9.3

# *IBM MQ -Technische Übersicht*

**Hinweis**

Vor Verwendung dieser Informationen und des darin beschriebenen Produkts sollten die Informationen unter „Bemerkungen” auf Seite 339 gelesen werden.

Diese Ausgabe bezieht sich auf Version 9 Release 3 von IBM® MQ und alle nachfolgenden Releases und Modifikationen, bis dieser Hinweis in einer Neuausgabe geändert wird.

Wenn Sie Informationen an IBMsenden, erteilen Sie IBM ein nicht ausschließliches Recht, die Informationen in beliebiger Weise zu verwenden oder zu verteilen, ohne dass eine Verpflichtung für Sie entsteht.

# Inhaltsverzeichnis

# IBM MQ - Technische Übersicht

Mit IBM MQ können Sie Verbindungen für Ihre Anwendungen herstellen und die Verteilung von Informationen in Ihrem Unternehmen verwalten.

IBM MQ ermöglicht Programmen, über ein Netz unähnlicher Komponenten (Prozessoren, Betriebssysteme, Subsysteme und Übertragungsprotokolle) mithilfe einer konsistenten Anwendungsprogrammierschnittstelle miteinander zu kommunizieren. Anwendungen, die mit dieser Schnittstelle entworfen und geschrieben wurden, werden als Message-Queuing-Anwendungen bezeichnet (Anwendungen zur Steuerung von Nachrichtenwarteschlangen).

In den folgenden Abschnitten erhalten Sie Informationen zur Steuerung von Nachrichtenwarteschlangen und anderen Features von IBM MQ.

**Zugehörige Konzepte**

Einführung in IBM MQ

Informationen zu Produktanforderungen und zum Support

**Zugehörige Tasks**

IBM MQ-Architektur planen

**Zugehörige Verweise**

„Hauptmerkmale und Vorteile des Message-Queuing” auf Seite 7

In diesem Abschnitt werden einige Merkmale und Vorteile des Message-Queuing hervorgehoben. Sie finden hier beispielsweise eine Beschreibung der Message-Queuing-Merkmale in Bezug auf Sicherheit und Datenintegrität.

## Einführung in die Nachrichtenwarteschlangensteuerung

Mit den IBM MQ-Produkten können Programme über ein Netz ungleicher Komponenten (Prozessoren, Betriebssysteme, Subsysteme und Kommunikationsprotokolle) unter Verwendung einer konsistenten Anwendungsprogrammierschnittstelle miteinander kommunizieren.

Anwendungen, die über diese Schnittstelle entwickelt und geschrieben wurden, werden als *Message-Queuing*-Anwendungen bezeichnet, da sie *Messaging* und *Queuing* (Warteschlangensteuerung) verwenden:

- Messaging bedeutet, dass Programme miteinander kommunizieren, indem sie sich gegenseitig Daten in Nachrichten zusenden, anstatt sich direkt anzurufen.
- Eine Warteschlangensteuerung bedeutet, dass Nachrichten in Warteschlangen gespeichert werden. Programme können somit unabhängig voneinander mit unterschiedlicher Verarbeitungsgeschwindigkeit und zu unterschiedlichen Zeiten an verschiedenen Standorten ausgeführt werden, ohne dass eine logische Verbindung zwischen ihnen bestehen muss.

Message-Queuing ist in der Datenverarbeitung seit vielen Jahren im Einsatz. Heutzutage wird es im Allgemeinen in E-Mails verwendet. Ohne Queuing (Warteschlangensteuerung) muss beim Senden einer elektronischen Nachricht über weite Entfernungen jeder einzelne Knoten auf der Route zur Weiterleitung von Nachrichten verfügbar sein. Außerdem müssen die Empfänger angemeldet sein und wissen, dass Sie ihnen eine Nachricht senden möchten. In einem Warteschlangensystem werden die Nachrichten an Zwischenknoten gespeichert, bis das System zu ihrer Weiterleitung bereit ist. Am Zielort werden sie in einer elektronischen Mailbox gespeichert, bis der Empfänger bereit ist, sie zu lesen.

Trotz alledem werden viele komplexe Geschäftstransaktionen heutzutage ohne Queuing verarbeitet. In einem großen Netz kann das System möglicherweise viele Tausende Verbindungen sofort einsatzfähig halten. Tritt in einem Teil des Systems ein Problem auf, können viele Systemteile nicht mehr eingesetzt werden.

