio, RQMNAME), sebbene non siano sensibili al maiuscolo / minuscolo.

Nota:

1. Il formato di un file MQSC non dipende dalla sua ubicazione nel file system.
2. I nomi attributo MQSC sono limitati a otto caratteri.
3. I comandi MQSC sono disponibili su tutte le piattaforme IBM MQ .

Per una descrizione di ciascun comando MQSC e della sua sintassi, consultare Comandi MQSC.

Procedura

1. Creare il file di origine QMQ.

   IBM MQ for IBM i non fornisce un file di origine denominato QMQSC. Per elaborare i comandi MQSC, è necessario creare il file di origine QMQSC in una libreria di propria scelta, immettendo il seguente comando:

   ```
   CRTSRCPF FILE(MYLIB/QMQSC) RCDLEN(240) TEXT('IBM MQ - MQSC Source')
   ```

2. Lavorare con i membri.

   L'origine MQSC è contenuta nei membri all'interno di questo file di origine. Per gestire i membri, immettere il comando seguente:

   ```
   WRKMBRPDM MYLIB/QMQSC
   ```

   È ora possibile aggiungere nuovi membri e gestire quelli esistenti.

   ```
   .
   .
   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. Estrarre da un file di comandi MQSC, `myprog.in`, che mostra un comando MQSC (DEFINE QLOCAL) con i relativi attributi.

Informazioni correlate

Amministrazione di IBM MQ utilizzando i comandi MQSC

IBM i Gestione utilizzando comandi PCF su IBM i

Lo scopo dei comandi PCF (IBM MQ programmable command format) è consentire la programmazione delle attività di amministrazione in un programma di amministrazione. In questo modo è possibile creare code e definizioni di processo e modificare gestori code da un programma.

I comandi PCF coprono la stessa gamma di funzioni fornite dai comandi MQSC. Tuttavia, diversamente dai comandi MQSC, i comandi PCF e le risposte non sono in un formato di testo che è possibile leggere.

È possibile scrivere un programma per emettere comandi PCF su qualsiasi gestore code nella rete da un singolo nodo. In questo modo, è possibile centralizzare e automatizzare le attività di gestione.

Ogni comando PCF è una struttura dati incorporata nella parte di dati dell'applicazione di un messaggio IBM MQ . Ogni comando viene inviato al gestore code di destinazione utilizzando la funzione MQI MQPUT allo stesso modo di qualsiasi altro messaggio. Il server dei comandi sul gestore code che riceve il messaggio lo interpreta come messaggio di comandi ed esegue il comando. Per ottenere le repliche, l'applicazione emette una chiamata MQGET e i dati di risposta vengono restituiti in un'altra struttura dati. La domanda può quindi elaborare la risposta e agire di conseguenza.

Brevemente, queste sono alcune delle cose che il programmatore dell'applicazione deve specificare per creare un messaggio di comando PCF:

Descrittore messaggio
Si tratta di un descrittore di messaggi IBM MQ standard, in cui:

- Il tipo di messaggio (*MsgType*) è MQMT_REQUEST.
- Il formato del messaggio (*Format*) è MQFMT_ADMIN.

Dati applicazione
Contiene il messaggio PCF che include l'intestazione PCF, in cui:

- Il tipo di messaggio PCF (*Type*) specifica MQCFT_COMMAND.
- L'identificativo del comando specifica il comando, ad esempio, *Change Queue* (MQCMD_CHANGE_Q).

I PCF di escape sono comandi PCF che contengono comandi MQSC all'interno del testo del messaggio. È possibile utilizzare i PCF per inviare comandi a un gestore code remoto. Consultare “Utilizzo di MQAI per semplificare l'utilizzo dei PCF” a pagina 38 per ulteriori informazioni.

Per una descrizione completa delle strutture dati PCF e come implementarle, consultare Strutture per comandi e risposte.

IBM i Utilizzo di IBM MQ Explorer con IBM MQ for IBM i

Utilizzare queste informazioni per gestire IBM MQ for IBM i utilizzando IBM MQ Explorer.

IBM MQ for Windows (piattaformax86) e IBM MQ per Linux (piattaformex86 e x86-64) forniscono un'interfaccia di gestione denominata Esplora risorse di IBM MQ per eseguire le attività di gestione come alternativa all'utilizzo dei comandi CL, di controllo o MQSC.

IBM MQ Explorer consente di eseguire la gestione locale o remota della propria rete da un computer su cui è in esecuzione Windows (piattaformax86) o Linux (piattaformex86 e x86-64), puntando IBM MQ Explorer ai gestori code e ai cluster a cui si è interessati.

Con IBM MQ Explorer, puoi:

- Avviare e arrestare un gestore code (solo sulla macchina locale).
- Definire, visualizzare e modificare le definizioni degli oggetti IBM MQ come code, argomenti e canali.
- Esaminare i messaggi su una coda.
- Avviare e arrestare un canale.
- Visualizzare le informazioni di stato su un canale.
- Visualizzare i gestori code in un cluster.

- Verificare quali applicazioni, utenti o canali hanno una particolare coda aperta.
- Creare un nuovo cluster di gestori code utilizzando la procedura guidata **Crea nuovo cluster** .
- Aggiungere un gestore code a un cluster utilizzando il wizard **Aggiungi gestore code al cluster** .
- Gestire l'oggetto delle informazioni di autenticazione, utilizzato con la sicurezza del canale TLS (Transport Layer Security).

Utilizzando la guida online, è possibile:

- Definire e controllare varie risorse, inclusi gestori code, code, canali, definizioni di processi, canali di connessione client, listener, argomenti, servizi, elenchi nomi e cluster.
- Avviare o arrestare un gestore code e i relativi processi associati.
- Visualizzare i gestori code e i relativi oggetti associati sulla stazione di lavoro o da altre stazioni di lavoro.
- Verificare lo stato dei gestori code, dei cluster e dei canali.

Accertarsi di aver soddisfatto i requisiti riportati di seguito prima di tentare di utilizzare IBM MQ Explorer per gestire IBM MQ su una macchina server. Verificare che:

1. Un server dei comandi è in esecuzione per qualsiasi gestore code gestito, avviato sul server dal comando CL **STRMQMCSVR**.
2. Esiste un listener TCP/IP adatto per ogni gestore code remoto. Questo è il listener IBM MQ avviato dal comando **STRMQMLSR** .
3. Il canale di connessione server, denominato `SYSTEM.ADMIN.SVRCONN`, esiste su ogni gestore code remoto. È necessario creare questo canale. È obbligatorio per ogni gestore code remoto gestito. Senza di esso, la gestione remota non è possibile.
4. Verificare che la coda `SYSTEM.MQEXPLORER.REPLY.MODEL` esista.

IBM i Gestione del server dei comandi per la gestione in remoto su IBM i

Utilizzare queste informazioni per informazioni sulla gestione remota di IBM MQ for IBM i Command Server.

Ciascun gestore code può avere un server dei comandi associato. Un server dei comandi elabora i comandi in entrata dai gestori code remoti o i comandi PCF dalle applicazioni. Presenta i comandi al gestore code per l'elaborazione e restituisce un codice di completamento o un messaggio dell'operatore in base all'origine del comando.

Un server dei comandi è obbligatorio per tutte le attività di gestione che coinvolgono PCF, MQAI e anche per la gestione remota.

Nota: Per la gestione remota, è necessario assicurarsi che il gestore code di destinazione sia in esecuzione. Altrimenti, i messaggi che contengono i comandi non possono lasciare il gestore code da cui vengono emessi. Invece, questi messaggi vengono accodati nella coda di trasmissione locale che serve il gestore code remoto. Evitare questa situazione, se possibile.

Esistono comandi di controllo separati per avviare e arrestare il server dei comandi. È possibile eseguire le operazioni descritte nelle sezioni seguenti utilizzando Esplora risorse di IBM MQ .

Avvio e arresto del server dei comandi

Per avviare il server dei comandi, utilizzare questo comando CL:

```
STRMQMCSVR MQMNAME('saturn.queue.manager')
```

dove `saturn.queue.manager` è il gestore code per cui viene avviato il server dei comandi.

Per arrestare il server dei comandi, utilizzare uno dei comandi CL riportati di seguito:

1.
```
ENDMQMCSVR MQMNAME('saturn.queue.manager') OPTION(*CNTRLD)
```

per eseguire un arresto controllato, dove `saturn.queue.manager` è il gestore code per cui si sta arrestando il server dei comandi. Questa è l'opzione predefinita, che significa che `OPTION(*CNTRLD)` può essere omesso.

2.
```
ENDMQMCSVR MQMNAME('saturn.queue.manager') OPTION(*IMMED)
```
per eseguire un arresto immediato, dove `saturn.queue.manager` è il gestore code per cui viene arrestato il server dei comandi.

Visualizzazione dello stato del server dei comandi

Per la gestione remota, verificare che sia in esecuzione il server dei comandi sul gestore code di destinazione. Se non è in esecuzione, i comandi remoti non possono essere elaborati. Tutti i messaggi che contengono i comandi vengono accodati nella coda comandi del gestore code di destinazione `SYSTEM.ADMIN.COMMAND.QUEUE`.

Per visualizzare lo stato del server dei comandi per un gestore code, denominato `saturn.queue.manager`, il comando CL è:

```
DSPMQMCSVR MQMNAME('saturn.queue.manager')
```

Immettere questo comando sulla macchina di destinazione. Se il server dei comandi è in esecuzione, viene visualizzato il pannello mostrato in Figura 21 a pagina 396 :

```
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. Visualizza pannello MQM Command Server

IBM i Esecuzione dei comandi della console Web

Devi configurare il tuo ambiente come descritto nel seguente testo, affinché i comandi Qshell correlati alla console web vengano eseguiti correttamente su IBM MQ for IBM i.

Informazioni su questa attività

Quando Qshell viene avviato, inizializza le tabelle interne per l'elaborazione dei comandi basati sul CCSID del lavoro. Affinché i comandi Qshell relativi alla console Web vengano eseguiti correttamente su IBM i, devi configurare il tuo ambiente.

Una locale viene impostata impostando la variabile di ambiente LANG sul nome percorso di un oggetto locale. Ad esempio, per impostare la locale per l'inglese americano, la variabile di ambiente LANG è impostata come segue:

```
LANG=/QSYS.LIB/EN_US.LOCALE
```

Nella Qshell, è possibile controllare l'impostazione immettendo la serie di comandi per elencare le variabili di ambiente. Di solito è LANG che può influire sulla locale per l'ambiente di runtime. Può anche avere LC_ALL.

Per eseguire correttamente i comandi Qshell, l'impostazione dell'ambiente locale deve essere congruente con l'impostazione del lavoro.

Procedura

Utilizzare il comando CL DSPJOB JOB (JobNumber/USERProfile/JobName)

a) Selezionare l'opzione 2 per visualizzare gli attributi di definizione lavoro.

b) I seguenti attributi devono essere congruenti con l'impostazione dell'ambiente LANG o LC_ALL

- Identificativo lingua
- Identificativo paese o regione
- CCSID (Coded character set identifier)

Ad esempio, se

```
LANG=/QSYS.LIB/FR_FR.LOCALE
```

gli attributi del lavoro devono essere:

- ID lingua..... FRA
- ID paese o regione..... FR
- CCSID (Coded character set ID).. 297

Operazioni successive

Per ulteriori informazioni sul supporto lingua nazionale, consultare l'argomento IBM Documentation Considerazioni sul supporto lingua nazionale (NLS).

IBM i Gestione del lavoro per IBM i

Queste informazioni descrivono il modo in cui IBM MQ gestisce le richieste di lavoro e descrivono le opzioni disponibili per assegnare una priorità e controllare i lavori associati a IBM MQ.

Avviso

Non modificare gli oggetti di gestione lavoro IBM MQ a meno che non si comprendano completamente i concetti di gestione lavoro IBM i e IBM MQ .

Ulteriori informazioni sui sottosistemi e le descrizioni dei lavori sono disponibili in Gestione lavoro nella documentazione del prodotto IBM i . Prestare particolare attenzione alle sezioni Avvio dei lavori e Lavori batch.

IBM MQ for IBM i integra l'ambiente IBM i UNIX e i thread IBM i . **Non** apportare alcuna modifica agli oggetti in IFS (Integrated File System).

Durante le normali operazioni, un gestore code IBM MQ avvia una serie di lavori batch per eseguire attività differenti. Per impostazione predefinita questi lavori batch vengono eseguiti nel sottosistema QMQM creato quando IBM MQ è installato.

La gestione del lavoro si riferisce al processo di personalizzazione delle attività IBM MQ per ottenere le prestazioni ottimali dal sistema o per semplificare l'amministrazione.

Ad esempio, puoi:

- Modificare la priorità di esecuzione dei lavori per rendere un gestore code più reattivo di un altro.
- Reindirizzare l'output di un numero di lavori ad una particolare coda di emissione.

- Eseguire tutti i lavori di un determinato tipo in uno specifico sottosistema.
- Isolare gli errori in un sottosistema.

La gestione lavoro viene eseguita creando o modificando le descrizioni lavoro associate ai lavori IBM MQ . È possibile configurare la gestione del lavoro per:

- Un'intera installazione di IBM MQ .
- Singoli gestori code.
- Singoli lavori per singoli gestori code.

IBM i Attività IBM MQ per IBM i

Questa è una tabella dei job IBM MQ for IBM i e una breve descrizione di ognuno.

Quando un gestore code è in esecuzione, vengono visualizzati alcuni o tutti i seguenti lavori batch in esecuzione nel profilo utente QMQM nel sottosistema IBM MQ . I lavori sono descritti brevemente in Tabella 21 a pagina 398.

È possibile visualizzare tutti i lavori connessi ad un gestore code utilizzando l'opzione 22 nel pannello **Gestione gestore code** (WRKMQM). È possibile visualizzare i listener utilizzando il comando WRKMQMLSR.

Tabella 21. Attività IBM MQ .

Nome lavoro	**Funzione**
AMQZMUC0	Gestore programmi di utilità. Questo lavoro esegue i programmi di utilità del gestore code critici, ad esempio il gestore della catena journal.
AMQZXMA0	Il controller di esecuzione che è il primo lavoro avviato dal gestore code. Gestisce le richieste MQCONN e avvia i processi dell'agent per elaborare le chiamate API IBM MQ .
AMQZFUMA	OAM (Object authority manager).
AMQZLAA0	Gli agent del gestore code che eseguono la maggior parte del lavoro per le applicazioni che si connettono al gestore code utilizzando MQCNO_STANDARD_BINDING.
AMQZLSA0	Agent gestore code.
AMQZMUF0	Gestore utilità
AMQZMGR0	Controller processi. Questo lavoro viene utilizzato per avviare e gestire listener e servizi.
AMQZMUR0	Gestore programmi di utilità. Questo lavoro esegue i programmi di utilità del gestore code critici, ad esempio il gestore della catena journal.
AMQFQPUB	Daemon di pubblicazione / sottoscrizione accodato.
AMQFCXBA	Lavoro di lavoro del broker.
RUNMQBRK	Lavoro di controllo broker.
AMQRMPPA	Lavoro di pool del processo del canale.
AMQCRSTA	Responder del canale richiamato da TCP/IP.
AMQCRS6B	LU62 canale ricevente e connessione client (vedere nota).
AMQRRMFA	Gestore repository per i cluster.
AMQCLMAA	Listener TCP/IP senza thread.
AMQPCSEA	Il processore comandi PCF che gestisce le richieste di gestione remota e PCF.
RUNMQTRM	Controllo trigger.
RUNMQDLQ	Gestore code di messaggi non recapitabili.

Tabella 21. Attività IBM MQ . (Continua)	
Nome lavoro	**Funzione**
RUNMQCHI	L'iniziatore del canale.
RUNMQCHL	Lavoro del canale mittente avviato per ogni canale mittente.
RUNMQLSR	Listener TCP/IP con thread.
AMQRCMLA	Canale MQSC e processore comandi PCF.

Nota: Il lavoro del ricevitore LU62 viene eseguito nel sottosistema di comunicazione e acquisisce le proprietà di runtime dalle voci di instradamento e di comunicazioni utilizzate per avviare il lavoro. Per ulteriori informazioni, consultare Fine avviata (Ricevitore) .

IBM i Oggetti di gestione lavoro su IBM i

Quando si installa IBM MQ , vengono forniti diversi oggetti nella libreria QMQM per assistere nella gestione del lavoro. Questi oggetti sono quelli necessari per l'esecuzione dei lavori IBM MQ nel relativo sottosistema.

Le descrizioni dei lavori di esempio vengono fornite per due dei lavori batch IBM MQ . Se non viene fornita alcuna descrizione lavoro specifica per un lavoro IBM MQ , viene eseguito con la descrizione lavoro predefinita QMQMJOBD.

Gli oggetti di gestione del lavoro forniti quando si installa IBM MQ sono elencati in Tabella 22 a pagina 399 e gli oggetti creati per un gestore code sono elencati in Tabella 23 a pagina 399

Nota: È possibile trovare gli oggetti di gestione del lavoro nella libreria QMQM e gli oggetti del gestore code nella libreria del gestore code.

Tabella 22. Oggetti di gestione lavoro		
Nome	**Tipo**	**Descrizione**
AMQZLAA0	*JOBD	La descrizione lavoro utilizzata dai processi dell'agente IBM MQ
AMQZLSA0	*JOBD	L'agent del gestore code dei bind isolati
AMQZXMA0	*JOBD	La descrizione lavoro utilizzata dai controller di esecuzione IBM MQ
QMQM	*SBSD	Il sottosistema in cui vengono eseguiti tutti i lavori IBM MQ
QMQM	*JOBQ	La coda lavori collegata al sottosistema fornito
QMQMJOBD	*JOBD	La descrizione lavoro IBM MQ predefinita, utilizzata se non esiste una descrizione lavoro specifica per un lavoro
MQMMSG	*MSGQ	La coda messaggi predefinita per i lavori IBM MQ .
QMQMRUN20	*CLS	Una descrizione di classe per lavori IBM MQ ad alta priorità
QMQMRUN35	*CLS	Una descrizione di classe per lavori IBM MQ con priorità media
QMQMRUN50	*CLS	Una descrizione della classe per lavori IBM MQ a bassa priorità

Tabella 23. Oggetti di gestione lavoro creati per un gestore code		
Nome	**Tipo**	**Descrizione**
AMQA000000	*JRNRCV	Ricevitore giornale locale
JRN AMQA	*JRN	Giornale locale

Tabella 23. Oggetti di gestione lavoro creati per un gestore code (Continua)		
Nome	**Tipo**	**Descrizione**
RNINF AMQJR	*USRSPC	Lo spazio utente aggiornato con gli ultimi ricevitori di giornale richiesti per l'avvio e il ripristino dei supporti di un gestore code. Questo spazio utente può essere interrogato da un'applicazione per determinare quali ricevitori di giornale richiedono l'archiviazione e quali possono essere cancellati in modo sicuro.
MQAJRNMSG	*MSGQ	Coda messaggi giornale locale
AMQCRC6B	*PGM	Programma per avviare la connessione LU6.2
AMQRFOLD	*FILE	File di definizione del canale del gestore code migrato
MQMMSG	*MSGQ	Coda messaggi del gestore code

IBM i Come IBM MQ utilizza gli oggetti di gestione del lavoro su IBM i

Queste informazioni descrivono il modo in cui IBM MQ utilizza gli oggetti di gestione lavoro e forniscono esempi di configurazione.

Attenzione: Non modificare le impostazioni della voce della coda lavori nel sottosistema QMQM per limitare il numero di lavori consentiti nel sottosistema in base alla priorità. Se si tenta di eseguire questa operazione, è possibile arrestare l'esecuzione dei lavori IBM MQ essenziali dopo che sono stati inoltrati e causare l'errore dell'avvio del gestore code.

Per comprendere come configurare la gestione del lavoro, è necessario prima comprendere come IBM MQ utilizza le descrizioni del lavoro.

La descrizione lavoro utilizzata per avviare il lavoro controlla molti attributi del lavoro. Ad esempio:

- La coda lavori su cui il lavoro è accodato e su quale sottosistema viene eseguito il lavoro.
- I dati di instradamento utilizzati per avviare il lavoro e la classe che il lavoro utilizza per i parametri di runtime.
- La coda di emissione utilizzata dal lavoro per i file di stampa.

Il processo di avvio di un lavoro IBM MQ può essere considerato in tre fasi:

1. IBM MQ seleziona una descrizione lavoro.

   IBM MQ utilizza la seguente tecnica per determinare quale descrizione lavoro utilizzare per un lavoro batch:

   a. Ricercare nella libreria del gestore code una descrizione lavoro con lo stesso nome del lavoro. Fare riferimento a Informazioni sui IBM MQ for IBM i nomi delle librerie dei gestori code per maggiori dettagli sulla libreria dei gestori code.

   b. Ricercare nella libreria del gestore code la descrizione lavoro predefinita QMQMJOBD.

   c. Ricercare nella libreria QMQM una descrizione lavoro con lo stesso nome del lavoro.

   d. Utilizzare la descrizione lavoro predefinita, QMQMJOBD, nella libreria QMQM.

2. Il lavoro viene inoltrato alla coda lavori.

   Le descrizioni dei lavori fornite con IBM MQ sono state impostate, per impostazione predefinita, per inserire i lavori nella coda lavori QMQM nella libreria QMQM. La coda lavori QMQM è collegata al sistema secondario QMQM fornito, quindi per impostazione predefinita i lavori vengono avviati nel sistema secondario QMQM.

3. Il lavoro entra nel sottosistema e passa attraverso le fasi di instradamento.

   Quando il lavoro entra nel sottosistema, i dati di instradamento specificati nella descrizione lavoro vengono utilizzati per trovare le voci di instradamento per il lavoro.

I dati di instradamento devono corrispondere a una delle voci di instradamento definite nel sistema secondario QMQM, e ciò definisce quale delle classi fornite (QMQMRUN20, QMQMRUN35o QMQMRUN50) viene utilizzata dal lavoro.

Nota: Se i lavori IBM MQ non sembrano essere in fase di avvio, assicurarsi che il sistema secondario sia in esecuzione e che la coda lavori non sia congelata,

Se sono stati modificati gli oggetti di gestione lavoro IBM MQ , assicurarsi che tutto sia associato correttamente. Ad esempio, se si specifica una coda lavori diversa da QMQM/QMQM nella descrizione del lavoro, assicurarsi che venga eseguito un ADDJOBQE per il sottosistema, ossia QMQM.

È possibile creare una descrizione lavoro per ogni lavoro documentato in Tabella 21 a pagina 398 utilizzando il seguente foglio di lavoro come esempio:

```
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
```

Se si risponde No a tutte queste domande, creare una descrizione del lavoro globale QMQMJOBD nella libreria QMQM.

La coda messaggi IBM MQ

Una coda messaggi IBM MQ , QMQMMSG, viene creata in ogni libreria del gestore code. I messaggi del sistema operativo vengono inviati a questa coda alla fine dei lavori del gestore code e IBM MQ invia i messaggi alla coda. Ad esempio, per notificare quali ricevitori di giornale sono necessari all'avvio. Mantenere il numero di messaggi in questa coda messaggi ad una dimensione gestibile per renderne più semplice il monitoraggio.

IBM i Esempi di sistema predefiniti per IBM i

Questi esempi mostrano come funziona un'installazione di IBM MQ non modificata quando alcuni dei lavori standard vengono inoltrati all'avvio del gestore code.

Innanzitutto, viene avviato il lavoro del controller di esecuzione AMQZXMA0 .

1. Immettere il comando **STRMQM** per il gestore code TESTQM.
2. IBM MQ ricerca la libreria del gestore code QMTESTQM, prima per la descrizione lavoro AMQZXMA0, e quindi la descrizione lavoro QMQMJOBD.

   Nessuna di queste descrizioni lavoro esiste, quindi IBM MQ ricerca la descrizione lavoro AMQZXMA0 nella libreria del prodotto QMQM. Questa descrizione lavoro esiste, quindi viene utilizzata per inoltrare il lavoro.
3. La descrizione del lavoro utilizza la coda lavori predefinita IBM MQ , quindi il lavoro viene inoltrato alla coda lavori QMQM/QMQM.
4. I dati di instradamento sulla descrizione del lavoro AMQZXMA0 è QMQMRUN20, quindi il sistema ricerca le voci di instradamento del sottosistema per una che corrisponda a tali dati.

   Per impostazione predefinita, la voce di instradamento con numero di sequenza 9900 ha dati di confronto che corrispondono a QMQMRUN20, quindi il lavoro viene avviato con la classe definita su tale voce di instradamento, denominata anche QMQMRUN20.
5. La classe QMQM/QMQMRUN20 ha la priorità di esecuzione impostata su 20, quindi il lavoro AMQZXMA0 viene eseguito nel sottosistema QMQM con la stessa priorità della maggior parte dei lavori interattivi sul sistema.

IBM i Configurazione degli esempi di gestione lavoro su IBM i

Utilizzare queste informazioni per informazioni su come modificare e creare descrizioni lavoro IBM MQ per modificare gli attributi di runtime dei lavori IBM MQ .

La chiave per la flessibilità della gestione lavoro IBM MQ risiede nel modo a due livelli in cui IBM MQ ricerca le descrizioni dei lavori:

- Se si creano o si modificano le descrizioni lavoro in una libreria del gestore code, tali modifiche sovrascrivono le descrizioni lavoro globali in QMQM, ma le modifiche sono locali e influiscono solo su quel particolare gestore code.
- Se si creano o si modificano le descrizioni lavoro globali nella libreria QMQM, tali descrizioni lavoro influiscono su tutti i gestori code sul sistema, a meno che non vengano sovrascritte localmente per i singoli gestori code.

1. Il seguente esempio aumenta la priorità dei lavori di controllo canale per un singolo gestore code.

   Per fare in modo che i lavori del gestore repository e dell'iniziatore del canale, AMQRRMFA e RUNMQCHI, vengano eseguiti il più rapidamente possibile per il gestore code TESTQM, effettuare le seguenti operazioni:

   a. Creare i duplicati locali della descrizione del lavoro QMQM/QMQMJOBD con i nomi dei processi IBM MQ che si desidera controllare nella libreria del gestore code. Ad esempio:

   ```
   CRTDUPOBJ OBJ(QMQMJOBD) FROMLIB(QMQM) OBJTYPE(*JOBD) TOLIB(QMTESTQM)
   NEWOBJ(RUNMQCHI)
   CRTDUPOBJ OBJ(QMQMJOBD) FROMLIB(QMQM) OBJTYPE(*JOBD) TOLIB(QMTESTQM)
   NEWOBJ(AMQRRMFA)
   ```

   b. Modificare il parametro dei dati di instradamento nella descrizione lavoro per assicurarsi che i lavori utilizzino la classe QMQMRUN20 .

   ```
   CHGJOBD JOBD(QMTESTQM/RUNMQCHI) RTGDTA('QMQMRUN20')
   CHGJOBD JOBD(QMTESTQM/AMQRRMFA) RTGDTA('QMQMRUN20')
   ```

   I lavori AMQRRMFA e RUNMQCHI per il gestore code TESTQM ora:

   - Utilizzare le nuove descrizioni dei lavori locali nella libreria del gestore code
   - Eseguire con priorità 20, poiché la classe QMQMRUN20 viene utilizzata quando i lavori entrano nel sottosistema.

2. Il seguente esempio definisce una nuova classe di priorità di esecuzione per il sottosistema QMQM.

   a. Creare una classe duplicata nella libreria QMQM, per consentire ad altri gestori code di accedere alla classe, immettendo il seguente comando:

   ```
   CRTDUPOBJ OBJ(QMQMRUN20) FROMLIB(QMQM) OBJTYPE(*CLS) TOLIB(QMQM)
   NEWOBJ(QMQMRUN10)
   ```

   b. Modificare la classe in modo che abbia la nuova priorità di esecuzione immettendo il seguente comando:

   ```
   CHGCLS CLS(QMQM/QMQMRUN10) RUNPTY(10)
   ```

   c. Aggiungere la nuova definizione di classe al sottosistema immettendo il comando seguente:

   ```
   ADDRTGE SBSD(QMQM/QMQM) SEQNBR(8999) CMPVAL('QMQMRUN10') PGM(QSYS/QCMD)
   CLS(QMQM/QMQMRUN10)
   ```

   Nota: È possibile specificare un qualsiasi valore numerico per il numero di sequenza di instradamento, ma i valori devono essere in ordine sequenziale. Questo numero di sequenza indica al sottosistema l'ordine in cui le voci di instradamento devono essere ricercate per una corrispondenza dei dati di instradamento.

   d. Modificare la descrizione del lavoro locale o globale per utilizzare la nuova classe di priorità immettendo il seguente comando:

```
CHGJOBD JOBD(QMQMlibname/QMQMJOBD) RTGDTA('QMQMRUN10')
```

Ora tutti i lavori del gestore code associati al QMlibraryname utilizzano una priorità di esecuzione di 10.

3. Il seguente esempio esegue un gestore code nel proprio sottosistema

   Per eseguire tutti i lavori per il gestore code TESTQM nel sottosistema QBATCH, effettuare le seguenti operazioni:

   a. Creare un duplicato locale della descrizione lavoro QMQM/QMQMJOBD nella libreria del gestore code con il comando

   ```
   CRTDUPOBJ OBJ(QMQMJOBD) FROMLIB(QMQM) OBJTYPE(*JOBD) TOLIB(QMTESTQM)
   ```

   b. Modificare il parametro della coda lavori nella descrizione lavoro per assicurarsi che i lavori utilizzino la coda lavori QBATCH.