Sie können sich Message-Queuing als E-Mail für Programme vorstellen. In einer Message-Queuing-Umgebung führt jedes Programm einer Anwendungssuite eine klar strukturierte eigenständige Funktion als Antwort auf eine bestimmte Anforderung aus. Für die Kommunikation mit einem anderen Programm

muss ein Programm eine Nachricht in eine vordefinierte Warteschlange einreihen. Diese Nachricht wird von dem anderen Programm aus der Warteschlange abgerufen und die Anforderungen und Angaben aus der Nachricht werden verarbeitet. Message-Queuing stellt somit eine Art der Kommunikation zwischen Programmen dar.

Queuing bezeichnet den Mechanismus, mit dem Nachrichten zurückbehalten werden, bis sie von einer Anwendung verarbeitet werden können. Queuing bietet folgende Möglichkeiten:

- Kommunikation zwischen Programmen (die sogar in verschiedenen Umgebungen ausgeführt werden können), ohne dass Kommunikationscode geschrieben werden muss.
- Auswahl der Reihenfolge, in welcher ein Programm Nachrichten verarbeitet.
- Ausgleich der Arbeitslast in einem System durch Festlegung mehrerer Programme, die für eine Warteschlange zuständig sind, wenn die Anzahl der Nachrichten einen Grenzwert überschreitet.
- Erhöhte Verfügbarkeit der Anwendungen, durch Festlegung eines alternativ für die Warteschlangen zuständigen Systems, wenn das primäre System nicht verfügbar ist.

## Was ist unter einer Nachrichtenwarteschlange zu verstehen?

Bei einer Nachrichtenwarteschlange, die auch einfach als Warteschlange bezeichnet wird, handelt es sich um ein benanntes Ziel, an das Nachrichten gesendet werden können. Nachrichten sammeln sich in Warteschlangen an, bis sie von Programmen abgerufen werden, die für diese Warteschlangen zuständig sind.

Warteschlangen befinden sich auf einem Warteschlangenmanager, von dem sie auch verwaltet werden (siehe Abschnitt „Terminologie zum Message-Queuing" auf Seite 9). Die physische Natur einer Warteschlange hängt von dem Betriebssystem ab, unter dem der Warteschlangenmanager ausgeführt wird. Warteschlangen können flüchtige Pufferbereiche im Hauptspeicher eines Computers oder aber Datenmengen in einer permanenten Speichereinheit (wie z. B. einer Festplatte) sein. Die physische Verwaltung der Warteschlangen ist Aufgabe der Warteschlangenmanager, für die beteiligten Anwendungsprogramme ist sie nicht offensichtlich.

Programme greifen auf Warteschlangen lediglich über die externen Services des Warteschlangenmanagers zu. Sie können eine Warteschlange öffnen, Nachrichten darin einreihen und daraus abrufen und die Warteschlange schließen. Außerdem können sie die Warteschlangenattribute festlegen und abfragen.

## Verschiedene Arten des Message-Queuing

### Punkt-zu-Punkt

Eine Nachricht wird in die Warteschlange eingereiht und dann von einer Anwendung empfangen.

Beim Punkt-zu-Punkt-Messaging muss eine sendende Anwendung Informationen zu einer anderen empfangenden Anwendung erhalten, damit sie eine Nachricht an diese Anwendung senden kann. Die sendende Anwendung muss beispielsweise den Namen der Warteschlange kennen, an die die Informationen gesendet werden sollen, und möglicherweise muss auch der Name eines Warteschlangenmanagers angegeben werden.

### Publish/Subscribe

Jeder interessierten Anwendung wird eine Kopie jeder Nachricht bereitgestellt, die von einer publizierenden Anwendung veröffentlicht wird. Dabei ist die Anzahl der interessierten Anwendungen - keine, eine oder mehrere - unbedeutend. Beim Publish/Subscribe wird eine interessierte Anwendung als Subskribent bezeichnet, und die Nachrichten werden in eine durch eine Subskription festgelegte Warteschlange eingereiht.