   ```
   CHGJOBD JOBD(QMTESTQM/QMQMJOBD) JOBQ(*LIBL/QBATCH)
   ```

   Nota: La coda lavori è associata alla descrizione del sottosistema. Se si rileva che i lavori si trovano sulla coda lavori, verificare che la definizione della coda lavori sia definita su SBSD. Utilizzare il comando DSPSBSD per il sistema secondario e scegliere l'opzione 6, Voci coda lavori.

   Tutti i lavori per il gestore code TESTQM ora:

   - Utilizzare la nuova descrizione del lavoro predefinito locale nella libreria del gestore code
   - Vengono inoltrati alla coda lavori QBATCH.

   Per garantire che i lavori vengano instradati e impostati correttamente le priorità:

   - Creare le voci di instradamento per i lavori IBM MQ nel sottosistema QBATCH oppure
   - Basarsi su una voce di instradamento catch-all che richiama QCMD, indipendentemente dai dati di instradamento utilizzati.

     Questa opzione funziona solo se l'opzione del numero massimo di lavori attivi per la coda lavori QBATCH è impostato su *NOMAX. Il valore predefinito del sistema è 1.

4. Il seguente esempio crea un altro sottosistema IBM MQ

   a. Creare un sottosistema duplicato nella libreria QMQM immettendo il seguente comando:

   ```
   CRTDUPOBJ OBJ(QMQM) FROMLIB(QMQM) OBJTYPE(*SBSD) TOLIB(QMQM) NEWOBJ(QMQM2)
   ```

   b. Rimuovere la coda lavori QMQM immettendo il seguente comando:

   ```
   RMVJOBQE SBSD(QMQM/QMQM2) JOBQ(QMQM/QMQM)
   ```

   c. Creare una nuova coda lavori per il sottosistema immettendo il seguente comando:

   ```
   CRTJOBQ JOBQ(QMQM/QMQM2) TEXT('Job queue for IBM MQ Queue Manager')
   ```

   d. Aggiungere una voce coda lavori al sottosistema immettendo il comando seguente:

   ```
   ADDJOBQE SBSD(QMQM/QMQM2) JOBQ(QMQM/QMQM2) MAXACT(*NOMAX)
   ```

   e. Creare un QMQMJOBD duplicato nella libreria del gestore code immettendo il seguente comando:

   ```
   CRTDUPOBJ OBJ(QMQMJOBD) FROMLIB(QMQM) OBJTYPE(*JOBD) TOLIB(QMlibraryname)
   ```

f. Modificare la descrizione lavoro per utilizzare la nuova coda lavori immettendo il seguente comando:

```
CHGJOBD JOBD(QMlibraryname/QMQMJOBD) JOBQ(QMQM/QMQM2)
```

g. Avviare il sottosistema immettendo il seguente comando:

```
STRSBS SBSD(QMQM/QMQM2)
```

Nota:

a. È possibile specificare il sottosistema in qualsiasi libreria. Se per qualsiasi motivo il prodotto viene reinstallato o la libreria QMQM viene sostituita, tutte le modifiche apportate vengono rimosse.

b. Tutti i lavori del gestore code associati al QMlibraryname vengono ora eseguiti nel sottosistema QMQM2.

IBM i Disponibilità, backup, ripristino e riavvio su IBM i

Utilizzare queste informazioni per comprendere in che modo IBM MQ for IBM i utilizza il supporto journal IBM i per facilitare la propria strategia di backup e ripristino.

Prima di leggere questa sezione, è necessario avere dimestichezza con i metodi di backup e ripristino standard di IBM i e con l'utilizzo dei giornali e dei relativi ricevitori di giornale associati su IBM i. Per informazioni su questi argomenti, consultare Backup e ripristino.

Per comprendere la strategia di backup e ripristino, è necessario prima comprendere il modo in cui IBM MQ for IBM i organizza i propri dati nel file system IBM i e nell'IFS (integrated file system).

IBM MQ for IBM i conserva i dati in una singola libreria per ciascuna istanza del gestore code e nei file di flusso nel file system IFS.

Le librerie specifiche del gestore code contengono i journal, i ricevitori del journal e gli oggetti richiesti per controllare la gestione del lavoro del gestore code. I file e gli indirizzari IFS contengono i file di configurazione IBM MQ , le descrizioni degli oggetti IBM MQ e i dati in essi contenuti.

Ogni modifica a questi oggetti, ripristinabile in un errore di sistema, viene registrata in un giornale *prima* che venga applicata all'oggetto appropriato. Ciò ha l'effetto che tali modifiche possono essere recuperate riproducendo le informazioni registrate nel giornale.

È possibile configurare IBM MQ for IBM i per utilizzare più istanze del gestore code su server differenti per fornire una maggiore disponibilità del gestore code e accelerare il ripristino in caso di errore del server o del gestore code.

IBM i Journal del gestore code su IBM i

Utilizzare queste informazioni per comprendere in che modo IBM MQ for IBM i utilizza i giornali nelle proprie operazioni per controllare gli aggiornamenti agli oggetti locali.

Ogni libreria del gestore code contiene un journal per tale gestore code e il journal ha il nome QM *GRLIB*/AMQ *A* JRN, dove QM *GRLIB* è il nome della libreria del gestore code e *A* è una lettera, A nel caso di un gestore code a istanza singola, che è univoco per l'istanza del gestore code.

QM *GRLIB* prende il nome QM, seguito dal nome del gestore code in un formato univoco. Ad esempio, un gestore code denominato TEST ha una libreria di gestori code denominata QMTEST. La libreria del gestore code può essere specificata quando si crea un gestore code utilizzando il comando **CRTMQM** .

I giornali hanno ricevitori di giornale associati che contengono le informazioni che si stanno registrando su giornale. I ricevitori sono oggetti a cui le informazioni possono essere solo accodate e si riempiranno alla fine.

I ricevitori di giornale utilizzano spazio su disco prezioso con informazioni non aggiornate. Tuttavia, è possibile inserire le informazioni nella memoria permanente per ridurre al minimo questo problema. Un

ricevitore di giornale è collegato al giornale in qualsiasi momento. Se il ricevitore di giornale raggiunge la sua dimensione di soglia predeterminata, viene scollegato e sostituito da un nuovo ricevitore di giornale. È possibile specificare la soglia dei ricevitori del journal quando si crea un gestore code utilizzando **CRTMQM** e il parametro **THRESHOLD** .

I ricevitori del journal associati al journal locale IBM MQ per IBM i esistono in ciascuna libreria del gestore code e adottano una convenzione di denominazione come segue:

```
AMQ Arnnnnn
```

dove

Una
è una lettera A-Z. È A per i gestori code a istanza singola. Varia per le diverse istanze di un gestore code a più istanze.

nnnnn
è decimale `00000 to 99999` che viene incrementato di 1 per il giornale successivo nella sequenza.

r
è decimale `0 to 9`, che viene incrementato di 1 ogni volta che un ricevitore viene ripristinato.

La sequenza dei journal si basa sulla data. Tuttavia, la denominazione del giornale successivo si basa sulle regole seguenti:

1. AMQArnnnnn passa a AMQAr(nnnnn+1)e nnnnn esegue il wrapping quando raggiunge 99999. Ad esempio, AMQA099999 va a AMQA000000e AMQA999999 va a AMQA900000.
2. Se già esiste un giornale con un nome generato dalla regola 1, il messaggio CPI70E3 viene inviato alla coda messaggi QSYSOPR e la commutazione automatica del ricevitore si arresta.

   Il ricevitore attualmente collegato continua ad essere utilizzato fino a quando non si esamina il problema e si collega manualmente un nuovo ricevitore.
3. Se non è disponibile alcun nuovo nome nella sequenza (ovvero, tutti i nomi di giornale possibili si trovano sul sistema), è necessario effettuare entrambe le seguenti operazioni:
   a. Eliminare i giornali non più necessari (consultare “Gestione giornale su IBM i” a pagina 410).
   b. Registrare le modifiche di giornale nell'ultimo ricevitore di giornale utilizzando (**RCDMQMIMG**) e quindi ripetere il passo precedente. Ciò consente il riutilizzo dei vecchi nomi dei ricevitori di giornale.

Il giornale AMQAJRN utilizza l'opzione MNGRCV(*SYSTEM) per consentire al sistema operativo di modificare automaticamente i ricevitori di giornale quando viene raggiunta la soglia. Per ulteriori informazioni su come il sistema gestisce i ricevitori, consultare *IBM i Backup e ripristino*.

Il valore di soglia predefinito del ricevitore di giornale è 100.000 KB. È possibile impostare questo valore su un valore maggiore quando si crea il gestore code. Il valore iniziale dell'attributo LogReceiverDimensione viene scritto nella stanza LogDefaults del file mqs.ini .

Quando un ricevitore di giornale supera la soglia specificata, il ricevitore viene scollegato e viene creato un nuovo ricevitore di giornale, ereditando gli attributi dal ricevitore precedente. Le modifiche apportate agli attributi LogReceiverSize o LogASP dopo la creazione di un gestore code vengono ignorate quando il sistema collega automaticamente un nuovo ricevitore di giornale

Per ulteriori dettagli sulla configurazione del sistema, consultare Modifica delle informazioni di configurazione di IBM MQ su Multiplatforms .

Se è necessario modificare la dimensione dei ricevitori di giornale dopo che il gestore code è stato creato, creare un nuovo ricevitore di giornale e impostarne il proprietario su QMQM utilizzando i seguenti comandi:

```
CRTJRNRCV JRNRCV(QM GRLIB/AMQ Arnnnnn) THRESHOLD(xxxxxx) +
TEXT('MQM LOCAL JOURNAL RECEIVER')
CHGOBJOWN OBJ(QM GRLIB/AMQ Arnnnnn) OBJTYPE(*JRNRCV) NEWOWN(QMQM)
```

dove

QMGRLIB
È il nome della libreria del gestore code

Una
È l'identificativo dell'istanza (di norma A).

rnnnnn
Indica il successivo ricevitore di giornale nella sequenza di denominazione descritta precedentemente

xxxxxx
Indica la nuova soglia del ricevitore (in KB)

Nota: La dimensione massima del ricevitore è regolata dal sistema operativo. Per controllare questo valore, consultare la parola chiave THRESHOLD sul comando **CRTJRNRCV** .

Ora collegare il nuovo ricevitore al giornale AMQAJRN con il comando:

```
CHGJRN JRN(QMGRLIB/AMQ A JRN) JRNRCV(QMGRLIB/AMQ Annnnnn)
```

Consultare “Gestione giornale su IBM i” a pagina 410 per i dettagli su come gestire questi ricevitori di giornale.

IBM i *Utilizzo del journal del gestore code su IBM i*

Utilizzare queste informazioni per comprendere in che modo IBM MQ for IBM i utilizza i giornali nelle proprie operazioni per controllare gli aggiornamenti agli oggetti locali.

Gli aggiornamenti permanenti alle code messaggi si verificano in due fasi. I record che rappresentano l'aggiornamento vengono prima scritti nel giornale, quindi il file di coda viene aggiornato.

I ricevitori di giornale possono quindi diventare più aggiornati dei file della coda. Per garantire che l'elaborazione del riavvio inizi da un punto congruente, IBM MQ utilizza i punti di controllo.

Un punto di controllo è un momento in cui il record descritto nel giornale è uguale al record nella coda. Il punto di controllo stesso è costituito dalla serie di record journal necessari per riavviare il gestore code. Ad esempio, lo stato di tutte le transazioni (ossia, le unità di lavoro) attive al momento del punto di controllo.

I punti di controllo vengono generati automaticamente da IBM MQ. Vengono presi quando il gestore code viene avviato e arrestato e dopo un certo numero di operazioni vengono registrate.

È possibile forzare un gestore code a prendere un punto di controllo immettendo il comando RCDMQMIMG su tutti gli oggetti su un gestore code e visualizzando i seguenti risultati:

```
RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) MQMNAME(Q_MGR_NAME) DSPJRNDTA(*YES)
```

Man mano che le code gestiscono ulteriori messaggi, il record del punto di controllo diventa incongruente con lo stato corrente delle code.

Quando IBM MQ viene riavviato, individua l'ultimo record del punto di controllo nel log. Queste informazioni vengono conservate nel file del punto di controllo aggiornato alla fine di ogni punto di controllo. Il record del punto di controllo rappresenta il punto di coerenza più recente tra il log e i dati. I dati da questo punto di controllo vengono utilizzati per ricreare le code esistenti al momento del punto di controllo. Quando le code vengono ricreate, il log viene riprodotto in avanti per riportare le code allo stato in cui si trovavano prima dell'errore di sistema o della chiusura.

Per comprendere in che modo IBM MQ utilizza il journal, considerare il caso di una coda locale denominata TESTQ nel gestore code TEST. Questo è rappresentato dal file IFS:

```
/QIBM/UserData/mqm/qmgrs/TEST/queues
```

Se un messaggio specificato viene inserito in questa coda e quindi richiamato dalla coda, le azioni che si verificano vengono mostrate nella figura Figura 22 a pagina 407.

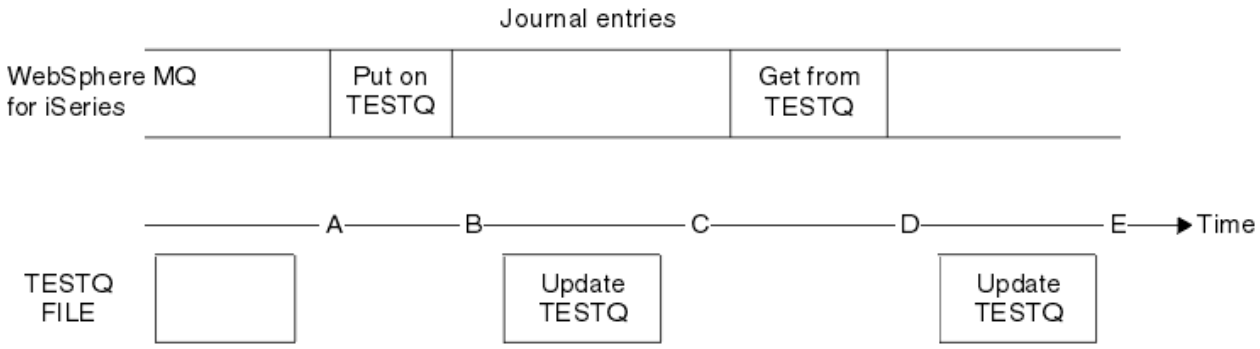


Figura 22. Sequenza di eventi durante l'aggiornamento degli oggetti MQM

I cinque punti, da A a E, mostrati nel diagramma rappresentano i punti nel tempo che definiscono i seguenti stati:

A
La rappresentazione del file IFS della coda è congruente con le informazioni contenute nel giornale.

B
Una voce di giornale viene scritta nel giornale che definisce un'operazione `Put` sulla coda.

C
L'aggiornamento appropriato viene effettuato sulla coda.

D
Una voce di giornale viene scritta nel giornale che definisce un'operazione `Get` dalla coda.

E
L'aggiornamento appropriato viene effettuato sulla coda.

La chiave per le funzioni di recupero di IBM MQ for IBM i è che l'utente può salvare la rappresentazione del file IFS di TESTQ come all'ora Ae successivamente ripristinare la rappresentazione del file IFS di TESTQ come all'ora E, ripristinando l'oggetto salvato e riproducendo le voci nel giornale dall'ora A in poi.

Questa strategia viene utilizzata da IBM MQ for IBM i per ripristinare i messaggi persistenti dopo un malfunzionamento del sistema. IBM MQ ricorda una particolare voce nei ricevitori del giornale e garantisce che all'avvio riproduce le voci nei giornali da questo punto in poi. Questa voce di avvio viene periodicamente ricalcolata in modo che IBM MQ debba eseguire solo la ripetizione minima necessaria al successivo avvio.

IBM MQ fornisce il ripristino individuale degli oggetti. Tutte le informazioni persistenti relative ad un oggetto vengono registrate nei giornali IBM MQ for IBM i locali. Qualsiasi oggetto IBM MQ danneggiato o danneggiato può essere completamente ricostruito dalle informazioni contenute nel giornale.

Per ulteriori informazioni su come il sistema gestisce i ricevitori, consultare “Disponibilità, backup, ripristino e riavvio su IBM i” a pagina 404.

IBM i *Immagini multimediali su IBM i*

Su IBM i, un'immagine di supporto è una copia completa di un oggetto IBM MQ registrato nel giornale. Alcuni oggetti corrotti o danneggiati possono essere automaticamente recuperati dalla loro immagine multimediale.

Un oggetto IBM MQ di lunga durata può rappresentare un gran numero di voci di giornale, tornando al punto in cui è stato creato. Per evitare ciò, IBM MQ for IBM i ha il concetto di immagine multimediale di un oggetto.

Questa immagine di supporto è una copia completa dell'oggetto IBM MQ registrato nel giornale. Se viene presa un'immagine di un oggetto, l'oggetto può essere ricostruito riproducendo le voci di giornale da questa immagine in poi. La voce nel giornale che rappresenta il punto di ripetizione per ogni oggetto IBM MQ viene indicata come voce di recupero del supporto. IBM MQ tiene traccia di:

- Voce di ripristino supporto per ciascun oggetto gestore code.

- La voce meno recente da questa serie (per i dettagli, consultare il messaggio di errore AMQ7462 in "Gestione giornale su IBM i" a pagina 410 .

Le immagini dell'oggetto *CTLG e dell'oggetto *MQM vengono prese regolarmente perché questi oggetti sono fondamentali per il riavvio del gestore code.

Le immagini di altri oggetti vengono prese quando è conveniente. Per impostazione predefinita, le immagini di tutti gli oggetti vengono prese quando un gestore code viene arrestato utilizzando il comando **ENDMQM** con il parametro ENDCCTJOB (*YES). Questa operazione può richiedere una quantità di tempo considerevole per i gestori code molto grandi. Se è necessario chiudere rapidamente, specificare il parametro RCDMQMIMG (*NO) con ENDCCTJOB (*YES). In questi casi, si consiglia di registrare un'immagine di supporto completa nei journal dopo il riavvio del gestore code, utilizzando il seguente comando:

```
RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) MQMNAME(Q_MGR_NAME)
```

IBM MQ registra automaticamente un'immagine di un oggetto, se trova un punto conveniente in cui un oggetto può essere descritto in modo compatto da una piccola voce nel giornale. Tuttavia, ciò potrebbe non verificarsi mai per alcuni oggetti, ad esempio, code che contengono in modo congruente un numero elevato di messaggi.

Invece di consentire alla data della voce di ripristino del supporto meno recente di continuare per un periodo inutilmente lungo, utilizzare il comando IBM MQ RCDMQMIMG, che consente di eseguire manualmente un'immagine degli oggetti selezionati.

Recupero da immagini multimediali

IBM MQ recupera automaticamente alcuni oggetti dalla relativa immagine del supporto se viene rilevato che sono danneggiati o danneggiati. In particolare, ciò si applica agli speciali oggetti *MQM e *CTLG come parte del normale avvio del gestore code. Se una transazione del punto di sincronizzazione era incompleta al momento dell'ultimo arresto del gestore code, anche qualsiasi coda interessata viene ripristinata automaticamente, al fine di completare l'operazione di avvio.

È necessario ripristinare altri oggetti manualmente, utilizzando il IBM MQ comando RCRMQMOBJ. Questo comando riproduce le voci nel giornale per ricreare l'oggetto IBM MQ . Se un oggetto IBM MQ viene danneggiato, le uniche azioni valide sono eliminarlo o ricrearlo con questo metodo. Notare, tuttavia, che i messaggi non persistenti non possono essere recuperati in questo modo.

IBM i *Checkpoint su IBM MQ for IBM i*

I punti di controllo vengono utilizzati in vari momenti per fornire un punto di partenza congruente noto per il ripristino.

Il thread del checkpoint all'interno del processo AMQZMUC0 è responsabile dell'acquisizione del punto di controllo nei punti seguenti:

- Avvio del gestore code (STRMQM).
- Chiusura del gestore code (ENDMQM).
- Dopo che è trascorso un periodo di tempo dall'ultimo punto di controllo (il periodo predefinito è 30 minuti) e un numero minimo di record di log è stato scritto dal punto di controllo precedente (il valore predefinito è 100).
- Dopo che è stato scritto un numero di record di log. Il valore predefinito è 10 000.
- Dopo che la dimensione della soglia del giornale è stata superata e un nuovo ricevitore di giornale è stato creato automaticamente.
- Quando un'immagine del supporto completo viene presa con:

```
RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) MQMNAME(Q_MGR_NAME) DSPJRNDTA(*YES)
```

IBM i Backup dei dati IBM MQ for IBM i

Utilizzare queste informazioni per comprendere i due tipi di backup IBM MQ per ogni gestore code.

Per ogni gestore code, esistono due tipi di backup IBM MQ da considerare:

- Backup di dati e journal.

  Per garantire che entrambe le serie di dati siano congruenti, eseguire questa operazione solo dopo aver chiuso il gestore code.
- Backup journal.

  È possibile eseguire questa operazione mentre il gestore code è attivo.

Per entrambi i metodi, è necessario trovare i nomi della directory IFS del gestore code e della libreria del gestore code. È possibile trovarle nel file di configurazione IBM MQ (mqs.ini). Per ulteriori informazioni, consultare Modifica delle informazioni di configurazione IBM MQ su Multiplatforms.

Utilizzare le procedure riportate di seguito per eseguire entrambi i tipi di backup:

Backup di dati e journal di uno specifico gestore code

Nota: Non utilizzare una richiesta salva - mentre - attivo quando il gestore code è in esecuzione. Tale richiesta non può essere completata a meno che non venga eseguito il commit o il rollback di tutte le definizioni di commit con modifiche in sospeso. Se questo comando viene utilizzato quando il gestore code è attivo, le connessioni del canale potrebbero non terminare normalmente. Utilizzare sempre la seguente procedura.

1. Creare un ricevitore di giornale vuoto, utilizzando il comando:

```
CHGJRN JRN(QMTEST/AMQAJRN) JRNRCV(*GEN)
```

2. Utilizzare il comando **RCDMQMIMG** per registrare un'immagine MQM per tutti gli oggetti IBM MQ , quindi forzare un punto di controllo utilizzando il comando:

```
RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) DSPJRNDTA(*YES) MQMNAME(TEST)
```

3. Terminare i canali e accertarsi che il gestore code non sia in esecuzione. Se il gestore code è in esecuzione, arrestarlo con il comando **ENDMQM** .
4. Eseguire il backup della libreria del gestore code immettendo il seguente comando:

```
SAVLIB LIB(QMTEST)
```

5. Eseguire il backup delle directory IFS del gestore code immettendo il seguente comando:

```
SAV DEV(...) OBJ(('/QIBM/UserData/mqm/qmgrs/test'))
```

Backup journal di un particolare gestore code

Poiché tutte le informazioni rilevanti sono conservate nei giornali, finché si esegue un salvataggio completo in un determinato momento, le copie di riserva parziali possono essere eseguite salvando i ricevitori di giornale. Queste registrano tutte le modifiche dall'ora del backup completo e vengono eseguite immettendo i seguenti comandi:

1. Creare un ricevitore di giornale vuoto, utilizzando il comando:

```
CHGJRN JRN(QMTEST/AMQAJRN) JRNRCV(*GEN)
```

2. Utilizzare il comando **RCDMQMIMG** per registrare un'immagine MQM per tutti gli oggetti IBM MQ , quindi forzare un punto di controllo utilizzando il comando:

```
RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) DSPJRNDTA(*YES) MQMNAME(TEST)
```

3. Salvare i ricevitori di giornale utilizzando il seguente comando:

```
SAVOBJ OBJ(AMQ*) LIB(QMTEST) OBJTYPE(*JRNRCV) .........
```

Una semplice strategia di backup è quella di eseguire un backup completo delle librerie IBM MQ ogni settimana ed eseguire un backup giornaliero del journal. Questo, ovviamente, dipende da come hai impostato la strategia di backup per la tua azienda.

IBM i *Gestione giornale su IBM i*

Come parte della strategia di backup, prendersi cura dei ricevitori di giornale. È utile eliminare i ricevitori di giornale dalle librerie IBM MQ per vari motivi:

- Per liberare spazio; questo si applica a tutti i ricevitori di giornale
- Per migliorare le prestazioni all'avvio (STRMQM)
- Per migliorare le prestazioni della ricreazione di oggetti (RCRMQMOBJ)

Prima di cancellare un ricevitore di giornale, è necessario fare attenzione che si disponga di una copia di backup e che non sia più necessario il ricevitore di giornale.

I ricevitori di journal possono essere rimossi dalla libreria del gestore code *dopo* che sono stati scollegati dai journal e salvati, purché siano disponibili per il ripristino se necessario per un'operazione di recupero.

Il concetto di gestione del giornale viene mostrato in Figura 23 a pagina 411.

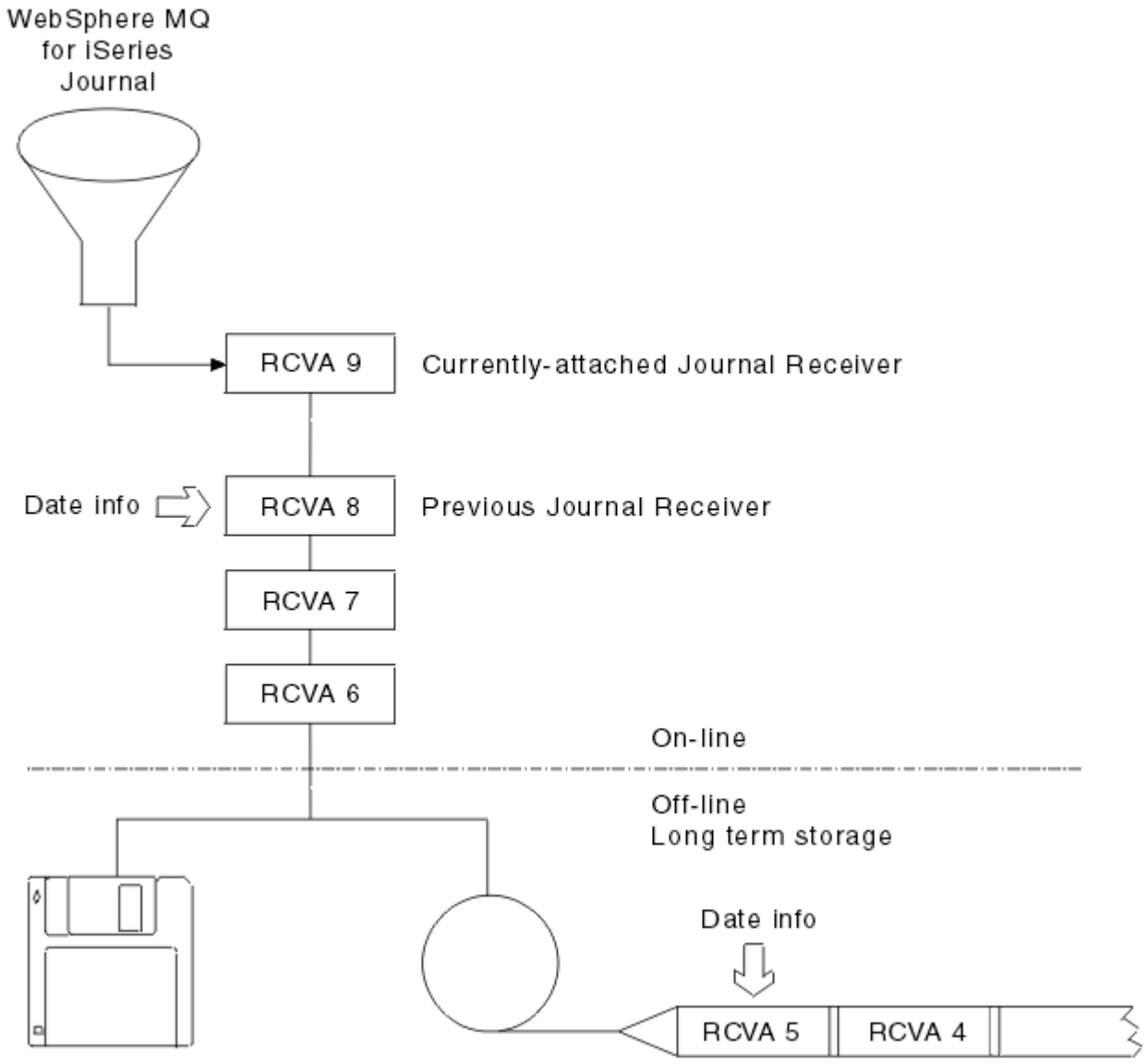


Figura 23. Registrazione su giornale su IBM i

È importante sapere fino a che punto nei journal IBM MQ è probabile che sia necessario, al fine di determinare quando un ricevitore di giornale di cui è stato eseguito il backup può essere rimosso dalla libreria del gestore code e quando il backup stesso può essere eliminato.

IBM MQ invia due messaggi alla coda messaggi del gestore code (QMQMMSG nella libreria del gestore code) per determinare questa ora. Questi messaggi vengono emessi all'avvio, quando si modifica un ricevitore di giornale locale e si utilizza RCDMQIMG per forzare un punto di controllo. I due messaggi sono:

AMQ7460
Punto di ripristino avvio. Questo messaggio definisce la data e l'ora della voce di avvio da cui IBM MQ riproduce il giornale nel caso di un passaggio di ripristino di avvio. Se il ricevitore di giornale che contiene questo record è disponibile nelle librerie IBM MQ , questo messaggio contiene anche il nome del ricevitore di giornale che contiene il record.

AMQ7462
Voce di ripristino del supporto meno recente. Questo messaggio definisce la data e l'ora della voce meno recente da utilizzare per ricreare un oggetto dalla relativa immagine del supporto.

Il ricevitore di giornale identificato è quello più vecchio richiesto. Tutti gli altri ricevitori di giornale IBM MQ con date di creazione precedenti non sono più necessari. Se vengono visualizzate solo le stelle, è

necessario ripristinare i backup dalla data indicata per determinare quale è il ricevitore di giornale più vecchio.

Quando questi messaggi vengono registrati, IBM MQ scrive anche un oggetto spazio utente nella libreria del gestore code che contiene solo una voce: il nome del ricevitore di giornale più vecchio che deve essere conservato nel sistema. Questo spazio utente è denominato AMQJRNINF e i dati vengono scritti nel formato:

```
JJJJJJJJJJLLLLLLLLLLYYYYMMDDHHMMSSmmm
```

dove:

JJJJJJJJJJ
È il nome del destinatario meno recente di cui IBM MQ ha ancora bisogno.

LLLLLLLLLL
Indica il nome della libreria del ricevitore di giornale.

YYYY
Indica l'anno della voce di journal più vecchia necessaria a IBM MQ .

MM
È il mese della voce di journal più vecchia di cui IBM MQ ha bisogno.

DD
Indica il giorno della voce di journal più vecchia di cui IBM MQ ha bisogno.

HH
Indica l'ora della voce di journal meno recente necessaria a IBM MQ .

SS
Indica i secondi della voce di journal più vecchia di cui IBM MQ ha bisogno.

mmm
Indica i millisecondi della voce di journal più vecchia di cui IBM MQ ha bisogno.

Quando il ricevitore di giornale più vecchio è stato cancellato dal sistema, questo spazio utente contiene asterischi (*) per il nome del ricevitore di giornale.

Nota: L'esecuzione periodica di `RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) DSPJRNDTA(*YES)` consente di risparmiare tempo di avvio per IBM MQ e di ridurre il numero di ricevitori di giornale locali necessari per salvare e ripristinare per il ripristino.

IBM MQ for IBM i non fa riferimento ai ricevitori di giornale a meno che non stia eseguendo un passaggio di ripristino per l'avvio o per la ricreazione di un oggetto. Se rileva che un journal richiesto non è presente, emette il messaggio AMQ7432 alla coda messaggi del gestore code (QMQMMSG), riportando l'ora e la data della voce di journal richiesta per completare il passaggio di recupero.

In questo caso, ripristinare tutti i ricevitori di giornale che sono stati scollegati dopo questa data dal backup, per consentire la riuscita del passaggio di ripristino.

Mantenere il ricevitore di giornale che contiene la voce di avvio e i successivi ricevitori di giornale disponibili nella libreria del gestore code.

Mantenere il ricevitore di giornale contenente il più vecchio `Media Recovery Entrye` tutti i successivi ricevitori di giornale, sempre disponibili e presenti nella libreria del gestore code o di cui è stato eseguito il backup.

Quando si forza un punto di controllo:

- Se il ricevitore di giornale denominato in AMQ7460 non è avanzato, ciò indica che è necessario eseguire il commit o il rollback di un'unità di lavoro incompleta.
- Se il ricevitore di giornale denominato in AMQ7462 non è avanzato, ciò indica che sono presenti uno o più oggetti danneggiati.

IBM i *Ripristino di un gestore code completo (dati e journal) su IBM i*

Utilizzare queste informazioni per ripristinare uno o più gestori code da un backup o da una macchina remota.

Se è necessario ripristinare uno o più gestori code IBM MQ da un backup, attenersi alla seguente procedura.

1. Disattivare i gestori code IBM MQ .
2. Individuare la serie di backup più recente, costituita dal backup completo più recente e, successivamente, dai ricevitori di giornale.
3. Eseguire un'operazione RSTLIB, dal backup completo, per ripristinare le librerie di dati IBM MQ al loro stato al momento del backup completo, immettendo i seguenti comandi:

   ```
   RSTLIB LIB(QMQRLIB1) .........
   RSTLIB LIB(QMQRLIB2) .........
   ```

   Se un ricevitore di giornale è stato parzialmente salvato in un backup di giornale e completamente salvato in un backup successivo, ripristinare solo quello completamente salvato. Ripristinare i journal singolarmente, in ordine cronologico.
4. Eseguire un'operazione RST per ripristinare gli indirizzari IFS IBM MQ sul file system IFS, utilizzando il seguente comando:

   ```
   RST DEV(...) OBJ(('/QIBM/UserData/mqm/qmgrs/testqm')) ...
   ```
5. Avviare il gestore code messaggi. Ciò riproduce tutti i record del journal scritti a partire dal backup completo e ripristina tutti gli oggetti IBM MQ allo stato congruente al momento del backup del journal.

Se si desidera ripristinare un gestore code completo su una macchina diversa, utilizzare la seguente procedura per ripristinare tutto dalla libreria del gestore code. (Utilizziamo TEST come nome del gestore code di esempio.)

1. `CRTMQM TEST`
2. `DLTLIB LIB(QMTEST)`
3. `RSTLIB SAVLIB(QMTEST) DEV(*SAVF) SAVF(QMGRLIBSAV)`
4. Eliminare i seguenti file 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 *Ripristino dei ricevitori di giornale per un determinato gestore code su IBM i*

Utilizzare queste informazioni per comprendere i diversi modi per ripristinare i ricevitori di giornale.

L'azione più comune consiste nel ripristinare un ricevitore di giornale di cui è stato eseguito il backup in una libreria del gestore code, se un ricevitore che è stato rimosso è necessario di nuovo per una funzione di ripristino successiva.

Si tratta di un'attività semplice e richiede il ripristino dei ricevitori di giornale utilizzando il comando RSTOBJ IBM i standard:

```
RSTOBJ OBJ(QMQMDATA/AMQA000005) OBJTYPE(*JRNRCV) .........
```

Potrebbe essere necessario ripristinare una serie di ricevitori di giornale, piuttosto che un singolo ricevitore. Ad esempio AMQA000007 è il ricevitore più vecchio nelle librerie IBM MQ e sia AMQA000005 che AMQA000006 devono essere ripristinati.

In questo caso, ripristinare i ricevitori singolarmente in ordine cronologico inverso. Non è sempre necessario, ma è una buona pratica. In situazioni gravi, potrebbe essere necessario utilizzare il IBM i comando WRKJRNA per associare i ricevitori di giornale ripristinati al giornale.

Quando si ripristinano i giornali, il sistema crea automaticamente un ricevitore di giornale collegato con un nuovo nome nella sequenza del ricevitore di giornale. Tuttavia, il nuovo nome creato potrebbe essere lo stesso di un ricevitore di giornale che è necessario ripristinare. L'intervento manuale è necessario per risolvere questo problema; per creare un nuovo ricevitore di giornale in sequenza e un nuovo giornale prima di ripristinare il ricevitore di giornale.

Ad esempio, considerare il problema con il giornale salvato AMQAJRN e i seguenti ricevitori di giornale:

- AMQA000000
- AMQA100000
- AMQA200000
- AMQA300000
- AMQA400000
- AMQA500000
- AMQA600000
- AMQA700000
- AMQA800000
- AMQA900000

Quando si ripristina il journal AMQAJRN in una libreria del gestore code, il sistema crea automaticamente il ricevitore di journal AMQA000000. Questo ricevitore generato automaticamente è in conflitto con uno dei ricevitori di giornale esistenti (AMQA000000) che si desidera ripristinare e che non è possibile ripristinare.

La soluzione è:

1. Creare manualmente il ricevitore di giornale successivo (consultare “Journal del gestore code su IBM i” a pagina 404):

   ```
   CRTJRNRCV JRNRCV(QMQRLIB/AMQA900001) THRESHOLD(XXXXX)
   ```

2. Creare manualmente il giornale con il ricevitore di giornale:

   ```
   CRTJRN JRN(QMGRLIB/AMQAJRN) MNGRCV(*SYSTEM) +
   JRNRCV(QMGRLIB/AMQA9000001) MSGQ(QMGRLIB/AMQAJRNMSG)
   ```

3. Ripristinare i ricevitori di giornale locali AMQA000000 in AMQA900000.

IBM i Gestori code a più istanze su IBM i

I gestori code a più istanze migliorano la disponibilità passando automaticamente a un server standby se il server attivo ha esito negativo. I server attivo e standby sono più istanze dello stesso gestore code; condividono gli stessi dati del gestore code. Se l'istanza attiva non riesce, è necessario trasferire il proprio journal allo standby che assume il controllo in modo che il gestore code possa ricreare le proprie code.

Configurare i sistemi IBM i su cui sono in esecuzione gestori code a più istanze in modo che, se l'istanza del gestore code attiva ha esito negativo, il journal che sta utilizzando sia disponibile per l'istanza in standby che assume il controllo. È possibile progettare le proprie attività di configurazione e amministrazione per rendere il journal dall'istanza attiva disponibile per l'istanza che assume il controllo. Se non si desidera perdere i messaggi, la progettazione deve garantire che il giornale di standby sia congruente con il giornale attivo nel punto di errore. È possibile adattare la propria progettazione da una delle due configurazioni descritte con esempi negli argomenti successivi che mantengono la coerenza.

1. Eseguire il mirroring del journal dal sistema su cui è in esecuzione l'istanza del gestore code attivo ai sistemi su cui sono in esecuzione istanze in standby.
2. Collocare il giornale in un IASP (Independent Auxiliary Storage Pool) trasferibile dal sistema che esegue l'istanza attiva ad un'istanza in standby.

La prima soluzione non richiede hardware o software aggiuntivi poiché utilizza gli ASP di base. La seconda soluzione richiede IASP commutabili che necessitano del supporto cluster IBM i disponibile come IBM i License Product 5761-SS1 Opzione 41 a parte.

IBM i *Affidabilità e disponibilità su IBM i*

I gestori code a più istanze mirano a migliorare la disponibilità delle applicazioni. I vincoli fisici e tecnologici richiedono soluzioni diverse per rispondere alle esigenze di disaster recovery, backup dei gestori code e operazioni continue.

Nella configurazione per l'affidabilità e la disponibilità, vengono scambiati un gran numero di fattori, con il risultato di quattro punti di progettazione distinti:

Ripristino di emergenza
Ottimizzato per il ripristino dopo un grave disastro che distrugge tutti gli asset locali.

Il ripristino di emergenza su IBM i è spesso basato sul mirroring geografico dell'IASP.

Backup
Ottimizzato per il recupero dopo un guasto localizzato, di solito un errore umano o qualche problema tecnico imprevisto.

IBM MQ fornisce gestori code di backup per eseguire periodicamente il backup dei gestori code. È anche possibile utilizzare la replica asincrona dei journal del gestore code per migliorare la valuta del backup.

Disponibilità
Ottimizzato per ripristinare rapidamente le operazioni, offrendo l'aspetto di un servizio quasi ininterrotto in seguito a malfunzionamenti tecnici prevedibili, come un errore del server o del disco.

Il recupero è in genere misurato in minuti, con il rilevamento che a volte richiede più tempo del processo di ripristino. Un gestore code a più istanze ti assiste nella configurazione per la *disponibilità*.

Funzionamento continuo
Ottimizzato per fornire un servizio ininterrotto.

Le soluzioni di funzionamento continuo devono risolvere il problema di rilevamento e quasi sempre comportano la presentazione dello stesso lavoro attraverso più di un sistema e utilizzando il primo risultato, o se la correttezza è una considerazione importante, confrontando almeno due risultati.

Un gestore code a più istanze ti assiste nella configurazione per la *disponibilità*. Un'istanza del gestore code è attiva alla volta. Il passaggio a un'istanza standby richiede da poco più di dieci secondi a quindici minuti o più, a seconda di come il sistema è configurato, caricato e regolato.

Un gestore code a più istanze può fornire l'aspetto di un servizio quasi ininterrotto, se utilizzato con IBM MQ MQI clientsricollegabile, che è in grado di continuare l'elaborazione senza che il programma applicativo sia necessariamente a conoscenza di un'interruzione del gestore code; consultare l'argomento Riconnessione client automatizzata.

IBM i *Componenti di una soluzione alta disponibilità su IBM i*

Creare una soluzione di alta disponibilità utilizzando gestori code a più istanze fornendo una memoria di rete solida per i dati del gestore code, la replica del journal o la memoria IASP robusta per i journal del gestore code e utilizzando i client ricollegabili, delle applicazioni configurate come servizi del gestore code riavviabili.

Un gestore code a più istanze reagisce al rilevamento dell'errore del gestore code riavviando un'altra istanza del gestore code su un altro server. Per completare l'avvio, l'istanza deve accedere ai dati del gestore code condiviso nella memoria di rete e alla relativa copia del journal del gestore code locale.

Per creare una soluzione di alta disponibilità, è necessario gestire la disponibilità dei dati del gestore code, la valuta del journal del gestore code locale e creare applicazioni client riconnettibili oppure distribuire le proprie applicazioni come servizi del gestore code per riavviare automaticamente quando il gestore code viene ripristinato. La riconnessione automatica del client non è supportata da IBM MQ classes for Java.

Dati del gestore code

Posizionare i dati del gestore code su una memoria di rete condivisa, altamente disponibile e affidabile, possibilmente utilizzando dischi RAID di livello 1 o superiore. Il file system deve soddisfare i requisiti per un file system condiviso per i gestori code a più istanze; per ulteriori informazioni sui requisiti per i file system condivisi, vedere Requisiti per i file system condivisi. Network File System 4 (NFS4) è un protocollo che soddisfa questi requisiti.

Journal del gestore code

È inoltre necessario configurare i journal IBM i utilizzati dalle istanze del gestore code in modo che l'istanza in standby sia in grado di ripristinare i dati del gestore code in uno stato congruente. Per un servizio ininterrotto, ciò significa che è necessario ripristinare i journal al loro stato quando l'istanza attiva ha avuto esito negativo. A differenza delle soluzioni di backup o di ripristino di emergenza, il ripristino dei journal in un punto di controllo precedente non è sufficiente.

Non è possibile condividere fisicamente i journal tra più sistemi IBM i sulla memoria di rete. Per ripristinare i journal del gestore code allo stato congruente nel punto in cui si è verificato l'errore, è necessario trasferire il journal fisico locale all'istanza del gestore code attivo al momento dell'errore alla nuova istanza che è stata attivata oppure un mirror di gestione del journal sulle istanze in standby in esecuzione. Il giornale sottoposto a mirroring è una replica del giornale remoto che è stata mantenuta esattamente sincronizzata con il giornale locale appartenente all'istanza non riuscita.

Tre configurazioni sono i punti di partenza per la progettazione della modalità di gestione dei journal per un gestore code a più istanze,

1. Utilizzo della replica del giornale sincronizzata (mirroring del giornale) dall'ASP dell'istanza attiva agli ASP delle istanze in standby.
2. Trasferimento di un IASP configurato per conservare il journal del gestore code dall'istanza attiva all'istanza in standby che sta subendo il controllo come istanza attiva.
3. Utilizzo dei mirror IASP secondari sincronizzati.

Consultare le opzioni ASP , per ulteriori informazioni sull'inserimento dei dati del gestore code in un iASP, nel comando CRTMQM di IBM MQ IBM i .

Consultare anche Alta disponibilità nelle informazioni IBM i in IBM Documentation.

Applicazioni

Per creare un client per riconnettersi automaticamente al gestore code quando il gestore code in standby riprende la normale attività, connettere l'applicazione al gestore code utilizzando `MQCONNX` e specificare `MQCNO_RECONNECT_Q_MGR` nel campo **MQCNO** `Opzioni` . Consultare Programmi di esempio ad alta disponibilità per tre programmi di esempio che utilizzano client ricollegabili e Recupero applicazione per informazioni sulla progettazione di applicazioni client per il recupero.

IBM i *Creazione di una condivisione di rete per i dati del gestore code utilizzando NetServer su IBM i*

Creare una condivisione di rete su un server IBM i per memorizzare i dati del gestore code. Configurare le connessioni da due server, che ospiteranno le istanze del gestore code, per accedere alla condivisione di rete.

Prima di iniziare

- Per questa attività sono richiesti tre server IBM i . La condivisione di rete è definita su uno dei server, GAMMA. Gli altri due server, ALPHA e BETA, devono connettersi a GAMMA.
- Installare IBM MQ su tutti e tre i server.
- Installare System i Navigator; consultare System i Navigator.

Informazioni su questa attività

- Creare la directory del gestore code su GAMMA e impostare la proprietà e le autorizzazioni corrette per i profili utente `QMQM` e `QMQMADM`. La directory e le autorizzazioni sono facilmente create installando IBM MQ su GAMMA.
- Utilizzare System i Navigator per creare una condivisione nella directory dei dati del gestore code su GAMMA.
- Creare directory su ALPHA e BETA che puntino alla condivisione.

Procedura

1. In GAMMA, creare la directory per ospitare i dati del gestore code con il profilo utente `QMQM` come proprietario e `QMQMADM` come gruppo principale.

   Suggerimento:

   Un modo rapido e affidabile per creare la directory con i permessi corretti è installare IBM MQ su GAMMA.

   Successivamente, se non si desidera eseguire IBM MQ su GAMMA, disinstallare IBM MQ. Dopo la disinstallazione, la directory `/QIBM/UserData/mqm/qmgrs` rimane su GAMMA con il profilo utente `QMQM` del proprietario e `QMQMADM` il gruppo primario.

   L'attività utilizza la directory `/QIBM/UserData/mqm/qmgrs` su GAMMA per la condivisione.
2. Avviare la procedura guidata System i Navigator **Aggiungi connessione** e connettersi al sistema GAMMA.
   a) Fare doppio clic sull'icona **System i Navigator** sul desktop Windows .
   b) Fare clic su **Sì** per creare una connessione.
   c) Seguire le istruzioni nella procedura guidata **Aggiungi connessione** e creare una connessione dal sistema IBM i a GAMMA.

      La connessione a GAMMA viene aggiunta a **Connessioni**.
3. Aggiungere una nuova condivisione file su GAMMA.
   a) Nella finestra **System i Navigator** , fare clic sulla cartella `File Shares` in `My Connections/GAMMA/File Systems`.
   b) Nella finestra **Attività personali** , fare clic su **Gestisci condivisioni IBM i NetServer**.

      Una nuova finestra, **IBM i NetServer - GAMMA**, si apre sul desktop e mostra oggetti condivisi.
   c) Fare clic con il tasto destro del mouse sulla cartella `Shared Objects` > **file** > **Nuovo** > **File**.

      Viene visualizzata una nuova finestra, **IBM i NetServer File Share - GAMMA**.
   d) Fornire un nome alla condivisione, ad esempio `WMQ` .
   e) Impostare il controllo accessi su `Read/Write`.

f) Selezionare il **Nome percorso** passando alla directory /QIBM/UserData/mqm/qmgrs creata in precedenza e fare clic su **OK**.

La finestra **IBM i NetServer Condivisione file - GAMMA** viene chiusa e WMQ viene elencato nella finestra degli oggetti condivisi.

4. Fare clic con il pulsante destro del mouse su **WMQ** nella finestra degli oggetti condivisi. Fare clic su **File** > **Autorizzazioni**.

   Si apre una finestra, **Qmgrs Permissions - GAMMA**, per l'oggetto /QIBM/UserData/mqm/qmgrs.

   a) Controllare le seguenti autorizzazioni per QMQM, se non sono già impostate:

   ```
   Read
   Write
   Execute
   Management
   Existence
   Alter
   Reference
   ```

   b) Controllare le seguenti autorizzazioni per QMQMADM, se non sono già impostate:

   ```
   Read
   Write
   Execute
   Reference
   ```

   c) Aggiungere altri profili utente a cui si desidera concedere le autorizzazioni a /QIBM/UserData/mqm/qmgrs.

   Ad esempio, è possibile fornire le autorizzazioni predefinite del profilo utente (Pubblico) Read e Execute a /QIBM/UserData/mqm/qmgrs.

5. Verificare che tutti i profili utente a cui è stato concesso l'accesso a /QIBM/UserData/mqm/qmgrs su GAMMA abbiano la stessa parola d'ordine dei server che accedono a GAMMA.

   In particolare, assicurarsi che i profili utente QMQM su altri server, che accederanno alla condivisione, abbiano la stessa password del profilo utente QMQM su GAMMA.

   Suggerimento: Fare clic sulla cartella My Connections/GAMMA/Users and Groups in System i Navigator per impostare la password. In alternativa, utilizzare i comandi **CHFUSRPRF** e **CHGPWD** .

Risultati

Verificare di poter accedere a GAMMA da altri server utilizzando la condivisione. Se si stanno eseguendo altre attività, verificare di poter accedere a GAMMA da ALPHA e BETA utilizzando il percorso /QNTC/GAMMA/WMQ. Se la directory /QNTC/GAMMA non esiste in ALPHA o BETA, è necessario creare la directory. A seconda del dominio NetServer , potrebbe essere necessario eseguire l'IPL ALPHA o BETA prima di creare la directory.

```
CRTDIR DIR('/QNTC/GAMMA')
```

Una volta verificato che si ha accesso a /QNTC/GAMMA/WMQ da ALPHA o BETA, immettendo il comando, CRTMQM MQMNAME('QM1') MQMDIRP('/QNTC/GAMMA/WMQ') crea /QIBM/UserData/mqm/qmgrs/QM1 su GAMMA.

Operazioni successive

Creare un gestore code a più istanze effettuando le operazioni riportate nelle attività “Creazione di un gestore code a più istanze utilizzando il mirroring del journal e NetServer su IBM i” a pagina 430 o “Conversione di un gestore code a istanza singola in un gestore code a più istanze utilizzando NetServer e il mirroring del journal su IBM i” a pagina 434.

IBM i *Prestazioni di failover su IBM i*

Il tempo impiegato per rilevare un'istanza del gestore code ha avuto esito negativo e quindi per riprendere l'elaborazione su uno standby può variare da decine di secondi a quindici minuti o più a seconda della configurazione. Le prestazioni devono essere una considerazione importante nella progettazione e nel test di una soluzione alta disponibilità.

Vi sono vantaggi e svantaggi da valutare quando si decide se configurare un gestore code a più istanze per utilizzare la replica del journal o per utilizzare un IASP. Il mirroring richiede che il gestore code scriva in modo sincrono su un journal remoto. Da un punto di vista hardware, non è necessario che ciò influisca sulle prestazioni, ma da un punto di vista software, la scrittura su un giornale remoto comporta una lunghezza del percorso maggiore rispetto a quella su un giornale locale e ciò potrebbe ridurre in qualche misura le prestazioni di un gestore code in esecuzione. Tuttavia, quando il gestore code in standby assume il controllo, il ritardo nella sincronizzazione del relativo journal locale dal journal remoto gestito dall'istanza attiva prima che abbia esito negativo, è generalmente piccolo rispetto al tempo impiegato da IBM i per rilevare e trasferire l'IASP sul server che esegue l'istanza in standby del gestore code. I tempi di trasferimento IASP possono essere da dieci a quindici minuti invece di essere completati in secondi. Il tempo di trasferimento IASP dipende dal numero di oggetti che devono essere *attivati* quando l'IASP viene trasferito al sistema di standby e dalla dimensione dei percorsi di accesso o degli indici che devono essere uniti.

Quando il gestore code in standby assume il controllo, il ritardo nella sincronizzazione del relativo journal locale dal journal remoto gestito dall'istanza attiva prima che abbia esito negativo, è generalmente piccolo rispetto al tempo impiegato da IBM i per rilevare e trasferire l'ASP indipendente sul server che esegue l'istanza in standby del gestore code. I tempi di trasferimento ASP indipendente possono essere da dieci a quindici minuti invece di essere completati in secondi. Il tempo di trasferimento ASP indipendente dipende dal numero di oggetti che devono essere *attivati* quando l'ASP indipendente viene trasferito al sistema standby e dalla dimensione dei percorsi di accesso o degli indici che devono essere uniti.

Tuttavia, il trasferimento del giornale non è l'unico fattore che influenza il tempo impiegato per la ripresa completa dell'istanza in standby. È inoltre necessario considerare il tempo impiegato dal file system di rete per rilasciare il blocco sui dati del gestore code che segnala all'istanza in standby di continuare con l'avvio e anche il tempo impiegato per ripristinare le code dal journal, in modo che l'istanza sia in grado di avviare nuovamente l'elaborazione dei messaggi. Queste altre fonti di ritardo si aggiungono tutte al tempo necessario per avviare un'istanza in standby. Il tempo totale di commutazione è costituito dai seguenti componenti:

Tempo di rilevamento degli errori
: Il tempo impiegato da NFS per rilasciare il blocco sui dati del gestore code e l'istanza in standby per continuare il processo di avvio.

Tempo di trasferimento
: Nel caso di un cluster HA, il tempo impiegato da IBM i per il trasferimento dell'IASP dal sistema che ospita l'istanza attiva all'istanza in standby e, in caso di replica del giornale, il tempo impiegato per aggiornare il giornale locale nello standby con i dati dalla replica remota.

Ora di riavvio
: Il tempo impiegato dall'istanza del gestore code appena attiva per ricreare le code dall'ultimo punto di controllo nel journal ripristinato e per riprendere l'elaborazione dei messaggi.

  Nota:

  Se l'istanza in standby che ha assunto il controllo è configurata per la replica sincrona sull'istanza precedentemente attiva, l'avvio potrebbe essere ritardato. La nuova istanza attivata potrebbe non essere in grado di eseguire la replica sul giornale remoto, se il giornale remoto si trova sul server che ospitava l'istanza precedentemente attiva e il server ha riportato un errore.

  Il tempo di attesa predefinito per una risposta sincrona è un minuto. È possibile configurare il ritardo massimo prima del timeout della replica. In alternativa, è possibile configurare le istanze in standby per l'avvio utilizzando la replica asincrona per l'istanza attiva non riuscita. Successivamente si passa alla replica sincrona, quando l'istanza in errore è nuovamente in esecuzione in standby. La stessa considerazione si applica all'utilizzo di mirroring ASP indipendenti sincroni.

È possibile effettuare misurazioni di baseline separate per questi componenti per consentire di valutare il tempo complessivo per il failover e per determinare quale approccio di configurazione utilizzare. Nel prendere la migliore decisione di configurazione, è anche necessario considerare il modo in cui altre applicazioni sullo stesso server eseguiranno il failover e se vi sono processi di backup o di ripristino di emergenza che utilizzano già l'IASP.

I tempi di trasferimento IASP possono essere ridotti ottimizzando la configurazione del cluster:

1. I profili utente tra i sistemi nel cluster devono avere lo stesso GID e UID per eliminare la necessità per il processo di attivazione di modificare UID e GID.
2. Ridurre al minimo il numero di oggetti di database nel sistema e nei lotti dischi utente di base, poiché questi devono essere uniti per creare la tabella di riferimenti incrociati per il gruppo di lotti dischi.
3. Ulteriori suggerimenti sulle prestazioni sono disponibili in IBM Redbook, *Implementazione PowerHA per IBM i, SG24-7405*.

Una configurazione che utilizza gli ASP di base, il mirroring del giornale e una piccola configurazione deve essere commutata nell'ordine di decine di secondi.

IBM i *Panoramica sulla combinazione delle funzionalità di clustering IBM i con il clustering IBM MQ*

L'esecuzione di IBM MQ su IBM ie l'utilizzo delle funzioni di clustering di IBM i possono fornire una soluzione di alta disponibilità più completa, rispetto all'utilizzo del solo clustering IBM MQ .

Per disporre di questa funzione, è necessario impostare:

1. Cluster sulla tua macchina IBM i ; vedi “IBM iCluster” a pagina 420
2. Un IASP (independent auxiliary storage pool), in cui si sposta il gestore code; consultare “IASP (Independent Auxiliary Storage Pools)” a pagina 420
3. Un CRG (cluster resource group); consultare “Gruppi di risorse cluster unità” a pagina 421, in cui si definisce:
   - Dominio di ripristino
   - IASP
   - Programma di uscita; consultare “Programma di uscita CRG unità” a pagina 421

IBM iCluster

Un cluster IBM i è una raccolta di istanze, ossia computer o partizioni IBM i , che sono collegati logicamente tra loro.

Lo scopo di questo raggruppamento è consentire il backup di ogni istanza, eliminando un singolo punto di errore e aumentando la resilienza dei dati e delle applicazioni. Con un cluster creato, è possibile configurare i vari tipi di CRG (cluster resource group) per gestire applicazioni, dati e unità nel cluster.

Per ulteriori informazioni, consultare Creazione di un cluster e il comando CRTCLU (Creazione cluster) .

IASP (Independent Auxiliary Storage Pools)

Un IASP è un tipo di ASP utente che serve come estensione della memoria a livello singolo. Si tratta di un pezzo di memoria che, a causa della sua indipendenza dalla memoria di sistema, può essere facilmente manipolato senza dover eseguire l'IPL del sistema.