Mithilfe des Publish/Subscribe-Messaging kann der Informationsanbieter von den Nutzern der Informationen abgekoppelt werden. Die Sender- und Empfängeranwendungen müssen zur Übertragung der Informationen nichts übereinander wissen. Weitere Informationen finden Sie unter „Publish/Subscribe-Messaging" auf Seite 67.

### Vorteile des Message-Queuing für Anwendungsentwickler

In IBM MQ können Anwendungsprogramme *Message-Queuing* einsetzen und somit an einer nachrichtengesteuerten Verarbeitung teilnehmen. Unter Verwendung der entsprechenden Message-Queuing-Softwareprodukte können Anwendungsprogramme über verschiedene Plattformen hinweg miteinander kommunizieren. Beispielsweise können z/OS-Anwendungen über IBM MQ for z/OS kommunizieren. Die Anwendungen sind von den Mechanismen der zugrunde liegenden Kommunikation abgeschirmt. Einige weitere Vorteile des Message-Queuing:

- Es ist möglich, für die Anwendungsentwicklung kleine Programme einzusetzen, die von vielen Anwendungen gemeinsam genutzt werden können.
- Durch die Wiederverwendung dieser logischen Bausteine können schnell neue Anwendungen erstellt werden.
- Anwendungen, die für die Verwendung von Message-Queuing-Verfahren geschrieben wurden, sind von Änderungen in der Funktionsweise der Warteschlangenmanager nicht betroffen.
- Es werden keine Kommunikationsprotokolle benötigt. Alle Aspekte der Kommunikation werden vom Warteschlangenmanager übernommen.
- Programme, die Nachrichten empfangen, müssen nicht aktiv sein, wenn ihnen Nachrichten zugesendet werden. Die Nachrichten verbleiben in Warteschlangen.

Geringere Kosten für die Anwendungen, da eine schnellere Entwicklung mit weniger Entwicklern möglich ist und weniger Programmierkenntnisse erforderlich sind als für Anwendungen, die Message-Queuing nicht einsetzen.

Überall, wo die Anwendungen ausgeführt werden, implementiert IBM MQ eine einheitliche Anwendungsprogrammierschnittstelle, das sogenannte *Message Queue Interface* (MQI). Auf diese Weise können Anwendungsprogramme leichter auf andere Plattformen portiert werden.

Details zum Message Queue Interface finden Sie im Abschnitt Übersicht über das Message Queue Interface.

## Hauptmerkmale und Vorteile des Message-Queuing

In diesem Abschnitt werden einige Merkmale und Vorteile des Message-Queuing hervorgehoben. Sie finden hier beispielsweise eine Beschreibung der Message-Queuing-Merkmale in Bezug auf Sicherheit und Datenintegrität.

Anwendungen, die Message-Queuing-Verfahren einsetzen, weisen folgende Hauptmerkmale auf:


- „Keine Direktverbindungen zwischen Programmen” auf Seite 7
- „Zeitunabhängige Kommunikation” auf Seite 8
- „Kleine Programme” auf Seite 8
- „Nachrichtengesteuerte Verarbeitung” auf Seite 9
- „Ereignisgesteuerte Verarbeitung” auf Seite 9
- „Nachrichtenpriorität” auf Seite 9
- „Sicherheit” auf Seite 9
- „Datenintegrität” auf Seite 9
- „Unterstützung für die Wiederherstellung” auf Seite 9


**Anmerkung:** Für IBM MQ-Clients und -Server gilt Folgendes: Es ist keine Änderung einer Serveranwendung erforderlich, damit weitere IBM MQ MQI clients auf neuen Plattformen unterstützt werden können. Entsprechend kann der IBM MQ MQI client ohne Änderung mit weiteren Servertypen eingesetzt werden.

### Keine Direktverbindungen zwischen Programmen

Message-Queuing stellt ein Verfahren für die indirekte Kommunikation zwischen Programmen dar. Dieses Verfahren kann in jeder Anwendung eingesetzt werden, in der Programme miteinander kommunizieren. Die Kommunikation erfolgt, indem ein Programm Nachrichten in eine (einem Warteschlangenmanager

zugehörige) Warteschlange einreiht und ein anderes Programm die Nachrichten aus der Warteschlange abruft.

Programme können Nachrichten abrufen, die von anderen Programmen in eine Warteschlange eingereiht wurden. Die anderen Programme können mit demselben Warteschlangenmanager wie das empfangende Programm oder aber mit einem anderen Warteschlangenmanager verbunden sein. Dieser andere Warteschlangenmanager kann sich auf einem anderen System, in einem anderen Computersystem oder sogar in einem anderen Unternehmen befinden.

Zwischen Programmen, die Nachrichtenwarteschlangen für die Kommunikation miteinander einsetzen, bestehen keine physischen Verbindungen. Ein Programm sendet Nachrichten an eine Warteschlange eines Warteschlangenmanagers und ein anderes Programm ruft Nachrichten aus der Warteschlange ab (siehe Abschnitt Abbildung 1 auf Seite 8).

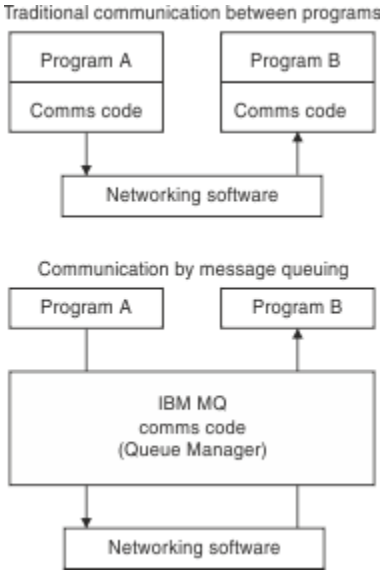



*Abbildung 1. Message-Queuing im Vergleich zur konventionellen Kommunikation*

Wie bei E-Mail durchqueren die einzelnen zu einer Transaktion gehörigen Nachrichten ein Netz auf Basis des Store-and-forward-Verfahrens. Schlägt eine Verbindung zwischen Knoten fehl, bleibt die Nachricht erhalten, bis die Verbindung wiederhergestellt ist oder der Bediener bzw. das Programm die Nachricht umleitet.

Der Mechanismus, mit dem Nachrichten von einer Warteschlange zur nächsten gelangen, bleibt vor den Programmen verborgen. Dies vereinfacht die Programme.

## Zeitunabhängige Kommunikation

Programme, die andere Programme zur Ausführung von Arbeiten auffordern, müssen nicht auf die Antwort zu der Anforderung warten. Sie können weitere Arbeiten erledigen und die Antwort bei Erhalt oder zu einem späteren Zeitpunkt verarbeiten. Beim Schreiben einer Messaging-Anwendung ist es irrelevant, wann ein Programm eine Nachricht sendet bzw. wann das Ziel die Nachricht empfangen kann. Die Nachricht geht nicht verloren, sie wird vom Warteschlangenmanager beibehalten, bis sie vom Ziel verarbeitet werden kann. Die Nachricht bleibt in der Warteschlange, bis sie von einem Programm daraus entfernt wird. Die sendende und die empfangende Anwendung sind also voneinander abgekoppelt. Der Sender kann seine Verarbeitung fortsetzen und muss nicht warten, bis der Empfänger den Erhalt der Nachricht bestätigt. Die Zielanwendung muss nicht einmal aktiv sein, wenn die Nachricht gesendet wird. Sie kann die Nachricht nach dem Start abrufen.

## Kleine Programme

Bei Einsatz des Message-Queuing können Sie die Vorteile kleiner unabhängiger Programme nutzen. Anstelle eines einzelnen umfangreichen Programms, das nacheinander alle Teile eines Jobs ausführt,

kann der Job auf mehrere kleinere unabhängige Programme verteilt werden. Das aufrufende Programm sendet Nachrichten an jedes einzelne dieser separaten Programme und fordert diese zur Ausführung ihrer jeweiligen Funktion auf. Sobald die einzelnen Programme mit der Verarbeitung fertig sind, werden die Ergebnisse in Form einer oder mehrerer Nachrichten zurückgesendet.

### Nachrichtengesteuerte Verarbeitung

Wenn Nachrichten in einer Warteschlange ankommen, können sie mithilfe der *Auslösefunktion* automatisch eine Anwendung starten. Gegebenenfalls können die Anwendungen nach erfolgter Verarbeitung der Nachricht(en) gestoppt werden.