Un IASP può essere facilmente commutato in un'altra istanza del sistema operativo o replicato in un IASP di destinazione su un'altra istanza del sistema operativo. È possibile utilizzare due metodi per commutare un IASP tra le istanze:

- Il primo metodo richiede che tutti i computer nel cluster e la tower disco commutabile contenente l'IASP, siano collegati utilizzando un loop HSL (High Speed Link).
- Il secondo metodo richiede che le istanze del sistema operativo siano partizioni sullo stesso computer IBM i in cui gli IOP (input/output processor) possono essere commutati tra le partizioni. Non è

necessario alcun hardware speciale per poter replicare un IASP. La replica viene eseguita utilizzando TCP/IP sulla rete.

Per ulteriori informazioni, consultare il comando Configurazione ASP unità (CFGDEVASP) .

Gruppi di risorse cluster unità

Esistono diversi tipi di CRG (cluster resource group). Per ulteriori informazioni sui diversi tipi di CRG disponibili, consultare Gruppo di risorse cluster.

Questo argomento si concentra su un CRG unità. Un CRG unità:

- Descrive e gestisce le risorse delle unità come gli IASP (independent auxiliary storage pools).
- Definisce il dominio di ripristino dei nodi cluster
- Assegna un dispositivo e
- Assegna il programma di uscita che gestirà gli eventi cluster.

Il dominio di ripristino indica quale nodo cluster verrà considerato come nodo primario. Il resto dei nodi sono considerati backup. I nodi di backup vengono ordinati anche nel dominio di ripristino, specificando quale nodo è il primo backup, il secondo backup e così via, a seconda del numero di nodi presenti nel dominio di ripristino.

In caso di errore del nodo primario, il programma di uscita viene eseguito su tutti i nodi nel dominio di ripristino. Il programma di uscita in esecuzione sul primo backup può quindi eseguire le inizializzazioni necessarie per rendere questo nodo il nuovo nodo primario.

Consultare Creazione CRG unità e il comando Creazione CRG (Cluster Resource Group) per ulteriori informazioni.

Programma di uscita CRG unità

Il servizio di risorse cluster del sistema operativo richiama un programma di uscita CRG unità quando si verifica un evento in uno dei nodi definiti dal dominio di ripristino; ad esempio, un evento di failover o di commutazione.

Un evento di failover si verifica quando il nodo primario del cluster ha esito negativo e i CRG vengono commutati con tutte le risorse che gestiscono, mentre un evento di commutazione si verifica quando un CRG specifico viene commutato manualmente dal nodo primario al nodo di backup.

In entrambi i casi, il programma di uscita è responsabile dell'inizializzazione e dell'avvio di tutti i programmi in esecuzione sul precedente nodo primario, che converte il primo nodo di backup nel nuovo nodo primario.

Ad esempio, con IBM MQ, il programma di uscita deve essere responsabile dell'avvio del sottosistema IBM MQ (QMQM) e dei gestori code. I gestori code devono essere configurati per avviare automaticamente listener e servizi, ad esempio i controlli dei trigger.

Un programma di uscita di esempio, AMQSCRG4, è disponibile su IBM i.

Configurazione IASP commutabile

IBM MQ può essere impostato per sfruttare le funzionalità di clustering di IBM i. Per far ciò:

1. Creare un cluster IBM i tra i sistemi del data center
2. Spostare il gestore code in un IASP.

   “Spostamento o rimozione di un gestore code in o da un ASP (auxiliary storage pool) indipendente” a pagina 422 contiene del codice di esempio che consente di eseguire questa operazione.
3. È necessario creare un CRG definendo il dominio di ripristino, l'IASP e il programma di uscita.

   “Configurazione di un CRG (cluster resource group) unità” a pagina 422 contiene del codice di esempio che consente di eseguire questa operazione.

Concetti correlati

“ASP indipendenti e alta disponibilità” a pagina 443
Gli ASP indipendenti consentono alle applicazioni e ai dati di essere spostati tra i server. La flessibilità degli ASP indipendenti significa che sono la base per alcune soluzioni di alta disponibilità IBM i . Nel considerare se utilizzare un ASP o un ASP indipendente per il journal del gestore code, è necessario considerare un'altra configurazione ad alta disponibilità basata su ASP indipendenti.

IBM i *Configurazione di un CRG (cluster resource group) unità*

Un programma di esempio per impostare un CRG (Cluster Resource Group) unità.

Informazioni su questa attività

Nel seguente esempio, notare che:

- [PRIMARY SITE NAME] e [BACKUP SITE NAME] possono essere due stringhe distinte di otto caratteri o meno.
- [PRIMARY IP] e [BACKUP IP] sono gli IP da utilizzare per il mirroring.

Procedura

1. Identificare il nome del cluster.
2. Identificare il nome del programma di uscita CRG e la libreria.
3. Determinare il nome del nodo primario e i nodi backup che devono essere definiti da questo CRG.
4. Identificare l'IASP che deve essere gestito da questo CRG e assicurarsi che sia stato creato nel nodo primario.
5. Creare una descrizione unità nei nodi di backup utilizzando il seguente comando:

```
CRTDEVASP DEVD([IASP NAME]) RSRCNAME([IASP NAME])
```

6. Aggiungere l'indirizzo IP di acquisizione a tutti i nodi utilizzando il seguente comando:

```
ADDTCPIFC INTNETADR(' [TAKEOVER IP]') LIND([LINE DESC])
SUBNETMASK('[SUBNET MASK]') AUTOSTART(*NO)
```

7. Avviare l'indirizzo IP di acquisizione solo nel nodo primario utilizzando il comando:

```
STRTCPIFC INTNETADR('[TAKEOVER IP')
```

8. Opzionale: Se l'IASP è commutabile, richiamare il seguente comando:

```
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. Opzionale: Se il proprio IASP deve essere sottoposto a mirroring, richiamare questo comando:

```
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 *Spostamento o rimozione di un gestore code in o da un ASP (auxiliary storage pool) indipendente*

Un programma di esempio per spostare un gestore code in un IASP (independent auxiliary storage pool) e comandi per rimuovere un gestore code da uno IASP.

Informazioni su questa attività

Nel seguente esempio, notare che:

- [MANAGER NAME] è il nome del gestore code.
- [IASP NAME] è il nome dell'IASP.
- [MANAGER LIBRARY] è il nome della libreria del gestore code.
- [MANAGER DIRECTORY] è il nome della tua directory del gestore code.

Procedura

1. Identificare il nodo primario e i nodi di backup.
2. Eseguire la seguente procedura sul nodo primario:
   a) Assicurarsi che il gestore code sia terminato.
   b) Verificare che l'IASP sia `vary on` utilizzando il comando

```
VRYCFG CFGOBJ([IASP NAME]) CFGTYPE(*DEV) STATUS(*ON)
```

   c) Creare la directory dei gestori code nell'IASP.
   Ci sarà una directory sotto la root con il nome del tuo IASP, che è:

```
QSH CMD('mkdir -p /[IASP_NAME]/QIBM/UserData/mqm/qmgrs/')
```

   d) Spostare gli oggetti IFS del proprio gestore nella directory dei gestori code appena creati nell'IASP utilizzando il seguente comando:

```
QSH CMD('mv /QIBM/UserData/mqm/qmgrs/[MANAGER NAME]
/[IASP NAME]/QIBM/UserData/mqm/qmgrs')
```

   e) Creare un file di salvataggio temporaneo denominato `MGRLIB` utilizzando il comando:

```
CRTSAVF QGPL/MGRLIB
```

   f) Salvare la libreria del gestore code nel file di salvataggio `MGRLIB` utilizzando il seguente comando:

```
SAVLIB LIB([MANGER LIBRARY]) DEV(*SAVF) SAVF(QGPL/MGRLIB)
```

   g) Eliminare la libreria del gestore code utilizzando il seguente comando e ignorare tutti i messaggi di interrogazione:

```
DLTLIB [MANAGER LIBRARY]
```

   h) Ripristinare la libreria del gestore code nell'IASP utilizzando il seguente comando:

```
RSTLIB SAVLIB([MANAGER LIBRARY]) DEV(*SAVF) SAVF(QGPL/MGRLIB)
RSTASPDEV([IASP NAME])
```

   i) Eliminare il file di salvataggio temporaneo utilizzando il seguente comando:

```
 DLTF FILE(QGPL/MGRLIB)
```

   j) Creare un collegamento simbolico agli oggetti IFS del gestore code nell'IASP, utilizzando il seguente comando:

```
ADDLNK OBJ('/[IASP NAME]/QIBM/UserData/mqm/qmgrs/[MANAGER NAME]')
NEWLNK('/QIBM/UserData/mqm/qmgrs/[MANAGER NAME]')
```

   k) Collegarsi all'IASP utilizzando il seguente comando:

```
SETASPGRP [IASP NAME]
```

l) Avviare il gestore code utilizzando il seguente comando:

```
STRMQM [MANAGER NAME]
```

3. Eseguire la seguente procedura sul nodo o sui nodi di backup:
   a) Creare una directory temporanea del gestore code utilizzando il seguente comando:

```
QSH CMD('mkdir -p /[IASP NAME]/QIBM/UserData/mqm/qmgrs/[MANAGER NAME]')
```

   b) Creare un link simbolico alla directory temporanea del gestore code utilizzando il seguente comando:

```
ADDLNK OBJ('/[IASP NAME]/QIBM/UserData/mqm/qmgrs/[MANAGER NAME]')
NEWLNK('/QIBM/UserData/mqm/qmgrs/[MANAGER NAME]')
```

   c) Eliminare la directory temporanea utilizzando il comando seguente:

```
QSH CMD('rm -r /[IASP NAME]')
```

   d) Aggiungere quanto segue alla fine del file `/QIBM/UserData/mqm/mqs.ini`:

```
QueueManager:
Name=[MANAGER NAME]
Prefix=/QIBM/UserData/mqm
Library=[MANAGER LIBRARY]
Directory=[MANAGER DIRECTORY]
```

4. Per rimuovere un gestore code da un IASP, immettere i seguenti comandi:
   a) VRYCFG CFGOBJ ([IASP NAME]) CFGTYPE (*DEV) STATUS (*ON)
   b) SETASPGRP [NOME IASP]
   c) ENDMQM [NOME GESTORE]
   d) DLTMQM [NOME GESTORE]

IBM i *Configurazione giornale di mirroring per ASP su IBM i*

Configurare un gestore code a più istanze solido utilizzando la replica sincrona tra i journal con mirroring.

Una configurazione del gestore code con mirroring utilizza i journal creati in ASP (auxiliary storage pool) di base o indipendenti.

Su IBM i, i dati del gestore code vengono scritti nei journal e in un filesystem. I journal contengono la copia principale dei dati del gestore code. I giornali sono condivisi tra sistemi che utilizzano la replica del giornale sincrona o asincrona. Per riavviare un'istanza del gestore code è necessaria una combinazione di journal locali e remoti. Il riavvio del gestore code legge i record del journal dalla combinazione di journal locali e remoti sul server e i dati del gestore code sul file system di rete condiviso. I dati nel file system accelera il riavvio del gestore code. I punti di controllo vengono memorizzati nel file system, contrassegnando i punti di sincronizzazione tra il file system e i journal. I record del journal memorizzati prima del punto di controllo non sono richiesti per i tipici riavvii del gestore code. Tuttavia, i dati nel file system potrebbero non essere aggiornati e i record del journal dopo il punto di controllo vengono utilizzati per completare il riavvio del gestore code. I dati nei journal collegati all'istanza vengono mantenuti aggiornati in modo che il riavvio possa essere completato correttamente.

Ma anche i record del giornale potrebbero non essere aggiornati, se il giornale remoto sul server standby è stato replicato in modo asincrono e l'errore si è verificato prima della sincronizzazione. Nel caso in cui si decida di riavviare un gestore code utilizzando un journal remoto non sincronizzato, l'istanza del gestore code in standby potrebbe rielaborare i messaggi eliminati prima dell'errore dell'istanza attiva o non elaborare i messaggi ricevuti prima dell'errore dell'istanza attiva.

Un'altra, rara possibilità, è che il file system contenga il record del punto di controllo più recente, mentre un giornale remoto non sincronizzato sullo standby non lo fa. In tal caso, il gestore code non viene riavviato automaticamente. È possibile attendere che il journal remoto sia sincronizzato o avviare a freddo il gestore code standby dal file system. Anche se, in questo caso, il file system contiene un punto di

controllo più recente dei dati del gestore code rispetto al journal remoto, potrebbe non contenere tutti i messaggi elaborati prima dell'errore dell'istanza attiva. Alcuni messaggi potrebbero essere rielaborati e altri non elaborati, dopo un riavvio a freddo che non è sincronizzato con i giornali.

Con un gestore code a più istanze, il file system viene utilizzato anche per controllare quale istanza di un gestore code è attiva e quale è in standby. L'istanza attiva acquisisce un blocco sui dati del gestore code. Lo standby attende di acquisire il blocco e quando lo fa, diventa l'istanza attiva. Il blocco viene rilasciato dall'istanza attiva, se termina normalmente. Il blocco viene rilasciato dal file system se il file system rileva che l'istanza attiva ha avuto esito negativo o non può accedere al file system. Il file system deve soddisfare i requisiti per la rilevazione degli errori; consultare Requisiti dei file system condivisi.

L'architettura dei gestori code a più istanze su IBM i fornisce il riavvio automatico dopo un errore del server o del gestore code. Supporta inoltre il ripristino dei dati del gestore code a seguito di un errore del file system in cui sono memorizzati i dati del gestore code.

In Figura 24 a pagina 426, se ALPHA ha esito negativo, è possibile riavviare manualmente QM1 su BETA, utilizzando il giornale di mirroring. Aggiungendo la funzionalità del gestore code a più istanze a QM1, l'istanza in standby di QM1 riprende automaticamente su BETA se l'istanza attiva su ALPHA ha esito negativo. QM1 può anche riprendere automaticamente se è il server ALPHA che ha esito negativo, non solo l'istanza attiva di QM1. Una volta che BETA diventa l'host dell'istanza del gestore code attivo, l'istanza in standby può essere avviata su ALPHA.

Figura 24 a pagina 426 mostra una configurazione che esegue il mirroring dei journal tra due istanze di un gestore code utilizzando NetServer per memorizzare i dati del gestore code. È possibile espandere il modello per includere più journal e quindi più istanze. Seguire le regole di denominazione del giornale descritte nell'argomento, “Journal del gestore code su IBM i” a pagina 404. Attualmente il numero di istanze in esecuzione di un gestore code è limitato a due, una è attiva e una è in standby.

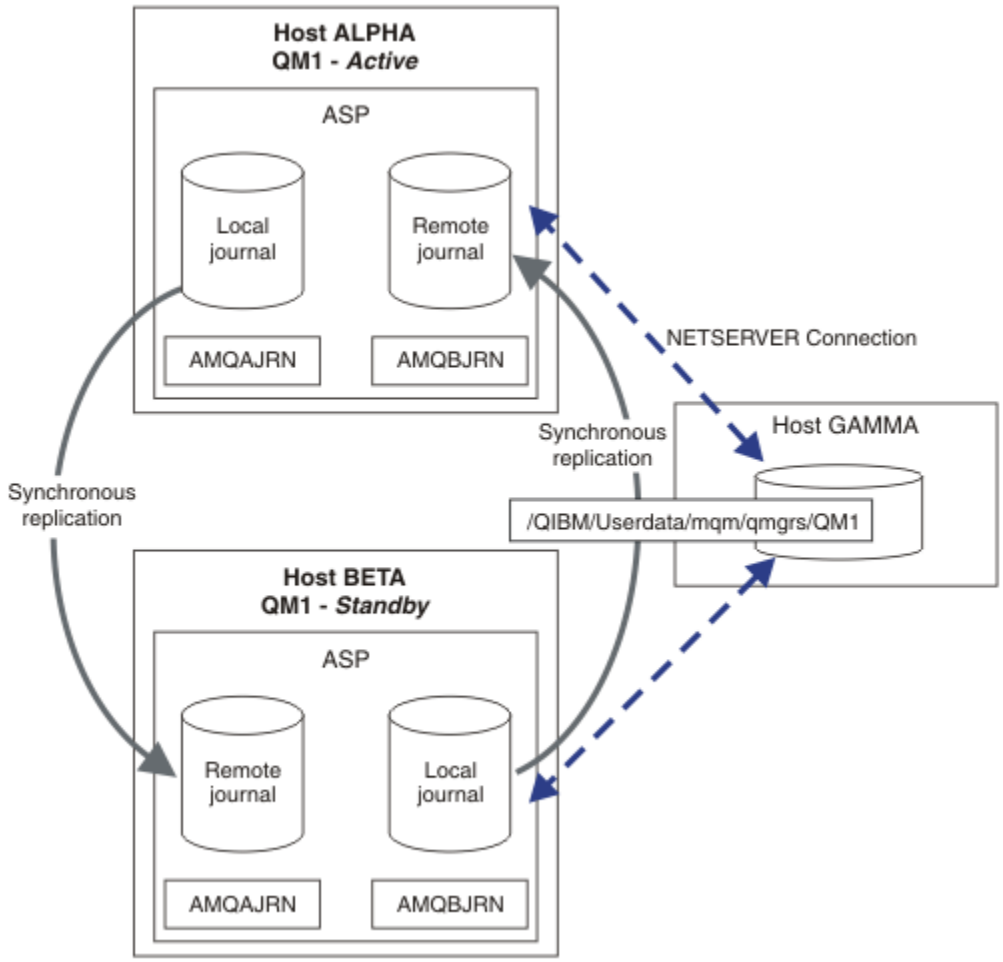


Figura 24. Eseguire il mirror di un journal del gestore code

Il giornale locale per QM1 sull'host ALPHA è denominato AMQAJRN (o più completamente, QMQM1/AMQAJRN) e su BETA il giornale è QMQM1/AMQBJRN. Ogni journal locale viene replicato sui journal remoti su tutte le altre istanze del gestore code. Se il gestore code è configurato con due istanze, un journal locale viene replicato su un journal remoto.

Replica del giornale remoto *SYNC o *ASYNC

I giornali IBM i vengono sottoposti a mirroring utilizzando uno dei due (*SYNC) o asincrono (*ASYNC) registrazione su giornale; consultare Gestione giornale remoto.

La modalità di replica in Figura 24 a pagina 426 è *SYNC, non *ASYNC. *ASYNC è più veloce, ma se si verifica un errore quando lo stato del giornale remoto è *ASYNCPEND, il giornale locale e quello remoto non sono congruenti. Il giornale remoto deve essere in linea con il giornale locale. Se si sceglie *SYNC, il sistema locale attende il giornale remoto prima di ritornare da una chiamata che richiede una scrittura completata. I giornali locali e remoti generalmente rimangono coerenti tra loro. Solo se l'operazione *SYNC impiega più tempo del tempo indicato[1], e la registrazione su giornale remota è disattivata, i giornali non sono sincronizzati. Un errore viene registrato nella coda messaggi di giornale e in QSYSOPR. Il gestore code rileva questo messaggio, scrive un errore nel log degli errori del gestore code e disattiva la replica remota del journal del gestore code. L'istanza del gestore code attiva viene ripristinata senza la registrazione del journal remoto su questo journal. Quando il server remoto è di nuovo disponibile, è necessario riattivare manualmente la replica del giornale remoto sincrona. I giornali vengono quindi risincronizzati.

[1] Il tempo designato è 60 secondi su IBM i 5 e nell'intervallo compreso tra 1 e 3600 secondi su IBM i 6.1 in poi.

Un problema con la configurazione *SYNC / *SYNC illustrata in Figura 24 a pagina 426 è il modo in cui l'istanza del gestore code in standby su BETA assume controllo. Non appena l'istanza del gestore code su BETA scrive il primo messaggio persistente, tenta di aggiornare il journal remoto su ALPHA. Se la causa del passaggio del controllo da ALPHA a BETA è stato il malfunzionamento di ALPHA e ALPHA è ancora inattivo, la registrazione su giornale remoto in ALPHA ha esito negativo. BETA attende la risposta ALPHA, quindi disattiva la registrazione su giornale remota e riprende l'elaborazione dei messaggi solo con la registrazione su giornale locale. BETA deve attendere un po' per rilevare che ALPHA è inattivo, causando un periodo di inattività.

La scelta tra l'impostazione della registrazione su giornale remota su *SYNC o *ASYNC è un trade - off. Tabella 24 a pagina 427 riepiloga i compromessi tra l'utilizzo di *SYNC e la registrazione su giornale *ASYNC tra una coppia di gestori code:

Tabella 24. Opzioni di registrazione su giornale remoto

Attivo	Standby	*SYNC	*ASYNC
***SYNC**		1. Commutazione e failover coerenti 2. L'istanza standby non viene ripresa immediatamente dopo il failover. 3. La registrazione su giornale remota deve essere sempre disponibile 4. Le prestazioni del gestore code dipendono dalla registrazione su giornale remota	1. Commutazione e failover coerenti 2. La registrazione su giornale remota deve essere commutata in *SYNC quando il server standby è disponibile 3. La registrazione su giornale remota deve rimanere disponibile dopo che è stata riavviata 4. Le prestazioni del gestore code dipendono dalla registrazione su giornale remota
***ASYNC**		1. Non è una combinazione sensata	1. Alcuni messaggi potrebbero essere persi o duplicati dopo un failover o una commutazione 2. Non è necessario che l'istanza in standby sia sempre disponibile perché l'istanza attiva possa continuare senza ritardi. 3. Le prestazioni non dipendono dalla registrazione su giornale remota

***SYNC / *SYNC**

L'istanza del gestore code attiva utilizza la registrazione su giornale *SYNC e quando l'istanza del gestore code in standby viene avviata, tenta immediatamente di utilizzare la registrazione su giornale *SYNC .

1. Il journal remoto è transazionalmente congruente con il journal locale del gestore code attivo. Se il gestore code è passato all'istanza in standby, può riprendere immediatamente. L'istanza in standby normalmente riprende senza alcuna perdita o duplicazione di messaggi. I messaggi vengono persi o duplicati solo se la registrazione su giornale remoto non è riuscita dall'ultimo checkpoint e il gestore code precedentemente attivo non può essere riavviato.
2. Se il gestore code esegue il failover sull'istanza in standby, potrebbe non essere in grado di avviarsi immediatamente. L'istanza del gestore code in standby è attivata con la registrazione su giornale *SYNC . La causa del failover potrebbe impedire la registrazione su giornale remota sul server che ospita l'istanza in standby. Il gestore code attende il rilevamento del problema prima di elaborare eventuali messaggi persistenti. Un errore viene registrato nella coda messaggi di giornale e in QSYSOPR. Il gestore code rileva questo messaggio, scrive un errore nel log degli errori del gestore code e disattiva la replica remota del journal del gestore code. L'istanza del gestore code attiva

viene ripristinata senza la registrazione del journal remoto su questo journal. Quando il server remoto è di nuovo disponibile, è necessario riattivare manualmente la replica del giornale remoto sincrona. I giornali vengono quindi risincronizzati.

3. Il server su cui viene duplicato il giornale remoto deve essere sempre disponibile per gestire il giornale remoto. Il journal remoto viene generalmente replicato sullo stesso server su cui è presente il gestore code in standby. Il server potrebbe diventare non disponibile. Un errore viene registrato nella coda messaggi di giornale e in QSYSOPR. Il gestore code rileva questo messaggio, scrive un errore nel log degli errori del gestore code e disattiva la replica remota del journal del gestore code. L'istanza del gestore code attiva viene ripristinata senza la registrazione del journal remoto su questo journal. Quando il server remoto è di nuovo disponibile, è necessario riattivare manualmente la replica del giornale remoto sincrona. I giornali vengono quindi risincronizzati.
4. La registrazione su giornale remota è più lenta della registrazione su giornale locale e sostanzialmente più lenta se i server sono separati da una grande distanza. Il gestore code deve attendere il journaling remoto, che riduce le prestazioni del gestore code.

La configurazione *SYNC / *SYNC tra una coppia di server ha lo svantaggio di un ritardo nella ripresa dell'istanza standby dopo il failover. La configurazione *SYNC / *ASYNC non ha questo problema.

*SYNC / *SYNC non garantisce alcuna perdita di messaggi dopo la commutazione o il failover, purché sia disponibile un giornale remoto. Se si desidera ridurre il rischio di perdita di messaggi dopo il failover o la commutazione, si hanno due scelte. Arrestare l'istanza attiva se il giornale remoto diventa inattivo oppure creare giornali remoti su più di un server.

***SYNC / *ASYNC**

L'istanza del gestore code attiva utilizza la registrazione su giornale *SYNC e quando viene avviata l'istanza del gestore code in standby, utilizza la registrazione su giornale *ASYNC . Poco dopo che il server su cui si trova la nuova istanza standby diventa disponibile, l'operatore di sistema deve commutare il giornale remoto sull'istanza attiva in *SYNC. Quando l'operatore commuta la registrazione su giornale remota da *ASYNC a *SYNC , l'istanza attiva si interrompe se lo stato del giornale remoto è *ASYNCPEND. L'istanza del gestore code attivo attende che le voci del journal rimanenti vengano trasferite al journal remoto. Quando il giornale remoto è stato sincronizzato con il giornale locale, il nuovo standby è di nuovo congruente con la nuova istanza attiva. Dal punto di vista della gestione dei gestori code a più istanze, in una configurazione *SYNC / *ASYNC l'operatore di sistema IBM i ha un'attività aggiuntiva. L'operatore deve commutare la registrazione su giornale remota in *SYNC oltre a riavviare l'istanza del gestore code in errore.

1. Il journal remoto è transazionalmente congruente con il journal locale del gestore code attivo. Se l'istanza del gestore code attivo viene commutata o se viene eseguito il failover sull'istanza in standby, l'istanza in standby può quindi riprendere immediatamente. L'istanza in standby normalmente riprende senza alcuna perdita o duplicazione di messaggi. I messaggi vengono persi o duplicati solo se la registrazione su giornale remoto non è riuscita dall'ultimo checkpoint e il gestore code precedentemente attivo non può essere riavviato.
2. L'operatore di sistema deve commutare il giornale remoto da *ASYNC a *SYNC poco dopo che il sistema che ospita l'istanza attiva diventa nuovamente disponibile. L'operatore potrebbe attendere il recupero del giornale remoto prima di passare il giornale remoto a *SYNC. In alternativa, l'operatore potrebbe commutare immediatamente l'istanza remota in *SYNC e forzare l'istanza attiva ad attendere che il giornale dell'istanza in standby si riattivi. Quando la registrazione su giornale remota è impostata su *SYNC, l'istanza in standby è generalmente transazionalmente congruente con l'istanza attiva. I messaggi vengono persi o duplicati solo se la registrazione su giornale remoto non è riuscita dall'ultimo checkpoint e il gestore code precedentemente attivo non può essere riavviato.
3. Quando la configurazione è stata ripristinata da una commutazione o da un failover, il server su cui risiede il giornale remoto deve essere sempre disponibile.

Scegliere *SYNC / *ASYNC quando si desidera che il gestore code in standby riprenda rapidamente dopo un failover. È necessario ripristinare manualmente l'impostazione del giornale remoto su *SYNC sulla nuova istanza attiva. La configurazione *SYNC / *ASYNC corrisponde al modello normale di gestione di una coppia di gestori code a più istanze. Dopo che un'istanza ha avuto esito negativo, c'è

un periodo di tempo prima che l'istanza standby venga riavviata, durante il quale l'istanza attiva non può eseguire il failover.

***ASYNC / *ASYNC**
Entrambi i server che ospitano i gestori code attivi e in standby sono configurati per utilizzare la registrazione su giornale remota *ASYNC .

1. Quando si verifica la commutazione o il failover, il gestore code continua con il journal sul nuovo server. Il giornale potrebbe non essere sincronizzato quando si verifica la commutazione o il failover. Di conseguenza, i messaggi potrebbero essere persi o duplicati.
2. L'istanza attiva viene eseguita, anche se il server che ospita il gestore code in standby non è disponibile. Il giornale locale viene replicato in modo asincrono con il server standby quando è disponibile.
3. Le prestazioni del gestore code locale non sono influenzate dalla registrazione su giornale remota.

Scegliere *ASYNC / *ASYNC se le prestazioni sono il requisito principale e si è pronti a perdere o duplicare alcuni messaggi dopo il failover o la commutazione.

***ASYNC / *SYNC**
Non esiste alcun motivo per utilizzare questa combinazione di opzioni.

Attivazione del gestore code da un journal remoto

I journal vengono replicati in modo sincrono o asincrono. Il giornale remoto potrebbe non essere attivo oppure potrebbe essere in linea con il giornale locale. Il giornale remoto potrebbe essere in fase di recupero, anche se è replicato in modo sincrono, perché potrebbe essere stato recentemente attivato. Le regole che il gestore code applica allo stato del journal remoto che utilizza durante l'avvio sono le seguenti.

1. L'avvio standby ha esito negativo se deve essere riprodotto dal giornale remoto sul giornale standby e lo stato del giornale è *FAILED o *INACTPEND.
2. Quando inizia l'attivazione dello standby, lo stato del giornale remoto sullo standby deve essere *ACTIVE o *INACTIVE. Se lo stato è *INACTIVE, è possibile che l'attivazione abbia esito negativo, se non tutti i dati del giornale sono stati replicati.

   L'errore si verifica se i dati del gestore code sul file system di rete hanno un record di checkpoint più recente rispetto a quello presente nel journal remoto. È improbabile che l'errore si verifichi, purché il giornale remoto sia attivato entro l'intervallo massimo predefinito di 30 minuti tra i punti di controllo. Se il gestore code in standby legge un record del punto di controllo più recente dal file system, non viene avviato.

   È possibile scegliere: attendere che il journal locale sul server attivo possa essere ripristinato oppure avviare a freddo il gestore code in standby. Se si sceglie l'avvio a freddo, il gestore code viene avviato senza dati journal e si basa sulla congruenza e sulla completezza dei dati del gestore code nel file system.

   Nota: Se si avvia a freddo un gestore code, si corre il rischio di perdere o duplicare i messaggi dopo l'ultimo checkpoint. Le transazioni del messaggio sono state scritte nel journal, ma alcune delle transazioni potrebbero non essere state scritte nei dati del gestore code nel file system. Quando si avvia a freddo un gestore code, viene avviato un nuovo journal e le transazioni non scritte sui dati del gestore code nel file system vengono perse.
3. L'attivazione del gestore code in standby attende che lo stato del giornale remoto sul database standby cambi da *ASYNCPEND o *SYNCPEND a *ASYNC o *SYNC. I messaggi vengono scritti periodicamente nella registrazione lavori del controller di esecuzione.

   Nota: In tal caso, l'attivazione è in attesa sul journal remoto locale per il gestore code standby che si sta attivando. Il gestore code attende anche un periodo di tempo prima di continuare senza un journal remoto. Attende quando tenta di scrivere in modo sincrono sul giornale remoto (o sui giornali) e il giornale non è disponibile.
4. L'attivazione si arresta se lo stato del giornale viene modificato in *FAILED o *INACTPEND.

I nomi e gli stati dei journal locali e remoti da utilizzare nell'attivazione vengono scritti nel log degli errori del gestore code.

IBM i *Creazione di un gestore code a più istanze utilizzando il mirroring del journal e NetServer su IBM i*

Creare un gestore code a più istanza da eseguire su due server IBM i . I dati del gestore code vengono memorizzati su un terzo server IBM i utilizzando NetServer. Il journal del gestore code viene sottoposto a mirroring tra i due server utilizzando il journal remoto. Il comando **ADDMQMJRN** viene utilizzato per semplificare la creazione dei giornali remoti.

Prima di iniziare

1. L'attività richiede tre server IBM i . Installare IBM MQ su due di essi, ALPHA e BETA nell'esempio. il prodotto deve essere almeno IBM WebSphere MQ 7.0.1 Fix Pack 1.
2. Il terzo server è un server IBM i , connesso da NetServer a ALPHA e BETA. Viene utilizzato per condividere i dati del gestore code. Non è necessario disporre di un'installazione IBM MQ . È utile installare IBM MQ sul server come passo temporaneo, per impostare le autorizzazioni e le directory del gestore code.
3. Verificare che il profilo utente `QMQM` abbia la stessa password su tutti e tre i server.
4. Installare IBM i NetServer; consultare i5/OS NetServer.

Informazioni su questa attività

Effettuare le seguenti operazioni per creare la configurazione mostrata in Figura 25 a pagina 433. I dati del gestore code sono connessi utilizzando IBM i NetServer.

- Creare le connessioni da ALPHA e BETA alla condivisione di directory su GAMMA che deve memorizzare i dati del gestore code. L'attività imposta anche le autorizzazioni, i profili utente e le parole d'ordine necessari.
- Aggiungere le voci del database relazionale (RDBE) ai sistemi IBM i che eseguiranno istanze del gestore code. Le voci RDBE sono utilizzate per collegarsi ai sistemi IBM i utilizzati per la registrazione su giornale remota.
- Creare il gestore code `QM1` sul server IBM i , ALPHA.
- Aggiungere le informazioni di controllo del gestore code per `QM1` sull'altro IBM i server, BETA.
- Creare journal remoti su entrambi i server IBM i per entrambe le istanze del gestore code. Ogni gestore code scrive nel journal locale. Il giornale locale viene replicato sul giornale remoto. Il comando, **ADDMQMJRN** semplifica l'aggiunta dei giornali e delle connessioni.
- Avviare il gestore code, consentendo un'istanza in standby.

Procedura

1. Eseguire l'attività, “Creazione di una condivisione di rete per i dati del gestore code utilizzando NetServer su IBM i” a pagina 417.

   Di conseguenza, ALPHA e BETA hanno una condivisione, `/QNTC/GAMMA/WMQ`, che punta a `/QIBM/UserData/mqm/qmgrs` su GAMMA. I profili utente `QMQM` e `QMQMADM` hanno le autorizzazioni necessarie e `QMQM` ha parole d'ordine corrispondenti su tutti e tre i sistemi.
2. Aggiungere le voci del database relazionale (RDBE) ai sistemi IBM i che ospiteranno le istanze del gestore code.
   a) Su ALPHA creare la connessione a BETA.

   ```
   ADDRDBDIRE RDB(BETA) RMTLOCNAME(BETA *IP) RMTAUTMTH(*USRIDPWD)
   ```

   b) Su BETA creare le connessioni a ALPHA.

```
ADDRDBDIRE RDB(ALPHA) RMTLOCNAME(ALPHA *IP) RMTAUTMTH(*USRIDPWD)
```

3. Creare il gestore code QM1 su ALPHA, salvando i dati del gestore code su GAMMA.

```
CRTMQM MQMNAME(QM1) UDLMSGQ(SYSTEM.DEAD.LETTER.QUEUE)
MQMDIRP(' /QNTC/GAMMA/WMQ ')
```

   Il percorso, utilizza NetServer per creare i dati del gestore code.

4. Esegui su ALPHA. Il comando aggiunge un giornale remoto su BETA per.

```
ADDMQMJRN MQMNAME(QM1) RMTJRNRDB(BETA)
```

   crea le voci di giornale nel relativo giornale locale su ALPHA quando l'istanza attiva è su ALPHA. Il giornale locale su ALPHA viene replicato sul giornale remoto su BETA.

5. Utilizzare il comando, per esaminare i dati di configurazione IBM MQ creati da ALPHA.

   Le informazioni sono necessarie nel passo successivo.

   In questo esempio, la seguente configurazione viene creata in ALPHA per:

```
Name=QM1
Prefix=/QIBM/UserData/mqm
Library=QMQM1
Directory=QM1
DataPath= /QNTC/GAMMA/WMQ /QM1
```

6. Creare un'istanza del gestore code di QM1 su BETA utilizzando il comando. Eseguire il seguente comando su BETA per modificare le informazioni di controllo del gestore code in BETA.

```
ADDMQMINF MQMNAME(QM1)
PREFIX('/QIBM/UserData/mqm')
MQMDIR(QM1)
MQMLIB(QMQM1)
DATAPATH(' /QNTC/GAMMA/WMQ /QM1 ')
```

   Suggerimento: Copiare e incollare le informazioni di configurazione. La stanza del gestore code è la stessa su ALPHA e BETA.

7. Esegui su BETA. Il comando aggiunge un giornale locale su BETA e un giornale remoto su ALPHA per.

```
ADDMQMJRN MQMNAME(QM1) RMTJRNRDB(ALPHA)
```

   crea voci di giornale nel relativo giornale locale su BETA quando l'istanza attiva è su BETA. Il giornale locale su BETA viene replicato sul giornale remoto su ALPHA.

   Nota: In alternativa, è possibile impostare la registrazione su giornale remota da BETA a ALPHA utilizzando la registrazione su giornale asincrona.

   Utilizzare questo comando per impostare la registrazione su giornale asincrona da BETA a ALPHA, invece del comando nel passo “7” a pagina 431.

```
ADDMQMJRN MQMNAME (QM1) RMTJRNRDB (ALPHA) RMTJRNDLV (*ASYNC)
```

   Se il server o la registrazione su giornale su ALPHA è l'origine dell'errore, BETA viene avviato senza attendere che le nuove voci di giornale vengano replicate su ALPHA.

   Passare la modalità di replica a *SYNC, utilizzando il comando, quando ALPHA è di nuovo in linea.

   Utilizzare le informazioni contenute in “Configurazione giornale di mirroring per ASP su IBM i” a pagina 424 per decidere se eseguire il mirroring dei giornali in modo sincrono, asincrono o una combinazione

di entrambi. Il valore predefinito è quello di replicare in modo sincrono, con un periodo di attesa di 60 secondi per una risposta dal giornale remoto.

8. Verificare che i giornali su ALPHA e BETA siano abilitati e che lo stato della replica del giornale remoto sia abilitato.
   a) Su ALPHA:

```
WRKMQMJRN MQMNAME(QM1)
```

   b) Su BETA:

```
WRKMQMJRN MQMNAME(QM1)
```

9. Avviare le istanze del gestore code su ALPHA e BETA.
   a) Avviare la prima istanza su ALPHA, rendendola l'istanza attiva. Abilitazione del passaggio a un'istanza standby.

```
STRMQM MQMNAME(QM1) STANDBY(*YES)
```

   b) Avviare la seconda istanza su BETA, rendendola l'istanza standby.

```
STRMQM MQMNAME(QM1) STANDBY(*YES)
```

Risultati

Utilizzare per controllare lo stato del gestore code:

1. Lo stato dell'istanza del gestore code su ALPHA deve essere.
2. Lo stato dell'istanza del gestore code su BETA dovrebbe essere.

Esempio

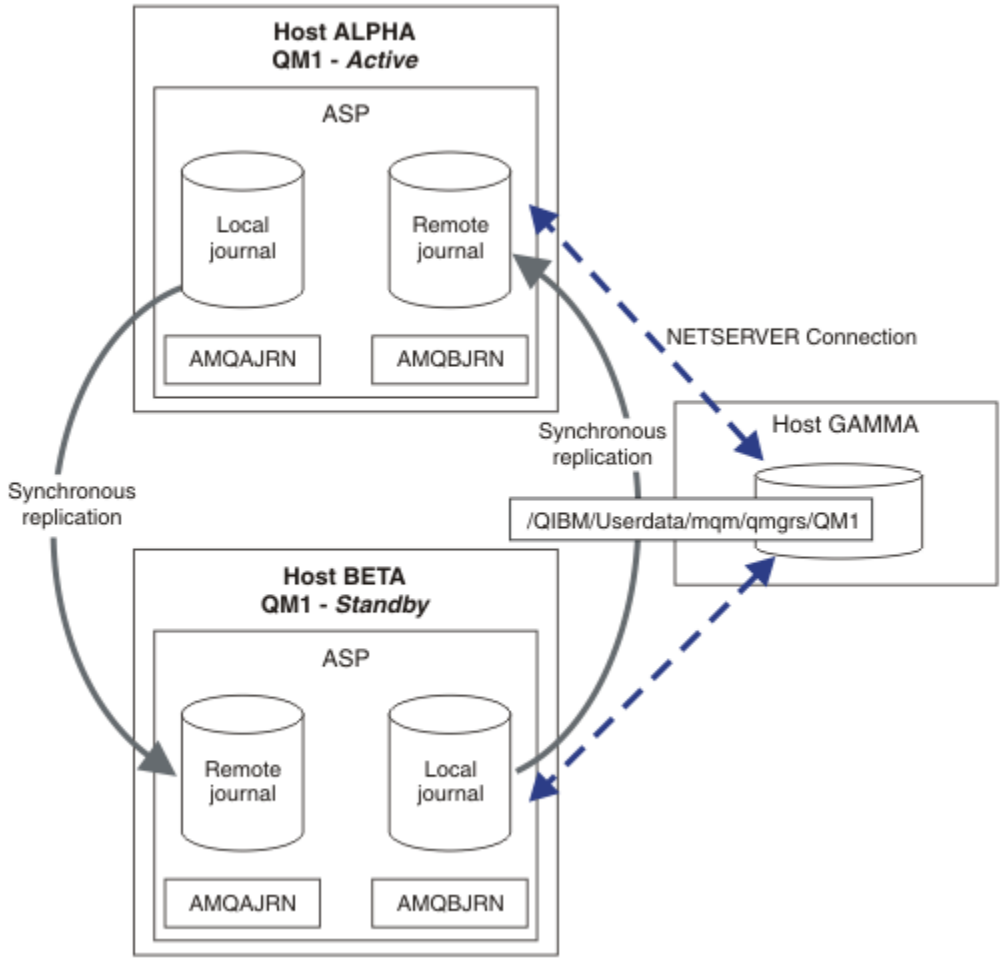


Figura 25. Configurazione giornale con mirroring

Operazioni successive

- Verificare che le istanze attive e in standby si spostino automaticamente. È possibile eseguire i programmi di esempio di alta disponibilità di esempio per verificare il passaggio; consultare Programmi di esempio di alta disponibilità. I programmi di esempio sono client ' C'. È possibile eseguirli da una piattaforma Windows o Unix.
  1. Avviare i programmi di esempio ad alta disponibilità.
  2. Su ALPHA, terminare il gestore code che richiede il passaggio su:

     ```
     ENDMQM MQMNAME(QM1) OPTION(*IMMED) ALSWITCH(*YES)
     ```

  3. Verificare che l'istanza su BETA sia attiva.
  4. Riavvia su ALPHA

     ```
     STRMQM MQMNAME(QM1) STANDBY(*YES)
     ```

- Esaminare le configurazioni alternative di alta disponibilità:
  1. Utilizzare NetServer per inserire i dati del gestore code su un server Windows .

2. Invece di utilizzare la registrazione su giornale remota per eseguire il mirroring del journal del gestore code, memorizzare il journal su un ASP indipendente. Utilizzare il cluster IBM i per trasferire l'ASP indipendente da ALPHA a BETA.

IBM i *Conversione di un gestore code a istanza singola in un gestore code a più istanze utilizzando NetServer e il mirroring del journal su IBM i*

Convertire un gestore code a istanza singola in gestore code a più istanze. Spostare i dati del gestore code in una condivisione di rete connessa da NetServer. Eseguire il mirroring del journal del gestore code su un secondo server IBM i utilizzando il journal remoto.

Prima di iniziare

1. L'attività richiede tre server IBM i . L'installazione di IBM MQ esistente, sul server ALPHA nell'esempio, deve essere almeno IBM WebSphere MQ 7.0.1 Fix Pack 1. ALPHA sta eseguendo un gestore code denominato QM1 nell'esempio.
2. Installare IBM MQ sul secondo server IBM i , BETA nell'esempio.
3. Il terzo server è un server IBM i , connesso da NetServer a ALPHA e BETA. Viene utilizzato per condividere i dati del gestore code. Non è necessario disporre di un'installazione IBM MQ . È utile installare IBM MQ sul server come passo temporaneo, per impostare le autorizzazioni e le directory del gestore code.
4. Verificare che il profilo utente QMQM abbia la stessa password su tutti e tre i server.
5. Installare IBM i NetServer; consultare i5/OS NetServer.

Informazioni su questa attività

Effettuare le seguenti operazioni per convertire un gestore code a istanza singola nel gestore code a più istanze mostrato in Figura 26 a pagina 438. Il gestore code a istanza singola viene eliminato nell'attività e quindi ricreato, memorizzando i dati del gestore code sulla condivisione di rete connessa da NetServer. Questa procedura è più affidabile dello spostamento delle directory e dei file del gestore code nella condivisione di rete utilizzando il comando **CPY** .

- Creare le connessioni da ALPHA e BETA alla condivisione di directory su GAMMA che deve memorizzare i dati del gestore code. L'attività imposta anche le autorizzazioni, i profili utente e le parole d'ordine necessari.
- Aggiungere le voci del database relazionale (RDBE) ai sistemi IBM i che eseguiranno istanze del gestore code. Le voci RDBE sono utilizzate per collegarsi ai sistemi IBM i utilizzati per la registrazione su giornale remota.
- Salvare le definizioni e i log del gestore code, arrestare il gestore code ed eliminarlo.
- Ricreare il gestore code, memorizzando i dati del gestore code sulla condivisione di rete su GAMMA.
- Aggiungere la seconda istanza del gestore code all'altro server.
- Creare journal remoti su entrambi i server IBM i per entrambe le istanze del gestore code. Ogni gestore code scrive nel journal locale. Il giornale locale viene replicato sul giornale remoto. Il comando, **ADDMQMJRN** semplifica l'aggiunta dei giornali e delle connessioni.
- Avviare il gestore code, consentendo un'istanza in standby.

Nota:

Nel passo “4” a pagina 435 dell'attività, si elimina il gestore code a istanza singola, QM1. L'eliminazione del gestore code elimina tutti i messaggi persistenti sulle code. Per questo motivo, completare l'elaborazione di tutti i messaggi memorizzati dal gestore code prima di convertire il gestore code. Se non è possibile elaborare tutti i messaggi, eseguire il backup della libreria del gestore code prima del passo “4” a pagina 435. Ripristinare la libreria del gestore code dopo il passo “5” a pagina 435.

Nota:

Nel passo “5” a pagina 435 dell'attività, creare nuovamente QM1. Anche se il gestore code ha lo stesso nome, ha un identificativo del gestore code diverso. Il cluster del gestore code utilizza l'identificativo

del gestore code. Per eliminare e creare nuovamente un gestore code in un cluster, è necessario prima rimuovere il gestore code dal cluster; fare riferimento a Rimozione di un gestore code da un cluster: metodo alternativo o Rimozione di un gestore code da un cluster. Una volta ricreato il gestore code, aggiungerlo al cluster. Anche se ha lo stesso nome di prima, sembra essere un nuovo gestore code per gli altri gestori code nel cluster.

Procedura

1. Eseguire l'attività, "Creazione di una condivisione di rete per i dati del gestore code utilizzando NetServer su IBM i" a pagina 417.

   Di conseguenza, ALPHA e BETA hanno una condivisione, `/QNTC/GAMMA/WMQ`, che punta a `/QIBM/UserData/mqm/qmgrs` su GAMMA. I profili utente `QMQM` e `QMQMADM` hanno le autorizzazioni necessarie e `QMQM` ha parole d'ordine corrispondenti su tutti e tre i sistemi.
2. Aggiungere le voci del database relazionale (RDBE) ai sistemi IBM i che ospiteranno le istanze del gestore code.
   a) Su ALPHA creare la connessione a BETA.

   ```
   ADDRDBDIRE RDB(BETA) RMTLOCNAME(BETA *IP) RMTAUTMTH(*USRIDPWD)
   ```

   b) Su BETA creare le connessioni a ALPHA.

   ```
   ADDRDBDIRE RDB(ALPHA) RMTLOCNAME(ALPHA *IP) RMTAUTMTH(*USRIDPWD)
   ```
3. Creare gli script che ricreano gli oggetti gestore code.

   ```
   QSAVEQMGR LCLQMGRNAM(QM1) FILENAME('*CURLIB/QMQSC(QM1)')
   OUTPUT(*REPLACE) MAKEAUTH(*YES) AUTHFN('*CURLIB/QMAUT(QM1)')
   ```
4. Arrestare il gestore code ed eliminarlo.

   ```
   ENDMQM MQMNAME(QM1) OPTION(*IMMED) ENDCCTJOB(*YES) RCDMQMIMG(*YES) TIMEOUT(15)
   DLTMQM MQMNAME(QM1)
   ```
5. Creare il gestore code `QM1` su ALPHA, salvando i dati del gestore code su GAMMA.

   ```
   CRTMQM MQMNAME(QM1) UDLMSGQ(SYSTEM.DEAD.LETTER.QUEUE)
   MQMDIRP(' /QNTC/GAMMA/WMQ ')
   ```

   Il percorso, utilizza NetServer per creare i dati del gestore code.
6. Ricreare gli oggetti del gestore code per `QM1` dalle definizioni salvate.

   ```
   STRMQMMQSC SRCMBR(QM1) SRCFILE(*CURLIB/QMQSC) MQMNAME(QM1)
   ```
7. Applicare le autorizzazioni dalle informazioni salvate.
   a) Compilare il programma di autorizzazione salvato.

   ```
   CRTCLPGM PGM(*CURLIB/QM1) SRCFILE(*CURLIB/QMAUT)
   SRCMBR(QM1) REPLACE(*YES)
   ```

   b) Eseguire il programma per applicare le autorizzazioni.

   ```
   CALL PGM(*CURLIB/QM1)
   ```

   c) Aggiornare le informazioni di sicurezza per `QM1`.

```
RFRMQMAUT MQMNAME(QM1)
```

8. Esegui su ALPHA. Il comando aggiunge un giornale remoto su BETA per.

```
ADDMQMJRN MQMNAME(QM1) RMTJRNRDB(BETA)
```

   crea le voci di giornale nel relativo giornale locale su ALPHA quando l'istanza attiva è su ALPHA. Il giornale locale su ALPHA viene replicato sul giornale remoto su BETA.

9. Utilizzare il comando, per esaminare i dati di configurazione IBM MQ creati da ALPHA.

   Le informazioni sono necessarie nel passo successivo.

   In questo esempio, la seguente configurazione viene creata in ALPHA per:

```
Name=QM1
Prefix=/QIBM/UserData/mqm
Library=QMQM1
Directory=QM1
DataPath= /QNTC/GAMMA/WMQ /QM1
```

10. Creare un'istanza del gestore code di QM1 su BETA utilizzando il comando. Eseguire il seguente comando su BETA per modificare le informazioni di controllo del gestore code in BETA.

```
ADDMQMINF MQMNAME(QM1)
PREFIX('/QIBM/UserData/mqm')
MQMDIR(QM1)
MQMLIB(QMQM1)
DATAPATH(' /QNTC/GAMMA/WMQ /QM1 ')
```

    Suggerimento: Copiare e incollare le informazioni di configurazione. La stanza del gestore code è la stessa su ALPHA e BETA.

11. Esegui su BETA. Il comando aggiunge un giornale locale su BETA e un giornale remoto su ALPHA per.

```
ADDMQMJRN MQMNAME(QM1) RMTJRNRDB(ALPHA)
```

    crea voci di giornale nel relativo giornale locale su BETA quando l'istanza attiva è su BETA. Il giornale locale su BETA viene replicato sul giornale remoto su ALPHA.

    Nota: In alternativa, è possibile impostare la registrazione su giornale remota da BETA a ALPHA utilizzando la registrazione su giornale asincrona.

    Utilizzare questo comando per impostare la registrazione su giornale asincrona da BETA a ALPHA, invece del comando nel passo “7” a pagina 431.

```
ADDMQMJRN MQMNAME (QM1) RMTJRNRDB (ALPHA) RMTJRNDLV (*ASYNC)
```

    Se il server o la registrazione su giornale su ALPHA è l'origine dell'errore, BETA viene avviato senza attendere che le nuove voci di giornale vengano replicate su ALPHA.

    Passare la modalità di replica a *SYNC, utilizzando il comando, quando ALPHA è di nuovo in linea.

    Utilizzare le informazioni contenute in “Configurazione giornale di mirroring per ASP su IBM i” a pagina 424 per decidere se eseguire il mirroring dei giornali in modo sincrono, asincrono o una combinazione di entrambi. Il valore predefinito è quello di replicare in modo sincrono, con un periodo di attesa di 60 secondi per una risposta dal giornale remoto.

12. Verificare che i giornali su ALPHA e BETA siano abilitati e che lo stato della replica del giornale remoto sia abilitato.

    a) Su ALPHA:

```
WRKMQMJRN MQMNAME(QM1)
```

b) Su BETA:

```
WRKMQMJRN MQMNAME(QM1)
```

13. Avviare le istanze del gestore code su ALPHA e BETA.
    a) Avviare la prima istanza su ALPHA, rendendola l'istanza attiva. Abilitazione del passaggio a un'istanza standby.

```
STRMQM MQMNAME(QM1) STANDBY(*YES)
```

    b) Avviare la seconda istanza su BETA, rendendola l'istanza standby.

```
STRMQM MQMNAME(QM1) STANDBY(*YES)
```

Risultati

Utilizzare per controllare lo stato del gestore code:

1. Lo stato dell'istanza del gestore code su ALPHA deve essere.
2. Lo stato dell'istanza del gestore code su BETA dovrebbe essere.

Esempio

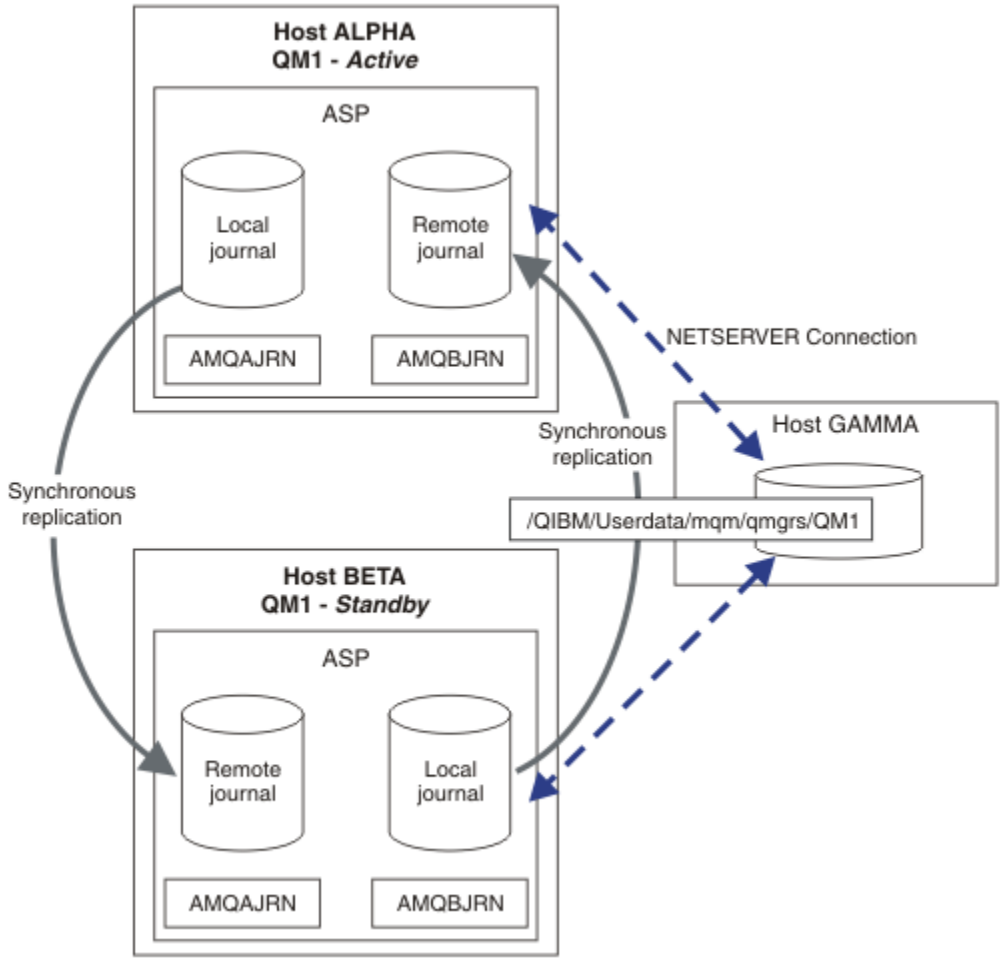


Figura 26. Configurazione giornale con mirroring

Operazioni successive

- Verificare che le istanze attive e in standby si spostino automaticamente. È possibile eseguire i programmi di esempio di alta disponibilità di esempio per verificare il passaggio; consultare Programmi di esempio di alta disponibilità. I programmi di esempio sono client ' C'. È possibile eseguirli da una piattaforma Windows o Unix.
  1. Avviare i programmi di esempio ad alta disponibilità.
  2. Su ALPHA, terminare il gestore code che richiede il passaggio su:

     ```
     ENDMQM MQMNAME(QM1) OPTION(*IMMED) ALSWITCH(*YES)
     ```

  3. Verificare che l'istanza su BETA sia attiva.
  4. Riavvia su ALPHA

     ```
     STRMQM MQMNAME(QM1) STANDBY(*YES)
     ```

- Esaminare le configurazioni alternative di alta disponibilità:
  1. Utilizzare NetServer per inserire i dati del gestore code su un server Windows .

2. Invece di utilizzare la registrazione su giornale remota per eseguire il mirroring del journal del gestore code, memorizzare il journal su un ASP indipendente. Utilizzare il cluster IBM i per trasferire l'ASP indipendente da ALPHA a BETA.

IBM i *Configurazione giornale ASP indipendente commutata su IBM i*

Non è necessario replicare un journal ASP indipendente per creare una configurazione del gestore code a più istanze. È necessario automatizzare un mezzo per trasferire l'ASP indipendente dal gestore code attivo al gestore code in standby. Esistono soluzioni alternative di alta disponibilità possibili utilizzando un ASP indipendente, non tutte richiedono l'utilizzo di un gestore code a più istanze.

Quando si utilizza un ASP indipendente non è necessario eseguire il mirroring del journal del gestore code. Se è stata installata la gestione del cluster e i server che ospitano le istanze del gestore code si trovano nello stesso gruppo di risorse cluster, il journal del gestore code può essere trasferito automaticamente su un altro server entro una breve distanza dal server attivo, se l'host che esegue l'istanza attiva ha esito negativo. È anche possibile trasferire il giornale manualmente, come parte di uno switch pianificato, oppure è possibile scrivere una procedura di comando per trasferire l'ASP indipendente in modo programmatico.