### Ereignisgesteuerte Verarbeitung

Programme können entsprechend dem Warteschlangenstatus gesteuert werden. So können Sie beispielsweise festlegen, dass bei Eintreffen einer Nachricht in einer Warteschlange ein Programm gestartet werden soll, oder Sie können angeben, dass das Programm erst gestartet werden soll, wenn sich in der Warteschlange beispielsweise 10 Nachrichten über einer gewissen Priorität oder aber mit einer beliebigen Priorität befinden.

### Nachrichtenpriorität

Wenn ein Programm eine Nachricht in eine Warteschlange einreiht, kann es der Nachricht eine Priorität zuordnen. Diese legt fest, an welcher Stelle in der Warteschlange die neue Nachricht hinzugefügt wird.

Programme können entweder die Nachrichten der Reihenfolge nach aus einer Warteschlange abrufen oder aber eine bestimmte Nachricht abrufen. (So könnte es z. B. sein, dass ein Programm eine bestimmte Nachricht abrufen möchte, wenn es auf die Antwort auf eine zuvor gesendete Anforderung wartet.)

### Sicherheit

Für bestimmte Vorgänge stehen bestimmte Sicherheitsfunktionen zur Verfügung. Zum Beispiel die Authentifizierung einer Anwendung beim Zugriff auf einen Warteschlangenmanager, Berechtigungsprüfungen, wenn eine Anwendung Ressourcen wie eine Warteschlange auf einem Warteschlangenmanager verwendet, oder die Verschlüsselung der Nachrichtendaten während der Übertragung im Netz und der Aufbewahrung in Warteschlangen. Weitere Informationen zur Sicherheit finden Sie im Abschnitt Übersicht über die Sicherheit.

### Datenintegrität

Datenintegrität wird über Arbeitseinheiten bereitgestellt. Die Synchronisation des Anfangs und Endes von Arbeitseinheiten wird bei jedem Aufruf MQGET oder MQPUT als Option vollständig unterstützt, sodass die Ergebnisse der Arbeitseinheit festgeschrieben bzw. rückgängig gemacht werden können. Die Synchronisationspunktunterstützung funktioniert entweder IBM MQ-intern oder -extern, je nachdem, welche Art der Synchronisationspunktkoordination für die Anwendung ausgewählt wurde.

### Unterstützung für die Wiederherstellung

Um eine Wiederherstellung zu ermöglichen, werden alle permanenten IBM MQ-Updates protokolliert. Sollte eine Wiederherstellung erforderlich sein, werden alle persistenten Nachrichten zurückgeschrieben, für alle unvollständigen Transaktionen erfolgt ein Rollback und alle Synchronisationspunktfestschreibungen und -Backouts werden nach dem für den zuständigen Synchronisationspunktmanager normalen Verfahren bearbeitet. Weitere Informationen zu persistenten Nachrichten finden Sie im Abschnitt Nachrichtenpersistenz.

## Terminologie zum Message-Queuing

In diesem Abschnitt werden einige Begriffe im Zusammenhang mit Message-Queuing erläutert.

Dazu gehören:

- Kanäle
- Cluster
- IBM MQ MQI client
- z/OS Gruppeninterne Steuerung von Warteschlangen
- Nachricht
- Nachrichtenkanalagent
- Nachrichtendeskriptor
- Punkt-zu-Punkt
- Publish/Subscribe
- Queue
- Queue Manager
- z/OS Gruppe mit gemeinsamer Warteschlange
- z/OS Gemeinsam genutzte Warteschlange
- Subskription
- Thema

## Kanäle

Kanäle werden für die Übertragung von Nachrichten zwischen Warteschlangenmanagern verwendet und schirmen Anwendungen von den verwendeten Kommunikationsprotokollen ab. Die Warteschlangenmanager können sich dabei auf demselben oder einem anderen System auf derselben Plattform oder auf verschiedenen Plattformen befinden. Die gesendeten Nachrichten können aus vielen Quellen stammen:

- Vom Benutzer geschriebene Anwendungsprogramme, die Daten von einem Knoten an einen anderen übertragen
- Vom Benutzer geschriebene Verwaltungsanwendungen, die PCF-Befehle oder die MQAI verwenden
- Die IBM MQ Explorer.
- Warteschlangenmanager, die Instrumentierungsereignisnachrichten an einen anderen Warteschlangenmanager senden
- Warteschlangenmanager, die Fernverwaltungsbefehle an einen anderen Warteschlangenmanager senden (z. B. mit MQSC-Befehlen oder über die administrative REST API)

Weitere Informationen zu Kanälen finden Sie unter „Kanal- definitionen” auf Seite 34.

## Cluster

Ein *Cluster* ist ein Netz von Warteschlangenmanagern, die logisch in gewisser Weise zusammengehören.

In einem IBM MQ-Netz mit verteilter Steuerung von Warteschlangen ohne Clustering ist jeder Warteschlangenmanager unabhängig. Wenn ein Warteschlangenmanager Nachrichten an einen anderen Warteschlangenmanager senden muss, müssen eine Übertragungswarteschlange sowie ein Kanal zu dem fernen Warteschlangenmanager definiert sein.

Für die Verwendung von Clustern gibt es zwei Gründe: geringerer Systemverwaltungsaufwand sowie verbesserte Verfügbarkeit und besserer Lastausgleich.

Schon bei Einrichtung eines sehr kleinen Clusters profitieren Sie von einer vereinfachten Systemverwaltung. Zu einem Cluster gehörige Warteschlangenmanager benötigen weniger Definitionen, wodurch sich das Risiko von Fehlern bei den Definitionen verringert.

Weitere Informationen zu Clustering finden Sie im Abschnitt Cluster.

## IBM MQ MQI client

IBM MQ MQI-*Clients* sind unabhängig voneinander installierbare Komponenten von IBM MQ. Auf einem MQI-Client können IBM MQ-Anwendungen über ein Kommunikationsprotokoll ausgeführt werden, es ist eine Interaktion mit einem oder mehreren MQI-Servern (MQI = Message Queue Interface) auf anderen Plattformen möglich und es kann eine Verbindung zu den zugehörigen Warteschlangenmanagern hergestellt werden.

Ausführliche Informationen zur Installation und Verwendung von IBM MQ MQI client-Komponenten finden Sie in den folgenden Abschnitten:

- 

IBM MQ-Client unter AIX installieren
- 

IBM MQ-Client unter Linux® installieren
- 

IBM MQ-Client unter Windows installieren
- 

IBM MQ-Client unter IBM i installieren

und Verbindungen zwischen Server und Client konfigurieren.

## Einreihung in Warteschlange innerhalb von Gruppen


Warteschlangenmanager in einer Gruppe mit gemeinsamer Warteschlange können über normale Kanäle miteinander kommunizieren, es kann aber auch das Verfahren der *gruppeninternen Warteschlangensteuerung* mit schneller Nachrichtenübertragung ohne Definition von Kanälen eingesetzt werden. Dies gilt nur für IBM MQ for z/OS.

Weitere Informationen zur gruppeninternen Warteschlangensteuerung finden Sie unter „Einreihung in Warteschlange innerhalb von Gruppen” auf Seite 232.

## Nachricht

Beim Message-Queuing bezeichnet der Begriff 'Nachricht' eine Datensammlung, die von einem Programm gesendet wurde und an ein anderes Programm gerichtet ist. Siehe IBM MQ-Nachrichten.

Informationen zu Nachrichtentypen finden Sie im Abschnitt Nachrichtentypen.

## Nachrichtenkanalagent

An jedem Ende eines Kanals befindet sich ein Nachrichtenkanalagent. Ein Kanal besteht somit aus einem Paar aus Nachrichtenkanalagenten, einem sendenden und einem empfangenden, über die Nachrichten von einem zum nächsten Warteschlangenmanager übertragen werden.

Informationen zur Verwendung von Nachrichtenkanalagenten finden Sie unter Einführung in das verteilte Warteschlangenmanagement.

## Nachrichtendeskriptor

Eine IBM MQ-Nachricht besteht aus Steuerinformationen und Anwendungsdaten.

Die Steuerinformationen sind in einer Nachrichtendeskriptorstruktur (MQMD) definiert und umfassen u. a. folgende Angaben:

- Nachrichtentyp
- ID für die Nachricht
- Zustellungspriorität der Nachricht

Struktur und Inhalt der Anwendungsdaten werden durch die beteiligten Programme und nicht durch IBM MQ bestimmt.

Weitere Informationen finden Sie unter MQMD.

## Punkt-zu-Punkt-Messaging

Beim Punkt-zu-Punkt-Messaging wird jede Nachricht von einer generierenden Anwendung an eine konsumierende Anwendung übertragen. Zur Übertragung stellt die generierende Anwendung die Nachricht in eine Warteschlange, aus der die konsumierende Anwendung die Nachricht abruft.