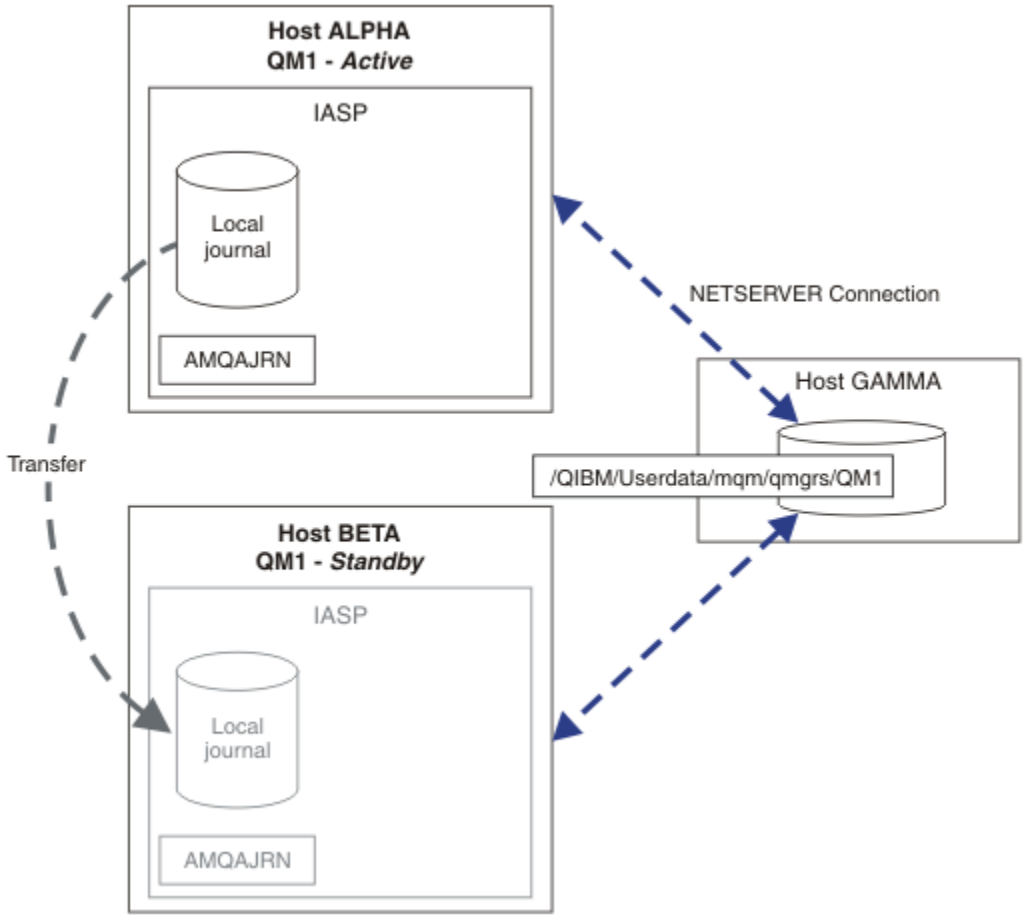


Figura 27. Trasferire un journal del gestore code utilizzando un ASP indipendente

Per l'operazione del gestore code a più istanze, i dati del gestore code devono essere memorizzati su un file system condiviso. Il file system può essere ospitato su una varietà di piattaforme differenti. Non è possibile memorizzare i dati del gestore code a più istanze su un ASP o un ASP indipendente.

Il file system condiviso esegue due ruoli nella configurazione: gli stessi dati del gestore code sono condivisi tra tutte le istanze del gestore code. Il file system deve avere un robusto protocollo di blocco che garantisce che una sola istanza del gestore code abbia accesso ai dati del gestore code una volta avviato. Se il gestore code ha esito negativo o le comunicazioni con il file server si interrompono, il file system

deve rilasciare il blocco sui dati del gestore code conservati dall'istanza attiva che non comunica più con il file system. L'istanza del gestore code in standby può ottenere l'accesso in lettura / scrittura ai dati del gestore code. Il protocollo del file system deve essere conforme a una serie di regole per funzionare correttamente con gestori code a più istanze; consultare “Componenti di una soluzione alta disponibilità su IBM i” a pagina 416.

Il meccanismo di blocco serializza il comando di avvio del gestore code e controlla quale istanza del gestore code è attiva. Una volta che un gestore code diventa attivo, ricrea le proprie code dal journal locale che l'utente, o il cluster HA, ha trasferito al server standby. I client ricollegabili in attesa di riconnessione allo stesso gestore code vengono riconnessi e viene eseguito il backout di tutte le transazioni in corso. Le applicazioni configurate per essere avviate come servizi del gestore code vengono avviate.

È necessario verificare che il journal locale dell'istanza del gestore code attivo non riuscita sull'ASP indipendente venga trasferito al server su cui è presente l'istanza del gestore code in standby appena attivata, configurando il gestore risorse cluster o trasferendo manualmente l'ASP indipendente. L'utilizzo di ASP indipendenti non preclude la configurazione dei giornali remoti e del mirroring, se si decide di utilizzare l'ASP indipendente per il backup e il ripristino di emergenza e utilizzare il mirroring del journal remoto per la configurazione del gestore code a più istanze.

Se si è scelto di utilizzare un ASP indipendente, è possibile considerare configurazioni alternative ad alta disponibilità. Lo sfondo di queste soluzioni è descritto in “ASP indipendenti e alta disponibilità” a pagina 443.

1. Invece di utilizzare i gestori code a più istanze, installare e configurare un gestore code a istanza singola interamente su un ASP indipendente e utilizzare i servizi di alta disponibilità IBM i per eseguire il failover del gestore code. È probabilmente necessario aumentare la soluzione con un monitor del gestore code per rilevare se il gestore code ha avuto esito negativo indipendentemente dal server. Questa è la base della soluzione fornita in *Supportpac MC41: Configurazione di IBM MQ per iSeries per l'alta disponibilità*.
2. Utilizzare gli ASP indipendenti e XSM (cross site mirroring) per eseguire il mirroring dell'ASP indipendente piuttosto che commutare l'ASP indipendente sul bus locale. Ciò estende l'intervallo geografico della soluzione ASP indipendente fino a quando il tempo impiegato per scrivere i record di registrazione su una lunga distanza lo consente.

IBM i *Creazione di un gestore code a più istanze utilizzando un ASP indipendente e NetServer su IBM i*

Creare un gestore code a più istanza da eseguire su due server IBM i . I dati del gestore code vengono memorizzati su un server IBM i utilizzando NetServer. Il journal del gestore code è memorizzato su un ASP indipendente. Utilizzare il cluster IBM i o una procedura manuale per trasferire l'ASP indipendente contenente il journal del gestore code sull'altro server IBM i .

Prima di iniziare

1. L'attività richiede tre server IBM i . Installare IBM MQ su due di essi, ALPHA e BETA nell'esempio. il prodotto deve essere almeno IBM WebSphere MQ 7.0.1 Fix Pack 1.
2. Il terzo server è un server IBM i , connesso da NetServer a ALPHA e BETA. Viene utilizzato per condividere i dati del gestore code. Non è necessario disporre di un'installazione IBM MQ . È utile installare IBM MQ sul server come passo temporaneo, per impostare le autorizzazioni e le directory del gestore code.
3. Verificare che il profilo utente `QMQM` abbia la stessa password su tutti e tre i server.
4. Installare IBM i NetServer; consultare i5/OS NetServer.
5. Creare le procedure per trasferire l'ASP indipendente dal gestore code in errore al gestore code in standby che sta subendo il controllo. È possibile che alcune delle tecniche contenute in *SupportPac MC41: Configurazione di IBM MQ per iSeries per l'alta disponibilità* siano utili nella progettazione delle procedure di trasferimento ASP indipendente.

Informazioni su questa attività

Effettuare le seguenti operazioni per creare la configurazione mostrata in Figura 28 a pagina 442. I dati del gestore code sono connessi utilizzando IBM i NetServer.

- Creare le connessioni da ALPHA e BETA alla condivisione di directory su GAMMA che deve memorizzare i dati del gestore code. L'attività imposta anche le autorizzazioni, i profili utente e le parole d'ordine necessari.
- Creare il gestore code QM1 sul server IBM i , ALPHA.
- Aggiungere le informazioni di controllo del gestore code per QM1 sull'altro IBM i server, BETA.
- Avviare il gestore code, consentendo un'istanza in standby.

Procedura

1. Eseguire l'attività, “Creazione di una condivisione di rete per i dati del gestore code utilizzando NetServer su IBM i” a pagina 417.

   Di conseguenza, ALPHA e BETA hanno una condivisione, /QNTC/GAMMA/WMQ, che punta a /QIBM/UserData/mqm/qmgrs su GAMMA. I profili utente QMQM e QMQMADM hanno le autorizzazioni necessarie e QMQM ha parole d'ordine corrispondenti su tutti e tre i sistemi.
2. Creare il gestore code QM1 su ALPHA, salvando i dati del gestore code su GAMMA.

   ```
   CRTMQM MQMNAME(QM1) UDLMSGQ(SYSTEM.DEAD.LETTER.QUEUE)
   MQMDIRP(' /QNTC/GAMMA/WMQ ')
   ```

   Il percorso, utilizza NetServer per creare i dati del gestore code.
3. Utilizzare il comando, per esaminare i dati di configurazione IBM MQ creati da ALPHA.

   Le informazioni sono necessarie nel passo successivo.

   In questo esempio, la seguente configurazione viene creata in ALPHA per:

   ```
   Name=QM1
   Prefix=/QIBM/UserData/mqm
   Library=QMQM1
   Directory=QM1
   DataPath= /QNTC/GAMMA/WMQ /QM1
   ```
4. Creare un'istanza del gestore code di QM1 su BETA utilizzando il comando. Eseguire il seguente comando su BETA per modificare le informazioni di controllo del gestore code in BETA.

   ```
   ADDMQMINF MQMNAME(QM1)
   PREFIX('/QIBM/UserData/mqm')
   MQMDIR(QM1)
   MQMLIB(QMQM1)
   DATAPATH(' /QNTC/GAMMA/WMQ /QM1 ')
   ```

   Suggerimento: Copiare e incollare le informazioni di configurazione. La stanza del gestore code è la stessa su ALPHA e BETA.
5. Avviare le istanze del gestore code su ALPHA e BETA.
   a) Avviare la prima istanza su ALPHA, rendendola l'istanza attiva. Abilitazione del passaggio a un'istanza standby.

   ```
   STRMQM MQMNAME(QM1) STANDBY(*YES)
   ```

   b) Avviare la seconda istanza su BETA, rendendola l'istanza standby.

   ```
   STRMQM MQMNAME(QM1) STANDBY(*YES)
   ```

Risultati

Utilizzare per controllare lo stato del gestore code:

1. Lo stato dell'istanza del gestore code su ALPHA deve essere.
2. Lo stato dell'istanza del gestore code su BETA dovrebbe essere.

Esempio

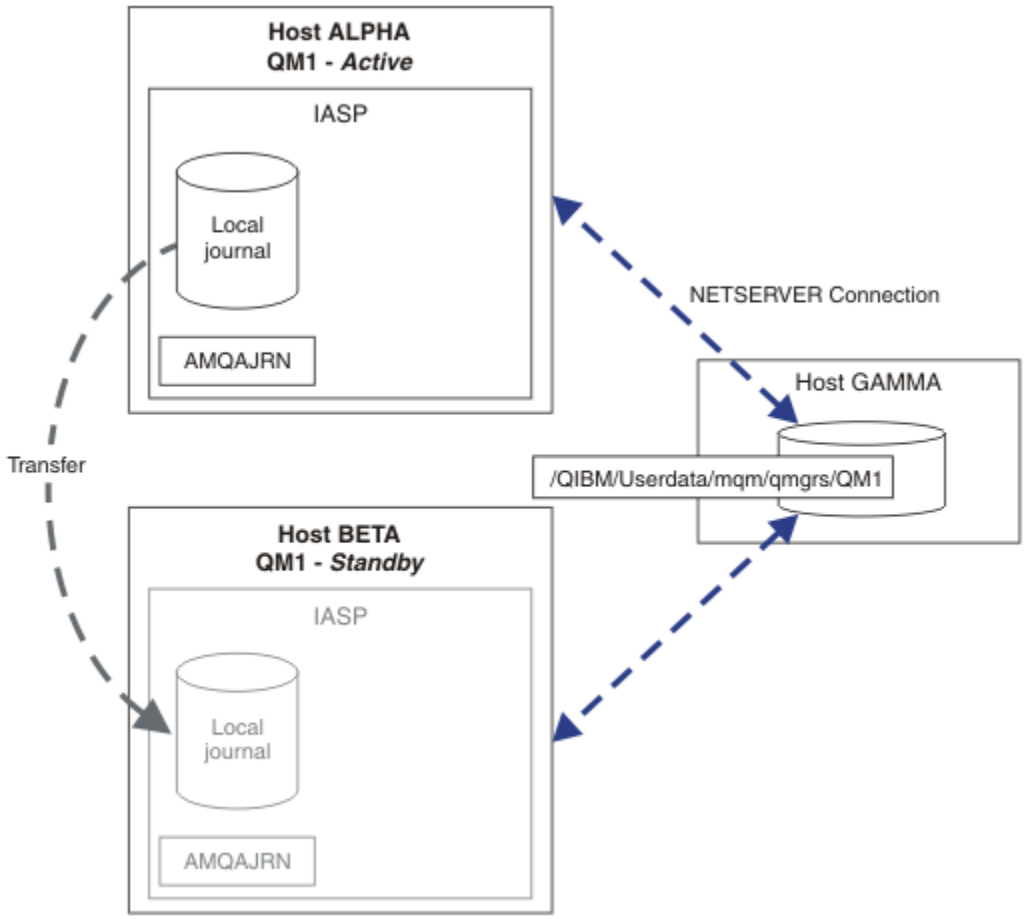


Figura 28. Trasferire un journal del gestore code utilizzando un ASP indipendente

Operazioni successive

- Verificare che le istanze attive e in standby si spostino automaticamente. È possibile eseguire i programmi di esempio di alta disponibilità di esempio per verificare il passaggio; consultare Programmi di esempio di alta disponibilità. I programmi di esempio sono client ' C'. È possibile eseguirli da una piattaforma Windows o Unix.
  1. Avviare i programmi di esempio ad alta disponibilità.
  2. Su ALPHA, terminare il gestore code che richiede il passaggio su:

     ```
     ENDMQM MQMNAME(QM1) OPTION(*IMMED) ALSWITCH(*YES)
     ```

  3. Verificare che l'istanza su BETA sia attiva.
  4. Riavvia su ALPHA

     ```
     STRMQM MQMNAME(QM1) STANDBY(*YES)
     ```

- Esaminare le configurazioni alternative di alta disponibilità:
  1. Utilizzare NetServer per posizionare i dati del gestore code su un server IBM i .
  2. Invece di utilizzare un ASP indipendente per trasferire il journal del gestore code sul server standby, utilizzare la registrazione su giornale remota per eseguire il mirroring del journal sul server standby.

IBM i *ASP indipendenti e alta disponibilità*

Gli ASP indipendenti consentono alle applicazioni e ai dati di essere spostati tra i server. La flessibilità degli ASP indipendenti significa che sono la base per alcune soluzioni di alta disponibilità IBM i . Nel considerare se utilizzare un ASP o un ASP indipendente per il journal del gestore code, è necessario considerare un'altra configurazione ad alta disponibilità basata su ASP indipendenti.

Gli ASP (auxiliary storage pools) sono un blocco di creazione dell'architettura di IBM i . Le unità disco sono raggruppate insieme per formare un singolo ASP. Posizionando gli oggetti in diversi ASP, è possibile proteggere i dati in un ASP da eventuali errori del disco in un altro ASP.

Ogni server IBM i ha almeno un ASP *di base* , noto come ASP di sistema. È designato come ASP1e talvolta è noto come *SYSBAS. È possibile configurare fino a 31 ASP *utente* di base aggiuntivi che non sono distinguibili dall'ASP di sistema dal punto di vista dell'applicazione, poiché condividono lo stesso spazio dei nomi. Utilizzando più ASP di base per distribuire le applicazioni su più dischi è possibile migliorare le prestazioni e ridurre il tempo di recupero. L'utilizzo di più ASP di base può anche fornire un certo grado di isolamento rispetto al malfunzionamento del disco, ma non migliora l'affidabilità in generale.

Gli ASP indipendenti sono un tipo speciale di ASP. Sono spesso chiamati lotti dischi indipendenti. I lotti dischi indipendenti sono componenti chiave dell'alta disponibilità IBM i . È possibile memorizzare i dati e le applicazioni che si considerano indipendenti dal sistema corrente a cui sono collegati sulle unità di memoria disco. È possibile configurare ASP indipendenti commutabili o non commutabili. Da una prospettiva di disponibilità, generalmente si è interessati solo agli ASP indipendenti commutabili, che possono essere trasferiti automaticamente da server a server. Come risultato è possibile spostare le applicazioni e i dati sull'ASP indipendente da server a server.

A differenza degli ASP utente di base, gli ASP indipendenti non condividono lo stesso spazio nome dell'ASP di sistema. Le applicazioni che gestiscono gli ASP utente richiedono modifiche per gestire un ASP indipendente. È necessario verificare che il software e il software di terze parti utilizzati funzionino in un ambiente ASP indipendente.

Quando l'ASP indipendente è collegato ad un altro server, lo spazio nome dell'ASP indipendente deve essere combinato con lo spazio nome del sistema ASP. Questo processo è denominato *attivazione* dell'ASP indipendente. È possibile attivare un ASP indipendente senza eseguire l'IPL del server. Il supporto cluster è richiesto per trasferire automaticamente gli ASP indipendenti da un server ad un altro.

Costruire soluzioni affidabili con ASP indipendenti

La registrazione su giornale in un ASP indipendente, piuttosto che la registrazione su giornale in un ASP e l'utilizzo della replica del journal, fornisce un mezzo alternativo per fornire al gestore code in standby una copia del journal locale dall'istanza del gestore code in errore. Per trasferire automaticamente l'ASP indipendente ad un altro server, è necessario avere installato e configurato il supporto cluster. Esistono diverse soluzioni di alta disponibilità per gli ASP indipendenti basate sul supporto del cluster e sul mirroring del disco di basso livello, che è possibile combinare o sostituire utilizzando gestori code a più istanze.

Il seguente elenco descrive i componenti necessari per creare una soluzione affidabile basata su ASP indipendenti.

Journaling
I gestori code e altre applicazioni utilizzano i journal locali per scrivere i dati persistenti in modo sicuro sul disco per proteggerli dalla perdita di dati in memoria a causa di un errore del server. Questa è a volte definita consistenza point - in - time. Non garantisce la coerenza di più aggiornamenti che si verificano in un periodo di tempo.

Controllo del commit
Utilizzando le transazioni globali, è possibile coordinare gli aggiornamenti di messaggi e database in modo che i dati scritti nel journal siano congruenti. Fornisce coerenza per un periodo di tempo utilizzando un protocollo di commit a due fasi.

Disco commutato

I dischi commutati vengono gestiti dal CRG unità in un cluster HA. CRG commuta automaticamente gli ASP indipendenti in un nuovo server in caso di interruzione non pianificata. I CRG sono geograficamente limitati all'estensione del bus IO locale.

Configurando il giornale locale su un ASP indipendente commutabile, è possibile trasferirlo su un server differente e riprendere l'elaborazione dei messaggi. Nessuna modifica ai messaggi persistenti effettuata senza il controllo del punto di sincronizzazione o di cui è stato eseguito il commit con il controllo del punto di sincronizzazione, viene persa, a meno che l'ASP indipendente non abbia esito negativo.

Se si utilizza sia la registrazione su giornale che il controllo del commit su ASP indipendenti commutabili, è possibile trasferire i journal del database e i journal del gestore code su un server differente e riprendere l'elaborazione delle transazioni senza alcuna perdita di congruenza o transazioni sottoposte a commit.

XSM (Cross - site mirroring)
XSM esegue il mirroring dell'ASP indipendente primario in un ASP indipendente secondario geograficamente remoto su una rete TCP/IP e trasferisce il controllo automaticamente in caso di errore. È possibile configurare un mirror sincrono o asincrono. Il mirroring sincrono riduce le prestazioni del gestore code poiché i dati vengono sottoposti a mirroring prima del completamento delle operazioni di scrittura sul sistema di produzione, ma garantisce che l'ASP indipendente secondario sia aggiornato. Mentre se si utilizza il mirroring asincrono, non è possibile garantire che l'ASP indipendente secondario sia aggiornato. Il mirroring asincrono mantiene la congruenza dell'ASP indipendente secondario.

Ci sono tre tecnologie XSM.

Mirroring geografico
Il mirroring geografico è un'estensione del cluster, che consente di commutare gli ASP indipendenti in un'ampia area. Ha entrambe le modalità sincrone e asincrone. È possibile garantire l'alta disponibilità solo in modalità sincrona, ma la separazione degli ASP indipendenti potrebbe influire troppo sulle prestazioni. È possibile combinare il mirroring geografico con il disco commutato per fornire l'alta disponibilità localmente e il ripristino di emergenza in remoto.

Mirroring metro
Il mirroring metro è un servizio a livello di unità che fornisce il mirroring sincrono locale veloce su distanze più lunghe rispetto al bus locale. È possibile combinarlo con un gestore code a più istanze per fornire l'alta disponibilità del gestore code e, avendo due copie dell'ASP indipendente, l'elevata disponibilità del journal del gestore code.

Mirroring globale
Il mirroring globale è un servizio a livello unità che fornisce il mirroring asincrono ed è adatto per il backup e il ripristino di emergenza su distanze più lunghe, ma non è una scelta normale per l'alta disponibilità, perché mantiene solo la congruenza del punto nel tempo piuttosto che la valuta.

I principali punti di decisione da considerare sono:

ASP o ASP indipendente?
Non è necessario eseguire un cluster IBM i HA per utilizzare gestori code a più istanze. È possibile scegliere gli ASP indipendenti, se si stanno già utilizzando gli ASP indipendenti, oppure si hanno requisiti di disponibilità per altre applicazioni che richiedono gli ASP indipendenti. Potrebbe essere utile combinare gli ASP indipendenti con i gestori code a più istanze per sostituire il monitoraggio del gestore code come mezzo per rilevare gli errori del gestore code.

Disponibilità?
Qual è l'obiettivo del tempo di recupero (RTO)? Se si richiede l'aspetto di un comportamento quasi ininterrotto, allora quale soluzione ha il tempo di recupero più veloce?

Disponibilità giornale?
Come eliminare il giornale come un singolo punto di errore. È possibile adottare una soluzione hardware, utilizzando periferiche RAID 1 o superiori, oppure combinare o utilizzare una soluzione software utilizzando i journal di replica o il mirroring del disco.

Distanza?
La distanza tra le istanze del gestore code attive e in standby. Gli utenti possono tollerare il degrado delle prestazioni della replica in modo sincrono su distanze superiori a circa 250 metri?

Competenze?
C'è del lavoro da fare per automatizzare le attività amministrative coinvolte nella manutenzione ed esercizio regolare della soluzione. Le competenze richieste per eseguire l'automazione sono diverse per le soluzioni basate su ASP e ASP indipendenti.

IBM i *Eliminazione di un gestore code a più istanze su IBM i*

Prima di eliminare un gestore code a più istanze, arrestare la registrazione su giornale remota e rimuovere istanze del gestore code.

Prima di iniziare

1. In questo esempio, due istanze del gestore code QM1 sono definite sui server ALPHA e BETA. ALPHA è l'istanza attiva e BETA è lo standby. I dati del gestore code associati al gestore code QM1 sono memorizzati sul IBM i server GAMMA, utilizzando NetServer. Consultare “Creazione di un gestore code a più istanze utilizzando il mirroring del journal e NetServer su IBM i” a pagina 430.
2. ALPHA e BETA devono essere collegati in modo che tutti i giornali remoti definiti possano essere eliminati da IBM MQ.
3. Verificare che sia possibile accedere alla directory /QNTC e alla condivisione file della directory del server utilizzando i comandi di sistema **EDTF** o **WRKLNK**

Informazioni su questa attività

Prima di eliminare un gestore code a più istanze da un server utilizzando il comando **DLTMQM** , rimuovere tutte le istanze del gestore code su altri server utilizzando il comando **RMVMQMINF** .

Quando si rimuove un'istanza del gestore code utilizzando il comando **RMVMQMINF** , i journal locali e remoti con prefisso AMQe associati all'istanza, vengono eliminati. Vengono eliminate anche le informazioni di configurazione relative all'istanza del gestore code, locale per il server.

Non eseguire il comando **RMVMQMINF** sul server che contiene l'istanza rimanente del gestore code. In questo modo si impedisce a **DLTMQM** di funzionare correttamente.

Eliminare il gestore code utilizzando il comando **DLTMQM** . I dati del gestore code vengono rimossi dalla condivisione di rete. I giornali locali e remoti con prefisso AMQ e associati all'istanza vengono eliminati. **DLTMQM** elimina anche le informazioni di configurazione sull'istanza del gestore code, locale per il server.

Nell'esempio, esistono solo due istanze del gestore code. IBM MQ supporta una configurazione a più istanze in esecuzione che ha un'istanza del gestore code attiva e un'istanza in standby. Se sono state create ulteriori istanze del gestore code da utilizzare nelle configurazioni in esecuzione, rimuoverle, utilizzando il comando **RMVMQMINF** , prima di eliminare l'istanza rimanente.

Procedura

1. Eseguire il comando **CHGMQMJRN RMTJRNSTS** (*INACTIVE) su ciascun server per rendere inattivo il journaling remoto tra le istanze del gestore code.
   a) Su ALPHA:

```
CHGMQMJRN MQMNAME('QM1')
RMTJRNRDB('BETA') RMTJRNSTS(*INACTIVE)
```

   b) Su BETA:

```
CHGMQMJRN MQMNAME('QM1')
RMTJRNRDB('ALPHA') RMTJRNSTS(*INACTIVE)
```

2. Eseguire il comando **ENDMQM** su ALPHA, l'istanza del gestore code attivo, per arrestare entrambe le istanze di QM1.

```
ENDMQM MQMNAME(QM1) OPTION(*IMMED) INSTANCE(*ALL) ENDCCTJOB(*YES)
```

3. Eseguire il comando **RMVMQMINF** su ALPHA per rimuovere le risorse del gestore code per l'istanza da ALPHA e BETA.

```
RMVMQMINF MQMNAME(QM1)
```

RMVMQMINF rimuove le informazioni di configurazione del gestore code per QM1 da ALPHA. Se il nome del journal ha come prefisso AMQ, elimina il journal locale associato a QM1 da ALPHA. Se il nome del giornale è preceduto dal prefisso AMQ ed è stato creato un giornale remoto, rimuove anche il giornale remoto da BETA.

4. Eseguire il comando **DLTMQM** su BETA per eliminare QM1.

```
DLTMQM MQMNAME(QM1)
```

DLTMQM elimina i dati del gestore code dalla condivisione di rete su GAMMA. Rimuove le informazioni di configurazione del gestore code per QM1 da BETA. Se il nome del giornale ha come prefisso AMQ, elimina il giornale locale associato a QM1 da BETA. Se il nome del giornale è preceduto da AMQ ed è stato creato un giornale remoto, rimuove anche il giornale remoto da ALPHA.

Risultati

DLTMQM e **RMVMQMINF** eliminano i giornali locali e remoti creati da **CRTMQM** e **ADDMQJRN**. I comandi cancellano anche i ricevitori di giornale. I giornali e i ricevitori di giornale devono seguire la convenzione di denominazione dei nomi che iniziano con AMQ. **DLTMQM** e **RMVMQMINF** rimuovono gli oggetti del gestore code, i dati del gestore code e le informazioni di configurazione del gestore code da `mqs.ini`.

Operazioni successive

Un approccio alternativo consiste nell'immettere i seguenti comandi dopo la disattivazione della registrazione su giornale nel passo “1” a pagina 445 e prima di terminare le istanze del gestore code. Oppure, se non è stata seguita la convenzione di denominazione, è necessario cancellare i giornali e i ricevitori di giornale per nome.

1. Su ALPHA:

```
RMVMQMJRN MQMNAME('QM1') RMTJRNRDB('BETA')
```

2. Su BETA:

```
RMVMQMJRN MQMNAME('QM1') RMTJRNRDB('ALPHA')
```

Dopo aver eliminato i journal, continuare con il resto dei passi.

IBM i *Esecuzione del backup di un gestore code a più istanze su IBM i*

La procedura mostra come eseguire il backup degli oggetti del gestore code sul server locale e i dati del gestore code sul server di file di rete. Adattare l'esempio per eseguire il backup dei dati per altri gestori code.

Prima di iniziare

In questo esempio, i dati del gestore code associati con il gestore code QM1 vengono memorizzati sul IBM i server denominato GAMMA, utilizzando NetServer. Consultare “Creazione di un gestore code a più istanze utilizzando il mirroring del journal e NetServer su IBM i” a pagina 430. IBM MQ è installato sui server ALPHA e BETA. Il gestore code, QM1, è configurato su ALPHA e BETA.

Informazioni su questa attività

IBM i non supporta il salvataggio dei dati da una directory remota. Salvare i dati del gestore code su un file system remoto utilizzando le procedure di backup locali sul server del filesystem. In questa attività, il file system di rete si trova su un server IBM i , GAMMA. Viene eseguito il backup dei dati del gestore code in un file di salvataggio su GAMMA.

Se il file system di rete si trovava su Windows o Linux, è possibile memorizzare i dati del gestore code in un file compresso e salvarli. Se si dispone di un sistema di backup, come Tivoli Storage Manager, utilizzarlo per eseguire il backup dei dati del gestore code.