## Publish/Subscribe-Messaging

Beim Publish/Subscribe-Messaging wird jeder interessierten Anwendung eine Kopie jeder Nachricht bereitgestellt, die von einer publizierenden Anwendung veröffentlicht wird. Dabei ist die Anzahl der interessierten Anwendungen - keine, eine oder mehrere - unbedeutend. Beim Publish/Subscribe wird eine interessierte Anwendung als Subskribent bezeichnet, und die Nachrichten werden in eine durch eine Subskription festgelegte Warteschlange eingereiht.

Weitere Informationen finden Sie unter „Publish/Subscribe-Messaging” auf Seite 67.

## Warteschlange

Ein benanntes Ziel, an das Nachrichten gesendet werden können. Nachrichten sammeln sich in Warteschlangen an, bis sie von Programmen abgerufen werden, die für diese Warteschlangen zuständig sind.

Weitere Informationen finden Sie unter „Warteschlangen” auf Seite 21.

## Warteschlangenmanager

Ein *Warteschlangenmanager* ist ein Systemprogramm, welches Warteschlangensteuerungsservices für Anwendungen bereitstellt.

Der Warteschlangenmanager stellt eine Anwendungsprogrammierschnittstelle zur Verfügung, über welche Programme Nachrichten in Warteschlangen einreihen und daraus abrufen können. Darüber hinaus bietet der Warteschlangenmanager weitere Funktionen, mit denen Administratoren neue Warteschlangen erstellen, die Eigenschaften bestehender Warteschlangen ändern und den Betrieb des Warteschlangenmanagers steuern können.

Die Message-Queueing-Services von IBM MQ sind auf einem System nur verfügbar, wenn ein Warteschlangenmanager aktiv ist. Auf einem einzelnen System können mehrere Warteschlangenmanager aktiv sein (beispielsweise um ein Testsystem von einem *Live*system zu trennen). Jeder Warteschlangenmanager wird gegenüber einer Anwendung über eine *Verbindungskennung* (*Hconn*) identifiziert.

Die Services des Warteschlangenmanagers können von vielen verschiedenen Anwendungen, die komplett unabhängig voneinander sein können, gleichzeitig verwendet werden. Damit ein Programm die Services eines Warteschlangenmanagers verwenden kann, muss es eine Verbindung zu dem betreffenden Warteschlangenmanager herstellen.

Anwendungen können Nachrichten nur dann an Anwendungen senden, die mit anderen Warteschlangenmanagern verbunden sind, wenn die betreffenden Warteschlangenmanager in der Lage sind, miteinander zu kommunizieren. Für die sichere Nachrichtenübermittlung zwischen solchen Anwendungen implementiert IBM MQ ein *Store-and-forward*-Protokoll.

Weitere Informationen finden Sie unter „Warteschlangenmanager” auf Seite 30.

## Gruppe mit gemeinsamer Warteschlange


Die Warteschlangenmanager, die auf dieselbe Gruppe gemeinsam genutzter Warteschlangen zugreifen können, bilden eine sogenannte *Gruppe mit gemeinsamer Warteschlange*. Sie kommunizieren über eine Coupling-Facility (CF) miteinander, auf der die gemeinsam genutzten Warteschlangen gespeichert werden. Dies gilt nur für IBM MQ for z/OS.

Weitere Informationen finden Sie unter „Gemeinsam genutzte Warteschlangen und Gruppen mit gemeinsamer Warteschlange” auf Seite 181.

### Gemeinsam genutzte Warteschlange

z/OS

Eine *gemeinsam genutzte Warteschlange* ist eine bestimmte Art von lokaler Warteschlange, auf deren Nachrichten ein oder mehrere Warteschlangenmanager in einem Sysplex zugreifen können. Dies ist nicht zu verwechseln mit einer Warteschlange, die von mehreren Anwendungen gemeinsam genutzt wird, die denselben Warteschlangenmanager verwenden. Dies gilt nur für IBM MQ for z/OS.

### Abonnement

Eine Publish/Subscribe-Anwendung kann ihr Interesse an Nachrichten zu bestimmten Themen registrieren lassen. Mit der Registrierung wird die