Procedura

1. Creare un file di salvataggio su ALPHA per la libreria del gestore code associato a QM1.

   Utilizzare il nome della libreria del gestore code per denominare il file di salvataggio.

   ```
   CRTSAVF FILE(QGPL/QMQM1)
   ```

2. Salvare la libreria del gestore code nel file di salvataggio su ALPHA.

   ```
   SAVLIB LIB(QMQM1) DEV(*SAVF) SAVF(QGPL/QMQM1)
   ```

3. Creare un file di salvataggio per la directory dei dati del gestore code su GAMMA.

   Utilizzare il nome gestore code per denominare il file di salvataggio.

   ```
   CRTSAVF FILE(QGPL/QMDQM1)
   ```

4. Salvare la copia dei dati del gestore code dalla directory locale su GAMMA.

   ```
   SAV DEV('/QSYS.LIB/QGPL.LIB/QMDQM1.FILE') OBJ('/QIBM/Userdata/mqm/qmgrs/QM1')
   ```

IBM i *Comandi per impostare gestori code a più istanze*

IBM MQ dispone di dei comandi per semplificare la configurazione della replica del journal, l'aggiunta di nuove istanze del gestore code e la configurazione dei gestori code per utilizzare l'ASP indipendente.

I comandi di giornale per creare e gestire i giornali locali e remoti sono:

ADDMQMJRN
Con questo comando è possibile creare dei journal locali e remoti denominati per un'istanza del gestore code e configurare se la replica è sincrona o asincrona, qual è il timeout sincrono e se il journal remoto deve essere attivato immediatamente.

CHGMQMJRN
Il comando modifica il timeout, lo stato e i parametri di consegna che interessano i journal di replica.

RMVMQMJRN
Rimuove i journal denominati *remoti* da un'istanza del gestore code.

WRKMQMJRN
Elenca lo stato dei journal locali e remoti per un'istanza del gestore code locale.

Aggiungere e gestire istanze aggiuntive del gestore code utilizzando i seguenti comandi, che modificano il file `mqs.ini` .

ADDMQMINF
Il comando usa le informazioni estratte dal file `mqs.ini` con il comando DSPMQMINF per aggiungere una nuova istanza del gestore code su un server IBM i differente.

RMVMQMINF
Rimuovere un'istanza del gestore code. Utilizzare questo comando per rimuovere un'istanza di un gestore code esistente o per rimuovere le informazioni di configurazione per un gestore code che è stato eliminato da un altro server.

Il comando **CRTMQM** dispone di tre parametri per assistere nella configurazione di un gestore code a più istanze,

MQMDIRP (*DFT | *prefisso - indirizzario*)
Utilizzare questo parametro per selezionare un punto di montaggio associato ai dati del gestore code sulla memoria di rete.

ASP (*SYSTEM|*ASPDEV| *numero - lotto - memoria - ausiliaria*)
Specificare *SYSTEMo un *numero ASP (auxiliary - storage - pool - number)* per posizionare il journal del gestore code sul sistema o su un ASP utente di base. Selezionare l'opzione *ASPDEV e impostare un nome unità utilizzando il parametro **ASPDEV** , per posizionare il journal del gestore code su un ASP indipendente.

ASPDEV (*ASP|*nome - unità*)
Specificare un *nome - unità* di un'unità ASP indipendente principale o secondaria. La selezione di *ASP ha lo stesso risultato della specifica di **ASP** (*SYSTEM).

IBM i Considerazioni sulle prestazioni e sul failover del disco su IBM i

Utilizzare diversi lotti di memoria ausiliaria per migliorare le prestazioni e l'affidabilità.

Se si utilizza un numero elevato di messaggi persistenti o messaggi di grandi dimensioni nelle applicazioni, il tempo impiegato per scrivere questi messaggi sul disco diventa un fattore significativo nelle prestazioni del sistema.

Accertarsi di disporre di un'attivazione disco sufficiente per far fronte a questa possibilità oppure considerare un ASP (Auxiliary Storage Pool) separato in cui conservare i ricevitori di giornale del gestore code.

È possibile specificare su quale ASP vengono memorizzati la libreria del gestore code e i journal quando si crea il gestore code utilizzando il parametro ASP di **CRTMQM**. Per impostazione predefinita, la libreria del gestore code e i journal e dati IFS vengono memorizzati nell'ASP di sistema.

Gli ASP consentono l'isolamento degli oggetti su una o più specifiche unità disco. Ciò può ridurre la perdita di dati a causa di un malfunzionamento del supporto disco. Nella maggior parte dei casi, vengono persi solo i dati memorizzati sulle unità disco nell'ASP interessato.

Si consiglia di archiviare la libreria del gestore code e i dati journal in ASP utente separati in quello del file system IFS root per fornire il failover e ridurre il conflitto del disco.

Per ulteriori informazioni, consultare Backup e ripristino nella documentazione di IBM i .

IBM i Utilizzo di SAVLIB per salvare le librerie IBM MQ su IBM i

Non è possibile utilizzare `SAVLIB LIB(*ALLUSR)` per salvare le librerie IBM MQ , poiché tali librerie hanno nomi che iniziano con Q.

È possibile utilizzare `SAVLIB LIB(QM*)` per salvare tutte le librerie del gestore code, ma solo se si sta utilizzando un'unità di salvataggio diversa da *SAVF. Per `DEV(*SAVF)`, è necessario utilizzare un comando SAVLIB per ogni libreria del gestore code sul sistema.

IBM i In corso di quiesceIBM MQ for IBM i

Questa sezione spiega come disattivare (terminare correttamente) IBM MQ for IBM i.

Per disattivare IBM MQ for IBM i:

1. Collegarsi a una nuova sessione IBM MQ for IBM i interattiva, assicurandosi di non accedere ad alcun oggetto.
2. Accertarsi di avere:
   - Autorizzazione *ALLOBJ o autorizzazione di gestione oggetto per la libreria QMQM
   - Autorizzazione sufficiente per utilizzare il comando ENDSBS
3. Avvisare tutti gli utenti che si sta per arrestare IBM MQ for IBM i.
4. Il modo in cui si procede dipende dal fatto che si desidera arrestare (disattivare) un singolo gestore code (dove potrebbero esistere altri) (consultare “Arresto di un singolo gestore code per IBM MQ for IBM i” a pagina 449) o tutti i gestori code (consultare “Arresto di tutti i gestori code per IBM MQ for IBM i” a pagina 451).
5. Arrestare il server mqweb immettendo il seguente comando in qshell:

```
/QIBM/ProdData/mqm/bin/endmqweb
```

ENDMQM parametro ENDCCTJOB (*YES)

Il parametro ENDMQM ENDCCTJOB (*YES) funziona in modo diverso in IBM MQ for IBM i V6.0 e successive rispetto alle versioni precedenti.

Nelle versioni precedenti, quando si specifica ENDCCTJOB (*YES), MQ termina forzatamente le applicazioni.

Su IBM MQ for IBM i V6.0 o successiva, quando si specifica ENDCCTJOB (*YES), le proprie applicazioni non vengono terminate ma vengono invece scollegate dal gestore code.

Se si specifica ENDCCTJOB (*YES) e si dispone di applicazioni che non vengono scritte per rilevare che un gestore code è in fase di chiusura, la volta successiva che viene emessa una nuova chiamata MQI, la chiamata restituirà un errore MQRC_CONNECTION_BROKEN (2009).

Come alternativa all'utilizzo di ENDCCTJOB (*YES), utilizzare il parametro ENDCCTJOB (*NO) e utilizzare l'opzione WRKMQM 22 (Gestione lavori) per terminare manualmente tutti i lavori dell'applicazione che impediranno il riavvio di un gestore code.

IBM i Arresto di un singolo gestore code per IBM MQ for IBM i

Utilizzare queste informazioni per comprendere i tre tipi di arresto.

Nelle procedure che seguono, viene utilizzato un nome di gestore code di esempio `QMgr1` e un nome di sottosistema di esempio SUBX. Sostituire questi nomi con i propri valori, se necessario.

Chiusura pianificata

Arresto pianificato di un gestore code su IBM i

1. Prima della chiusura, eseguire:

```
RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) MQMNAME(QMgr1) DSPJRNDTA(*YES)
```

2. Per arrestare il gestore code, eseguire:

```
ENDMQM MQMNAME(QMgr1) OPTION(*CNTRLD)
```

   Se QMgr1 non termina, il canale o le applicazioni sono probabilmente occupati.

3. Se è necessario arrestare immediatamente QMgr1 , eseguire quanto riportato di seguito:

```
ENDMQM MQMNAME(QMgr1) OPTION(*IMMED)
ENDCCTJOB(*YES) TIMEOUT(15)
```

Arresto non pianificato

1. Per arrestare il gestore code, eseguire:

```
ENDMQM MQMNAME(QMgr1) OPTION(*IMMED)
```

   Se QMgr1 non termina, il canale o le applicazioni sono probabilmente occupati.
2. Se è necessario arrestare QMgr1 immediatamente, eseguire quanto riportato di seguito:

```
ENDMQM MQMNAME(QMgr1) OPTION(*IMMED)
ENDCCTJOB(*YES) TIMEOUT(15)
```

Chiusura in condizioni anomale

1. Per arrestare il gestore code, eseguire:

```
ENDMQM MQMNAME(QMgr1) OPTION(*IMMED)
```

   Se QMgr1 non termina, continuare con il passo 3 a condizione che:

   - QMgr1 si trova nel proprio sottosistema oppure
   - È possibile terminare tutti i gestori code che condividono lo stesso sottosistema di QMgr1. Utilizzare la procedura di arresto non pianificato per tutti i gestori code.
2. Dopo aver eseguito tutte le operazioni della procedura per tutti i gestori code che condividono il sottosistema (SUBX negli esempi), eseguire:

```
ENDSBS SUBX *IMMED
```

   Se il completamento di questo comando non riesce, arrestare tutti i gestori code, utilizzando la procedura di chiusura non pianificata ed eseguire un IPL sulla macchina.

   Avviso: Non utilizzare ENDJOBABN per i lavori IBM MQ che non riescono a terminare come risultato di ENDJOB o ENDSBS, a meno che non si sia pronti ad eseguire un IPL sulla macchina immediatamente dopo.
3. Avviare il sottosistema eseguendo:

```
STRSBS SUBX
```

4. Chiudere immediatamente il gestore code, eseguendo:

```
ENDMQM MQMNAME(QMgr1) OPTION(*IMMED)
ENDCCTJOB(*YES) TIMEOUT(10)
```

5. Riavviare il gestore code eseguendo:

```
STRMQM MQMNAME(QMgr1)
```

   Se questo non riesce, e si:

   - Aver riavviato la macchina eseguendo un IPL oppure

- Avere solo un singolo gestore code

Riordinare la memoria condivisa IBM MQ eseguendo:

```
ENDMQM MQMNAME(*ALL) OPTION(*IMMED)
ENDCCTJOB(*YES) TIMEOUT(15)
```

prima di ripetere il passo 5.

Se il riavvio del gestore code richiede più di pochi secondi, IBM MQ aggiunge i messaggi di stato in modo intermittente al log del lavoro che descrive l'avanzamento dell'avvio.

Se hai ancora problemi a riavviare il tuo gestore code, contatta il supporto IBM . Qualsiasi ulteriore azione che potrebbe essere intrapresa potrebbe danneggiare il gestore code, lasciando IBM MQ nell'impossibilità di eseguire il ripristino.

IBM i Arresto di tutti i gestori code per IBM MQ for IBM i

Utilizzare queste informazioni per comprendere i tre tipi di arresto.

Le procedure sono quasi le stesse di un singolo gestore code, ma utilizzando *ALL invece del nome del gestore code, laddove possibile, e utilizzando un comando che utilizza ripetutamente ciascun nome di gestore code. Nelle procedure, viene utilizzato un nome di gestore code di esempio QMgr1 e un nome di sottosistema di esempio SUBX. Sostituirli con i propri.

Chiusura pianificata

1. Un'ora prima della chiusura, eseguire:

   ```
   RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) MQMNAME(QMgr1) DSPJRNDTA(*YES)
   ```

   Ripetere questa operazione per ogni gestore code che si desidera arrestare.
2. Per arrestare il gestore code, eseguire:

   ```
   ENDMQM MQMNAME(QMgr1) OPTION(*CNTRLD)
   ```

   Ripetere questa operazione per ogni gestore code che si desidera arrestare; è possibile eseguire comandi separati in parallelo.

   Se un gestore code non termina entro un periodo di tempo ragionevole (ad esempio 10 minuti), procedere con il passo 3.
3. Per arrestare immediatamente tutti i gestori code, eseguire quanto segue:

   ```
   ENDMQM MQMNAME(*ALL) OPTION(*IMMED)
   ENDCCTJOB(*YES) TIMEOUT(15)
   ```

Arresto non pianificato

1. Per arrestare un gestore code, eseguire:

   ```
   ENDMQM MQMNAME(QMgr1) OPTION(*IMMED)
   ```

   Ripetere questa operazione per ogni gestore code che si desidera arrestare; è possibile eseguire comandi separati in parallelo.

   Se i gestori code non vengono terminati, il canale o le applicazioni sono probabilmente occupati.
2. Se è necessario arrestare immediatamente i gestori code, eseguire le seguenti operazioni:

```
ENDMQM MQMNAME(*ALL) OPTION(*IMMED)
ENDCCTJOB(*YES) TIMEOUT(15)
```

Chiusura in condizioni anomale

1. Per arrestare i gestori code, eseguire:

   ```
   ENDMQM MQMNAME(QMgr1) OPTION(*IMMED)
   ```

   Ripetere questa operazione per ogni gestore code che si desidera arrestare; è possibile eseguire comandi separati in parallelo.
2. Terminare i sistemi secondari (SUBX negli esempi), eseguendo:

   ```
   ENDSBS SUBX *IMMED
   ```

   Ripetere questa operazione per ogni sottosistema che si desidera arrestare; i comandi separati possono essere eseguiti in parallelo.

   Se questo comando non viene completato, eseguire un IPL sul sistema.

   Avviso: Non utilizzare ENDJOBABN per i lavori che non riescono a terminare come risultato di ENDJOB o ENDSBS, a meno che non si sia pronti ad eseguire un IPL sul sistema immediatamente dopo.
3. Avviare i sottosistemi eseguendo:

   ```
   STRSBS SUBX
   ```

   Ripetere questa operazione per ogni sottosistema che si desidera avviare.
4. Chiudere immediatamente i gestori code, eseguendo:

   ```
   ENDMQM MQMNAME(*ALL) OPTION(*IMMED)
   ENDCCTJOB(*YES) TIMEOUT(15)
   ```
5. Riavviare i gestori code eseguendo:

   ```
   STRMQM MQMNAME(QMgr1)
   ```

   Ripetere questa operazione per ogni gestore code che si desidera avviare.

   Se un riavvio del gestore code impiega più di pochi secondi, IBM MQ visualizzerà i messaggi di stato in modo intermittente che descrivono in dettaglio l'avanzamento dell'avvio.

   Se hai ancora problemi a riavviare un gestore code, contatta il supporto IBM . Qualsiasi ulteriore azione che si potrebbe intraprendere potrebbe danneggiare i gestori code, lasciando MQSeries o IBM MQ incapaci di eseguire il ripristino.

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 “Amministrazione mediante IBM MQ Explorer” on page 120.

Related concepts

IBM MQ for z/OS concepts

Related tasks

“AmministrazioneIBM MQ” on page 7
Per gestire i gestori code IBM MQ e le risorse associate, scegliere il metodo preferito da una serie di attività che è possibile utilizzare per attivare e gestire tali risorse.

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 453.

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 “Utilizzo di IBM MQ Programmable Command Format” on page 26.

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 467.

  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 464.
- 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 475.
- 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 483 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 454 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

Table 25. Summary of the main MQSC and PCF commands by object type (continued)				
MQSC command	**ALTER**	**DEFINE**	**DISPLAY**	**DELETE**
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 454.

Table 26 on page 455 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

Table 26. Sources from which to run MQSC commands (continued)

Command	CSQINP1	CSQINP2	z/OS console	Command input queue and server
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
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

Table 26. Sources from which to run MQSC commands (continued)

Command	CSQINP1	CSQINP2	z/OS console	Command input queue and server
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
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

Table 26. Sources from which to run MQSC commands (continued)				
Command	**CSQINP1**	**CSQINP2**	**z/OS console**	**Command input queue and server**
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
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

Table 26. Sources from which to run MQSC commands (continued)				
Command	**CSQINP1**	**CSQINP2**	**z/OS console**	**Command input queue and server**
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 542.

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 467
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 477.

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			
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 475
  - 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 467. 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 477.

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 481.

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 481.
- If you are using a staged approach, see “Changing your cluster-sender channels using a phased approach” on page 482.

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 "Utilizzo di IBM MQ Programmable Command Format" on page 26.

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 489 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 489.

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 490.

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 Messaggi IBM MQ for z/OS , codici di completamento e di errore 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 453.

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 496
- “Managing the bootstrap data set (BSDS)” on page 505
- “Managing page sets” on page 512
- “How to back up and recover page sets” on page 519
- “How to back up and restore queues using CSQUTIL” on page 522
- “Managing buffer pools” on page 522

- “Managing queue sharing groups and shared queues on z/OS” on page 524

Related concepts

IBM MQ for z/OS concepts

“Administering IBM MQ for z/OS” on page 452
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 453
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 533
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 475
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 498:

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 “Introduzione a IBM MQ Programmable Command Formats” 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 506. 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 508.

     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 508 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 504. 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 503 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 504

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 505
- “Changing the BSDS” on page 506
- “Recovering the BSDS” on page 510

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 511. Do not copy any data to the second replacement BSDS, you do that in step "5" on page 512.

   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 519
- “How to delete page sets” on page 522
- “How to back up and restore queues using CSQUTIL” on page 522

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 516 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 514.

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 514 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 516 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 514. 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 540). 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 518.
5. Restart the queue manager using the page set created in step “3” on page 518.
6. Run the LOAD function of CSQUTIL to load back all the messages saved during step “2” on page 518.
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 519
- "Backing up page sets" on page 520
- "Recovering page sets" on page 521
- How to delete page sets

For information about how to create a point of recovery for shared resources, see "Recovering shared queues" on page 527.

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 520).

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 520).
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 520, 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 523
- "How to delete a buffer pool" on page 523

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 Pianificazione su 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 522 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 525
5. Define application structures to IBM MQ by following the steps in "Adding a coupling facility structure" on page 532.
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 526).

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 527
- “Moving shared queues” on page 528
- “Migrating non-shared queues to shared queues” on page 530
- 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 528), 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 528.
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 528 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 529 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 528). Figure 31 on page 529 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 528).

Figure 32 on page 530 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 529 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 531 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 529 or Figure 33 on page 531.
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 531.

```
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 532
- “Removing a coupling facility structure” on page 532

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 533
- “Restarting after an abnormal termination” on page 534
- “Restarting if you have lost your page sets” on page 534
- “Restarting if you have lost your log data sets” on page 534
- 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 459.

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 459. 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 546.

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 519.

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 538 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 535.
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 535), 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 544 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.

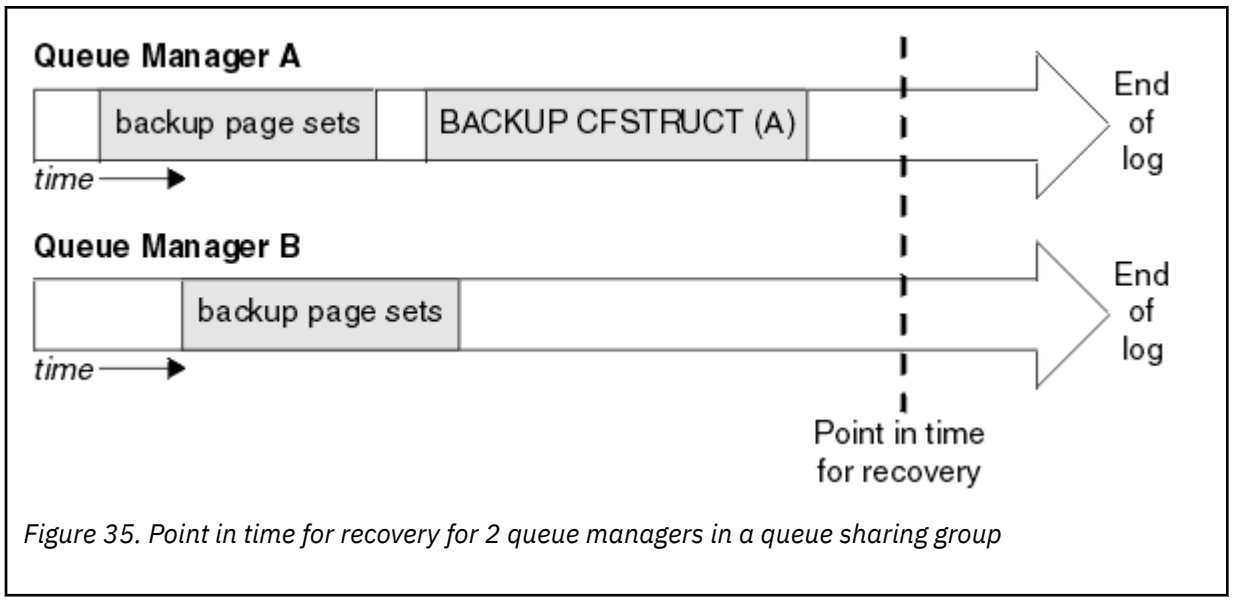


Figure 35. Point in time for recovery for 2 queue managers in a queue sharing group

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 539.
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 539), 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 539.

   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 541).

   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 541 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 542
- “ARM policies” on page 543
- “Using ARM in an IBM MQ network” on page 544

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 544).

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 544.

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 543 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 545
- “Restarting on a different z/OS image with LU 6.2” on page 546

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 548 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 551.
- 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 Messaggi IBM MQ for z/OS , codici di completamento e di errore 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 562
- “Controlling IMS connections” on page 563
- Controlling bridge queues
- “Resynchronizing the IMS bridge” on page 564
- Working with tpipe names
- Deleting messages from IMS
- Deleting tpipes
- “IMS Transaction Expiration” on page 566

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.

AmministrazioneIBM MQ Internet Pass-Thru

Questa sezione descrive come amministrare IBM MQ Internet Pass-Thru (MQIPT).

Configurare MQIPT apportando modifiche al file di configurazione `mqipt.conf` come descritto in Configurazione di IBM MQ Internet Pass-Thru. Per amministrare MQIPT, incluso l'aggiornamento di MQIPT per rendere effettive le modifiche alla configurazione senza riavviare MQIPT, utilizzare il comando **mqiptAdmin** . Per informazioni sulla gestione di MQIPT utilizzando il comando **mqiptAdmin** , consultare “Amministrazione di MQIPT utilizzando la riga comandi” a pagina 570.

avvio e arrestoMQIPT

È possibile avviare MQIPT dalla riga comandi o avviarlo automaticamente quando il sistema viene avviato. È possibile arrestare MQIPT utilizzando il comando **mqiptAdmin** .

Avvio di MQIPT dalla riga comandi

MQIPT viene installato in una directory di installazione, ad esempio:

- Windows `C:\MQIPT` su sistemi Windows , con script eseguibili in `C:\MQIPT\bin`
- Linux AIX `/opt/mqipt` su sistemi AIX and Linux , con script eseguibili in `/opt/mqipt/bin`

MQIPT utilizza anche una directory home, contenente il file di configurazione `mqipt.conf` e tutti i file emessi da MQIPT quando è in esecuzione. Le seguenti sottodirectory della directory home MQIPT vengono create automaticamente quando MQIPT viene richiamato per la prima volta:

- Una directory `errors` in cui vengono scritti i file di traccia e First Failure Support Technology (FFST)
- Una directory `logs` in cui viene conservato il log di connessione

L'ID utente con cui viene eseguito MQIPT deve disporre dell'autorizzazione per creare tali directory oppure, in alternativa, le directory devono già esistere e l'ID utente deve disporre dell'autorizzazione per creare, leggere e scrivere file in esse. Inoltre, se si utilizza una politica Java security manager , la politica di sicurezza deve concedere le autorizzazioni richieste per queste directory. Per ulteriori informazioni sulle impostazioni della politica di Security Manager, fare riferimento a Java security manager.

È possibile utilizzare la directory di installazione come directory home. Se si utilizza questa directory, è necessario verificare che l'ID utente con cui viene eseguito MQIPT disponga delle autorizzazioni appropriate e che qualsiasi politica di Security Manager sia configurata correttamente.

Per avviare MQIPT, utilizzare il comando **mqipt** , che si trova nella directory `bin` della directory di installazione MQIPT . Ad esempio, il seguente comando avvia un'istanza di MQIPT che utilizza la directory `C:\mqiptHome` come directory home:

```
mqipt C:\mqiptHome
```

Per ulteriori informazioni sul comando **mqipt** , consultare mqipt (start MQIPT).

È possibile utilizzare il comando **mqipt** per specificare un nome da assegnare all'istanza MQIPT in fase di avvio. Il nome dell'istanza MQIPT viene utilizzato per gestire le istanze locali di MQIPT con il comando **mqiptAdmin** senza dover utilizzare una porta comandi. Se questo parametro non viene specificato, il nome della directory home MQIPT viene utilizzato come nome dell'istanza MQIPT .

I messaggi della console mostrano lo stato di MQIPT. Se si verifica un errore, consultare Risoluzione dei problemi IBM MQ Internet Pass-Thru. I seguenti messaggi sono un esempio dell'output quando MQIPT viene avviato correttamente:

```
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
```

Avvio automatico di MQIPT

È possibile installare MQIPT come servizio di sistema che viene avviato automaticamente all'avvio del sistema. Utilizzare il comando **mqiptService** per installare e disinstallare il servizio MQIPT .

- Windows Sui sistemi Windows , il comando **mqiptService** installa MQIPT come servizio Windows .
- Linux AIX Sui sistemi AIX and Linux , il comando **mqiptService** installa MQIPT come servizio init System V che viene avviato all'avvio del sistema. Su sistemi di Linux che non supportano System V init, utilizzare un altro metodo, come `systemd`, per gestire MQIPT come un servizio.

Quando il servizio MQIPT viene avviato, vengono avviati tutti gli instradamenti MQIPT attivi. Quando il servizio viene arrestato, tutti i percorsi vengono immediatamente arrestati.

È possibile installare solo un servizio MQIPT su un sistema, anche se sul sistema è presente più di una installazione di MQIPT .

Per ulteriori informazioni relative al comando **mqiptService** , consultare mqiptService (gestione del servizio MQIPT).

arrestoMQIPT

È possibile arrestare MQIPT utilizzando il comando **mqiptAdmin** con il parametro **-stop** .

Ad esempio, il seguente comando arresta un'istanza di MQIPT con il nome `mqipt1` in esecuzione localmente con lo stesso ID utente del comando **mqiptAdmin** :

```
mqiptAdmin -stop -n ipt1
```

Il comando **mqiptAdmin** si connette all'istanza attiva di MQIPT da gestire utilizzando uno dei seguenti metodi:

- collegandosi a un'istanza locale di MQIPT senza utilizzare la porta comandi.
- creando una connessione di rete a una porta comandi.

La chiusura remota deve essere abilitata impostando la proprietà **RemoteShutDown** su `true` prima che il comando **mqiptAdmin** possa essere utilizzato per arrestare MQIPT inviando un comando a una porta comandi.

Per ulteriori informazioni sulla gestione di MQIPT mediante l'utilizzo del comando **mqiptAdmin** , consultare “Amministrazione di MQIPT utilizzando la riga comandi” a pagina 570.

Specifica della chiave di codifica della parola d'ordine

Se la configurazione di MQIPT contiene password codificate utilizzando una chiave di codifica diversa dalla chiave predefinita, è necessario fornire la chiave di codifica password in un file che MQIPT può leggere all'avvio.

Il file della chiave di crittografia della parola d'ordine

Le password crittografate per essere memorizzate e utilizzate da MQIPT possono essere crittografate utilizzando una chiave di crittografia fornita dall'utente. Se non si fornisce una chiave di codifica, viene utilizzata la chiave di codifica predefinita. Non è necessario specificare una chiave di codifica della password, tuttavia è più sicuro farlo. Se non si specifica la propria chiave di codifica, viene utilizzata la chiave di codifica predefinita.

Se si fornisce una chiave di codifica della password, questa deve essere memorizzata in un file a cui è possibile accedere tramite il comando **mqiptPW** utilizzato per codificare le password e MQIPT. Le uniche limitazioni sul contenuto del file sono che deve contenere almeno un carattere e una sola riga di testo.

Nota: È necessario assicurarsi che le autorizzazioni file appropriate siano impostate sul file della chiave di codifica della parola d'ordine per impedire agli utenti non autorizzati di leggere la chiave di codifica. Solo l'utente che esegue il comando **mqiptPW** e l'utente con cui viene eseguito MQIPT necessitano dell'autorità per leggere la chiave di codifica della parola d'ordine.

La stessa chiave di codifica della parola d'ordine viene utilizzata per codificare e decodificare tutte le parole d'ordine memorizzate per un'istanza di MQIPT. Pertanto, è necessario solo un singolo file di chiavi di codifica della parola d'ordine per ogni installazione di MQIPT .

Se la chiave di codifica della password per un'installazione di MQIPT viene modificata, tutte le password codificate devono essere ricodificate utilizzando la nuova chiave di codifica.

In fase di avvioMQIPT

Il nome predefinito del file della chiave di codifica della password è *MQIPT_HOME_DIR*`/mqipt_cred.key`, dove *MQIPT_HOME_DIR* è la directory in cui è memorizzato il file di configurazione `mqipt.conf` . Se si prevede di eseguire MQIPT come un servizio che viene avviato automaticamente, è necessario creare il file della chiave di codifica della password con il nome predefinito.

Se il file della chiave di codifica della password viene creato con un nome diverso dal nome predefinito, il nome del file deve essere fornito a MQIPT quando viene avviato. Il nome del file della chiave di crittografia della password può essere specificato utilizzando uno dei seguenti metodi, in ordine di preferenza:

1. il parametro **-sf** sul comando **mqipt** utilizzato per avviare MQIPT.
2. la variabile di ambiente `MQS_MQIPTCRED_KEYFILE` .
3. la proprietà di `com.ibm.mq.ipt.cred.keyfile` Java .

Se non viene fornito alcun nome file della chiave di codifica della password, verrà utilizzato il nome file predefinito, se il file esiste. Se il file della chiave di codifica della password predefinita non esiste, viene utilizzata la chiave di codifica della password predefinita.

Amministrazione di MQIPT utilizzando la riga comandi

È possibile utilizzare il comando **mqiptAdmin** sulla riga comandi per gestire MQIPT.

È possibile utilizzare il comando **mqiptAdmin** per eseguire le seguenti funzioni di gestione:

- Elencare le istanze locali attive di MQIPT.
- Aggiornare un'istanza di MQIPT dopo aver apportato le modifiche al file di configurazione.
- Arrestare un'istanza di MQIPT.

Il comando `mqiptAdmin` si trova nella sottodirectory `bin` della directory di installazione di MQIPT .

Il comando **mqiptAdmin** si connette all'istanza attiva di MQIPT da gestire utilizzando uno dei seguenti metodi:

- creando una connessione di rete a una porta comandi.
- collegandosi a un'istanza locale di MQIPT senza utilizzare la porta comandi.

Il comando **mqiptAdmin** è compatibile con le versioni precedenti di MQIPT, ma non è possibile utilizzare il comando per gestire le versioni di MQIPT che sono di una versione superiore rispetto alla versione del comando **mqiptAdmin** . In un ambiente che include diverse versioni di MQIPT, è necessario utilizzare la versione più recente del comando **mqiptAdmin** .

Per ulteriori informazioni relative alla sintassi del comando **mqiptAdmin** , consultare mqiptAdmin (amministrare MQIPT).

Gestione locale senza una porta comandi

Le istanze locali di MQIPT possono essere gestite senza utilizzare una porta comandi. L'amministrazione locale consente di gestire MQIPT utilizzando il comando **mqiptAdmin** solo quando viene eseguita nello stesso sistema dell'istanza MQIPT che si desidera amministrare.

Affinché **mqiptAdmin** sia autorizzato a gestire un'istanza locale di MQIPT senza utilizzare la porta comandi, l'istanza MQIPT deve essere in esecuzione sullo stesso sistema e con lo stesso ID utente di **mqiptAdmin**. In alternativa, su AIX and Linux, **mqiptAdmin** può essere eseguito come `root`.

L'amministrazione locale è abilitata per impostazione predefinita. Per disabilitare l'amministrazione locale, utilizzare la proprietà di configurazione **LocalAdmin** . Per ulteriori informazioni sulla proprietà **LocalAdmin** , consultare LocalAdmin.

Per gestire le istanze locali di MQIPT, è necessario assegnare un nome a ciascuna istanza. È possibile assegnare un nome a un'istanza di MQIPT utilizzando il parametro **-n** quando si avvia MQIPT con il comando **mqipt** . Se non si specifica un nome all'avvio di MQIPT, il nome della directory home viene utilizzato come nome dell'istanza MQIPT . Ad esempio, il seguente comando avvia MQIPT e assegna il nome `ipt1` all'istanza:

```
mqipt /opt/mqipt1 -n ipt1
```

Una volta che l'istanza ha un nome, è possibile gestire tale istanza specificando il nome nel comando **mqiptAdmin** con il parametro **-n** . Ad esempio, il seguente comando arresta l'istanza locale di MQIPT con nome ipt1:

```
mqiptAdmin -stop -n ipt1
```

Puoi elencare tutte le istanze attive locali di MQIPT che il comando **mqiptAdmin** è autorizzato a gestire senza utilizzare una porta comandi utilizzando il comando **mqiptAdmin** con il parametro **-list** . Ad esempio, il seguente comando elenca tutte le istanze attive locali di MQIPT che l'utente che ha avviato il comando **mqiptAdmin** è autorizzato a gestire:

```
mqiptAdmin -list
```

Gestione mediante una porta comandi

È possibile configurare MQIPT con una porta comandi non protetta e una porta comandi protetta con TLS. È possibile utilizzare queste porte comandi per gestire MQIPT come qualsiasi utente che si trova sullo stesso sistema dell'istanza MQIPT che si desidera gestire o da un sistema remoto.

Le versioni precedenti di MQIPT accettarono solo i comandi di gestione emessi sulla porta comandi non protetta.

Nota: Le connessioni alla porta comandi non protetta non vengono codificate, pertanto i dati inviati sulla rete alla porta comandi non protetta, inclusa la password di accesso MQIPT , possono essere visualizzati da altri utenti sulla rete.

Affinché MQIPT sia in ascolto su una porta comandi per i comandi emessi dal comando **mqiptAdmin** , è necessario specificare un valore per le proprietà **CommandPort** o **SSLCommandPort** nella sezione globale del file di configurazione mqipt.conf .

Esaminare le considerazioni sulla sicurezza in Altre considerazioni sulla sicurezza prima di abilitare una delle porte di comandi MQIPT . Abilitare l'autenticazione per i comandi ricevuti dalle porte dei comandi. Per ulteriori informazioni sull'autenticazione della porta comandi, consultare “Autenticazione porta comandi” a pagina 575.

Per gestire un'istanza di MQIPT utilizzando una porta comandi, specificare l'indirizzo di rete dell'host su cui è in esecuzione MQIPT e il numero di porta comandi, come parametri per il comando **mqiptAdmin** . Ad esempio, per aggiornare l'istanza MQIPT in esecuzione su mqipt.example.come con la porta comandi non protetta configurata per l'ascolto sulla porta 1890, immettere il seguente comando:

```
mqiptAdmin -refresh -r mqipt.example.com:1890
```

Se non si specificano il nome host e il numero di porta, **mqiptAdmin** tenta di connettersi a localhost, porta 1881.

Per ulteriori informazioni sulla gestione di MQIPT utilizzando la porta comandi TLS, consultare “Amministrazione di MQIPT utilizzando la porta comandi TLS” a pagina 571.

Amministrazione di MQIPT utilizzando la porta comandi TLS

MQIPT può essere configurato per utilizzare una porta comandi TLS per ascoltare i comandi di gestione emessi dal comando **mqiptAdmin** . L'uso della porta comandi TLS protegge i dati sensibili come la password di accesso MQIPT sulla rete tra **mqiptAdmin** e MQIPT. Utilizzare questa procedura per configurare la porta comandi TLS e gestire MQIPT utilizzando la porta comandi TLS.

Informazioni su questa attività

La porta comandi TLS deve essere configurata con un certificato server memorizzato in un keystore PKCS #12 o in un hardware crittografico che supporta PKCS #11 Cryptographic Token Interface. Il certificato del server della porta comandi viene inviato al comando **mqiptAdmin** durante l'handshake TLS. Questa attività presuppone che si richieda un nuovo certificato del server da una CA (Certificate Authority) attendibile e che il certificato venga restituito in un file. Il comando **mqiptAdmin** convalida il certificato

della porta comandi utilizzando il certificato CA della CA che ha firmato il certificato server. Il certificato CA deve essere memorizzato in un keystore PKCS #12 a cui il comando **mqiptAdmin** può accedere.

L'autenticazione del certificato del client non è supportata dalla porta comandi TLS. Per abilitare l'autenticazione per i comandi di gestione emessi su una porta comandi, vedere “Autenticazione porta comandi” a pagina 575.

Questa procedura descrive come gestire i keystore e i certificati digitali richiesti per utilizzare la porta comandi TLS utilizzando il comando V 9.4.0 V 9.4.0 **mqiptKeytool** . Per ulteriori informazioni sulla gestione dei keystore utilizzati da MQIPT , consultare Gestione dei keystore MQIPT.

Procedura

1. Seguire questa procedura per configurare la porta comandi TLS per l'istanza di MQIPT.
   a) Creare una coppia di chiavi pubblica e privata e un certificato server porta comandi TLS associato in un keystore PKCS #12 .

   V 9.4.0 V 9.4.0 Per creare il keystore che contiene il certificato server della porta comandi TLS, immettere il comando seguente:

   ```
   mqiptKeytool -genkeypair -keystore filename -storetype pkcs12 -storepass password
                -dname distinguished_name -alias label
                -keyalg key_algorithm -keysize key_size -sigalg sig_algorithm
   ```

   dove:

   -keystore *nomefile*
   Specifica il nome del keystore.

   -storepass *password*
   Specifica la password del keystore.

   -alias *etichetta*
   Specifica l'etichetta del certificato.

   -keyalg *algoritmo_chiave*
   Specifica l'algoritmo utilizzato per creare la coppia di chiavi.

   -keysize *dimensione_chiave*
   Specifica la dimensione della chiave.

   -sigalg *algoritmo*
   Specifica l'algoritmo utilizzato per firmare il certificato.

   -dname *nome_distinto*
   Specifica il nome distinto X.500 racchiuso tra virgolette.

   b) Creare una richiesta di certificato per il certificato del server della porta comandi TLS firmato dalla CA.

   V 9.4.0 V 9.4.0 Per creare una richiesta di certificato, immettere il comando seguente:

   ```
   mqiptKeytool -certreq -keystore filename -storetype pkcs12 -storepass password
               -alias label -file certreq_filename
   ```

   dove:

   -keystore *nomefile*
   Specifica il nome del keystore.

   -storepass *password*
   Specifica la password del keystore.

   -alias *etichetta*
   Specifica l'etichetta del certificato.

   -file *nomefile_certreq*
   Specifica il nome file per la richiesta di certificato.

c) Inviare il file di richiesta di certificato creato nel passo “1.b” a pagina 572 alla propria CA da firmare.

d) Dopo che la CA invia il certificato firmato, ricevere il certificato firmato nel keystore.

V 9.4.0 V 9.4.0 Per ricevere il certificato firmato nel keystore, immettere il comando seguente:

```
mqiptKeytool -importcert -keystore cert_filename -storetype pkcs12 -storepass password
             -file cert_filename
```

dove *nomefile_cert* è il nome del file che contiene il certificato, *nomefile* è il nome del keystore e *password* è la password del keystore.

e) Codificare la password del keystore utilizzando il comando **mqiptPW** .

Immettere il seguente comando:

```
mqiptPW -sf encryption_key_file
```

dove *encryption_key_file* è il nome di un file che contiene la chiave di codifica della password per l'installazione di MQIPT . Non è necessario specificare il parametro **-sf** se l'installazione di MQIPT utilizza la chiave di codifica della password predefinita. Immettere la password del keystore da codificare quando richiesto.

Per ulteriori informazioni sul comando **mqiptPW** , consultare Codifica di una password key ring.

f) Modificare il file di configurazione `mqipt.conf` e specificare le seguenti proprietà per configurare la porta comandi TLS:

i) Impostare il valore della proprietà **SSLCommandPort** sul numero di porta del comando TLS.

ii) Impostare il valore della proprietà **SSLCommandPortKeyRing** sul nome file del keystore creato nel passo “1.a” a pagina 572.

iii) Impostare il valore di **SSLCommandPortKeyRingPW** sull'output della stringa mediante il comando **mqiptPW** nel passo “1.e” a pagina 573.

iv) Impostare il valore della proprietà **SSLCommandPortSiteLabel** sul nome etichetta del certificato della porta comandi TLS, specificato durante la creazione della richiesta di certificato nel passo “1.b” a pagina 572.

v) Se si desidera limitare le connessioni in entrata alla porta comandi TLS a quelle provenienti da una particolare interfaccia di rete, impostare il valore della proprietà **SSLCommandPortListenerAddress** su un indirizzo di rete appartenente a una delle interfacce di rete sul sistema su cui è in esecuzione MQIPT . Ad esempio, per limitare le connessioni in entrata alla porta comandi TLS solo a quelle provenienti dalla macchina locale, impostare il valore della proprietà **SSLCommandPortListenerAddress** su `localhost`.

g) Avviare o aggiornare MQIPT per abilitare la porta comandi TLS.

MQIPT emette messaggi della console come quelli riportati di seguito per visualizzare la configurazione della porta comandi TLS in vigore:

```
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. Sul sistema in cui il comando **mqiptAdmin** viene utilizzato per amministrare MQIPT, attenersi alla seguente procedura per consentire a **mqiptAdmin** di connettersi alla porta comandi TLS.

a) Importare il certificato CA della CA che ha firmato il certificato della porta comandi TLS in un keystore PKCS #12 da utilizzare come truststore dal comando **mqiptAdmin** .

V 9.4.0 V 9.4.0 Per importare il certificato CA, immettere il comando seguente:

```
mqiptKeytool -importcert -keystore filename -storetype pkcs12 -storepass password
             -file cert_filename -alias certlabel
```

dove:

nome file
Specifica il nome del keystore da creare

password
Specifica la password del keystore

etichetta certificato
Specifica l'etichetta da fornire al certificato CA

nomefile_cert
Specifica il nome del file che contiene il certificato CA

b) Codificare la password del keystore utilizzando il comando **mqiptPW** .

Immettere il seguente comando:

```
mqiptPW -sf encryption_key_file
```

dove *encryption_key_file* è il nome del file che contiene la chiave di codifica della password. Il file della chiave di codifica della parola d'ordine può essere diverso da quello utilizzato per codificare le password nella configurazione MQIPT . La chiave di codifica della password predefinita viene utilizzata se non si specifica un file della chiave di codifica con il parametro **-sf** . Immettere la password del keystore da codificare quando richiesto.

Per ulteriori informazioni sul comando **mqiptPW** , consultare Codifica di una password key ring.

c) Creare un file delle proprietà che deve essere utilizzato dal comando **mqiptAdmin** e specificare le seguenti proprietà:

```
SSLClientCAKeyRing=key_ring_file_name
SSLClientCAKeyRingPW=key_ring_password
PasswordProtectionKeyFile=encryption_key_file
```

dove:

nome_file_ring_chiavi
è il nome del keystore creato nel passo “2.a” a pagina 573.

password_ring_chiave
è l'emissione della password codificata dal comando **mqiptPW** nel passo “2.b” a pagina 574.

ENCRYPTION_KEY_FILE
è il nome del file che contiene la chiave di codifica password. È necessario specificare la proprietà **PasswordProtectionKeyFile** solo se è stato utilizzato un file della chiave di codifica per codificare la password del keystore nel passo “2.b” a pagina 574.

d) Immettere il comando **mqiptAdmin** per gestire MQIPT, specificando il parametro **-s** per indicare che è richiesta una connessione TLS e il parametro **-p** per specificare il nome del file delle proprietà creato nel passo “2.c” a pagina 574.

Ad esempio, immetti il seguente comando per aggiornare un'istanza di MQIPT inviando un comando di aggiornamento alla porta comandi TLS:

```
mqiptAdmin -refresh -r hostname:port -s -p properties_file
```

Il comando **mqiptAdmin** emette un messaggio come il seguente per confermare che la connessione a MQIPT è protetta con TLS:

```
MQCAI109 The connection to MQIPT is secured with TLSv1.2.
```

Operazioni successive

Per abilitare l'autenticazione per i comandi ricevuti dalla porta comandi TLS, seguire la procedura in “Autenticazione porta comandi” a pagina 575.

Autenticazione porta comandi

MQIPT può essere configurato per autenticare i comandi ricevuti dalla porta comandi non protetta e dalla porta comandi TLS utilizzando una password. Utilizzare questa procedura per abilitare l'autenticazione della porta comandi.

Informazioni su questa attività

Il comando **mqiptAdmin** richiede agli utenti di immettere una parola d'ordine quando il comando si connette alla porta comandi di una istanza di MQIPT con l'autenticazione della porta comandi abilitata. MQIPT convalida la password immessa nel comando **mqiptAdmin** rispetto alla password di accesso specificata nella configurazione di MQIPT .

Le proprietà impostate per l'autenticazione della porta comandi si applicano sia alla porta comandi TLS che alla porta comandi non protetta.

Procedura

1. Codificare la password di accesso MQIPT utilizzando il comando **mqiptPW** .

   Immettere il seguente comando:

   ```
   mqiptPW -sf encryption_key_file
   ```

   dove *encryption_key_file* è il nome del file che contiene la chiave di codifica della password per l'installazione di MQIPT . Non è necessario specificare il parametro **-sf** se l'installazione di MQIPT utilizza la chiave di codifica della password predefinita. Immettere la password di accesso da codificare quando richiesto.

   Per ulteriori informazioni sulla codifica delle password nella configurazione di MQIPT , vedi Codifica delle password memorizzate.
2. Modificare il file di configurazione `mqipt.conf` e specificare le seguenti proprietà:

   ```
   AccessPW=encrypted_password
   RemoteCommandAuthentication=auth_setting
   ```

   dove:

   password_codificata
   è l'emissione della password codificata dal comando **mqiptPW** nel passo “1” a pagina 575.

   auth_setting
   è il requisito di autenticazione. L'autenticazione della porta comandi è abilitata se questa proprietà è impostata su uno dei seguenti valori:

   facoltativo
   Una password non è richiesta, ma se viene fornita deve essere valida. Questa opzione potrebbe essere utile durante la migrazione, ad esempio.

   obbligatorio
   È necessario fornire una password valida con ogni comando ricevuto da una porta comandi.

   Per ulteriori informazioni su queste proprietà, consultare MQIPT proprietà globali.
3. Avviare o aggiornare MQIPT per rendere effettive le modifiche.

   MQIPT emette un messaggio che indica se l'autenticazione della porta comandi è abilitata. Ad esempio, se MQIPT è configurato per richiedere l'immissione di una password valida ogni volta che viene eseguito il comando **mqiptAdmin** , viene emesso il seguente messaggio:

   ```
   MQCPI021 Password checking has been enabled on the command port
   ```

Esecuzione di backup

È necessario eseguire il backup di alcuni file MQIPT come parte delle normali procedure di backup.

Eseguire regolarmente il backup dei seguenti file:

- Il file di configurazione, `mqipt.conf`
- I file keyring SSL/TLS specificati dalle seguenti proprietà in `mqipt.conf`:
  - **SSLClientKeyRing**
  - **SSLClientCAKeyRing**
  - **SSLServerKeyRing**
  - **SSLServerCAKeyRing**
  - **SSLCommandPortKeyRing**
- I file di password keyring SSL/TLS specificati dalle proprietà seguenti in `mqipt.conf`:
  - **SSLClientKeyRingPW**
  - **SSLClientCAKeyRingPW**
  - **SSLServerKeyRingPW**
  - **SSLServerCAKeyRingPW**
- Il file della chiave di codifica della password, se la configurazione MQIPT contiene password codificate con una chiave di codifica diversa da quella predefinita.
- Il file della politica specificato da **SecurityManagerPolicy**, se tale proprietà è stata impostata.
- I file di uscita di sicurezza e i file di uscita del certificato specificati dalle proprietà riportate di seguito in `mqipt.conf`:
  - **SecurityExitName**
  - **SSLExitName**
- I file di log della connessione nella sottodirectory `log` della directory home MQIPT , se sono necessari per scopi di controllo.

Ottimizzazione delle prestazioni

È possibile ottimizzare le prestazioni relative di ciascun instradamento MQIPT utilizzando una combinazione di un pool di thread e una specifica di timeout di inattività.

Thread di connessione

A ciascun instradamento MQIPT viene assegnato un lotto di lavoro di thread in esecuzione simultanea che gestiscono richieste di comunicazione in entrata. All'inizializzazione, viene creato un lotto di thread (della dimensione specificata nell'attributo `MinConnectionThreads` dell'instradamento) e viene assegnato un thread per gestire la prima richiesta in entrata. Quando arriva questa richiesta, viene assegnato un altro thread, pronto per la successiva richiesta in entrata. Quando tutti i thread vengono assegnati per il lavoro, viene creato un nuovo thread, aggiunto al lotto di lavoro e assegnato per il lavoro.

In questo modo, il pool cresce fino a quando non viene raggiunto il numero massimo di thread (specificato in **MaxConnectionThreads**). I thread vengono rilasciati di nuovo nel lotto quando termina una conversazione o quando è trascorso il periodo di timeout di inattività specificato. Quando viene raggiunto il numero massimo di sottoprocessi di lavoro, la successiva richiesta in entrata attende che un sottoprocesso venga rilasciato nuovamente al lotto di lavoro.

È possibile ridurre il tempo di attesa delle richieste aumentando il numero di thread disponibili. Tuttavia, è necessario bilanciare questo aumento con le risorse di sistema disponibili.

Timeout di inattività

Per impostazione predefinita, i thread di lavoro non vengono terminati a causa di inattività. Quando un thread è stato assegnato a una conversazione, rimane assegnato a tale conversazione fino a quando non

viene chiuso normalmente, l'instradamento viene disattivato o MQIPT viene arrestato. Facoltativamente, è possibile specificare un intervallo di timeout di inattività (in minuti) nella proprietà **IdleTimeout** in modo che i thread che sono stati inattivi per il periodo di tempo specificato vengano riciclati. I thread vengono riciclati per essere utilizzati rimettendoli nel pool di lavoro.

Se l'attività IBM MQ è intermittente, impostare il relativo intervallo di heartbeat su un valore inferiore a quello del timeout MQIPT in modo che i thread non vengano costantemente riciclati.

Informazioni particolari

Queste informazioni sono state sviluppate per prodotti e servizi offerti negli Stati Uniti.

IBM potrebbe non offrire i prodotti, i servizi o le funzioni descritti in questo documento in altri paesi. Consultare il rappresentante IBM locale per informazioni sui prodotti e sui servizi disponibili nel proprio paese. Ogni riferimento relativo a prodotti, programmi o servizi IBM non implica che solo quei prodotti, programmi o servizi IBM possano essere utilizzati. In sostituzione a quelli forniti da IBM possono essere usati prodotti, programmi o servizi funzionalmente equivalenti che non comportino la violazione dei diritti di proprietà intellettuale o di altri diritti dell'IBM. Tuttavia, è responsabilità dell'utente valutare e verificare il funzionamento di qualsiasi prodotto, programma o servizio non IBM.

IBM potrebbe disporre di applicazioni di brevetti o brevetti in corso relativi all'argomento descritto in questo documento. La fornitura di tale documento non concede alcuna licenza a tali brevetti. Chi desiderasse ricevere informazioni relative a licenze può rivolgersi per iscritto a:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Per richieste di licenze relative ad informazioni double-byte (DBCS), contattare il Dipartimento di Proprietà Intellettuale IBM nel proprio paese o inviare richieste per iscritto a:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

Il seguente paragrafo non si applica al Regno Unito o a qualunque altro paese in cui tali dichiarazioni sono incompatibili con le norme locali: INTERNATIONAL BUSINESS MACHINES CORPORATION FORNISCE LA PRESENTE PUBBLICAZIONE "NELLO STATO IN CUI SI TROVA" SENZA GARANZIE DI ALCUN TIPO, ESPRESSE O IMPLICITE, IVI INCLUSE, A TITOLO DI ESEMPIO, GARANZIE IMPLICITE DI NON VIOLAZIONE, DI COMMERCIABILITÀ E DI IDONEITÀ PER UNO SCOPO PARTICOLARE. Alcuni stati non consentono la rinuncia a garanzie esplicite o implicite in determinate transazioni; quindi la presente dichiarazione potrebbe non essere applicabile.

Questa pubblicazione potrebbe contenere imprecisioni tecniche o errori tipografici. Le informazioni incluse in questo documento vengono modificate su base periodica; tali modifiche vengono incorporate nelle nuove edizioni della pubblicazione. IBM si riserva il diritto di apportare miglioramenti o modifiche al prodotto/i e/o al programma/i descritti nella pubblicazione in qualsiasi momento e senza preavviso.

Tutti i riferimenti a siti Web non dell'IBM contenuti in questo documento sono forniti solo per consultazione e non rappresenta in alcun modo un'approvazione di tali siti. I materiali reperibili in tali siti Web non fanno parte dei materiali relativi a questo prodotto IBM e l'utilizzo di tali siti è responsabilità dell'utente.

Tutti i commenti e i suggerimenti inviati potranno essere utilizzati liberamente da IBM e diventeranno esclusiva della stessa.

Coloro che detengono la licenza su questo programma e desiderano avere informazioni su di esso allo scopo di consentire (i) uno scambio di informazioni tra programmi indipendenti ed altri (compreso questo) e (ii) l'uso reciproco di tali informazioni, dovrebbero rivolgersi a:

IBM Corporation
Coordinatore interoperabilità software, Dipartimento 49XA
Autostrada 3605 52 N

Rochester, MN 55901
U.S.A.

Queste informazioni possono essere rese disponibili secondo condizioni contrattuali appropriate, compreso, in alcuni casi, il pagamento di un addebito.

Il programma su licenza descritto in queste informazioni e tutto il materiale su licenza disponibile per esso sono forniti da IBM in base ai termini dell' IBM Customer Agreement, IBM International Program License Agreement o qualsiasi altro accordo equivalente tra le parti.

Tutti i dati relativi alle prestazioni contenuti in questo documento sono stati determinati in un ambiente controllato. Pertanto, i risultati ottenuti in altri ambienti operativi possono variare in modo significativo. Alcune misurazioni potrebbero essere state fatte su sistemi a livello di sviluppo e non vi è alcuna garanzia che queste misurazioni saranno le stesse sui sistemi generalmente disponibili. Inoltre, alcune misurazioni potrebbero essere state stimate mediante estrapolazione. I risultati quindi possono variare. Gli utenti di questo documento dovrebbero verificare i dati applicabili per il loro ambiente specifico.

Le informazioni relative a prodotti non IBM provengono dai fornitori di tali prodotti, dagli annunci pubblicati o da altre fonti pubblicamente disponibili. IBM non ha verificato tali prodotti e, pertanto, non può garantirne l'accuratezza delle prestazioni. Eventuali commenti relativi alle prestazioni dei prodotti non IBM devono essere indirizzati ai fornitori di tali prodotti.

Tutte le dichiarazioni riguardanti la direzione o l'intento futuro di IBM sono soggette a modifica o ritiro senza preavviso e rappresentano solo scopi e obiettivi.

Questa pubblicazione contiene esempi di dati e prospetti utilizzati quotidianamente nelle operazioni aziendali, Per poterli illustrare nel modo più completo possibile, gli esempi riportano nomi di persone, società, marchi e prodotti. Tutti questi nomi sono fittizi e qualsiasi somiglianza con nomi ed indirizzi adoperati da imprese realmente esistenti sono una mera coincidenza.

LICENZA SUL COPYRIGHT:

Queste informazioni contengono programmi applicativi di esempio in lingua originale, che illustrano le tecniche di programmazione su diverse piattaforme operative. È possibile copiare, modificare e distribuire questi programmi di esempio sotto qualsiasi forma senza alcun pagamento alla IBM, allo scopo di sviluppare, utilizzare, commercializzare o distribuire i programmi applicativi in conformità alle API (application programming interface) a seconda della piattaforma operativa per cui i programmi di esempio sono stati scritti. Questi esempi non sono stati testati approfonditamente tenendo conto di tutte le condizioni possibili. IBM, quindi, non può garantire o sottintendere l'affidabilità, l'utilità o il funzionamento di questi programmi.

Se si sta visualizzando queste informazioni in formato elettronico, le fotografie e le illustrazioni a colori potrebbero non apparire.

Informazioni sull'interfaccia di programmazione

Le informazioni sull'interfaccia di programmazione, se fornite, consentono di creare software applicativo da utilizzare con questo programma.

Questo manuale contiene informazioni sulle interfacce di programmazione che consentono al cliente di scrivere programmi per ottenere i servizi di IBM MQ.

Queste informazioni, tuttavia, possono contenere diagnosi, modifica e regolazione delle informazioni. La diagnosi, la modifica e la regolazione delle informazioni vengono fornite per consentire il debug del software applicativo.

Importante: Non utilizzare queste informazioni di diagnosi, modifica e ottimizzazione come interfaccia di programmazione poiché sono soggette a modifica.

Marchi

IBM, il logo IBM , ibm.com, sono marchi di IBM Corporation, registrati in molte giurisdizioni nel mondo. Un elenco aggiornato dei marchi IBM è disponibile sul web in "Copyright and trademark

information"www.ibm.com/legal/copytrade.shtml. Altri nomi di prodotti e servizi potrebbero essere marchi di IBM o altre società.

Microsoft e Windows sono marchi di Microsoft Corporation negli Stati Uniti, in altri paesi o entrambi.

UNIX è un marchio registrato di The Open Group negli Stati Uniti e/o in altri paesi.

Linux è un marchi registrato di Linus Torvalds negli Stati Uniti e/o in altri paesi.

Questo prodotto include il software sviluppato da Eclipse Project (https://www.eclipse.org/).

Java e tutti i marchi e i logo Java sono marchi registrati di Oracle e/o di società affiliate.

Numero parte:

(1P) P/N: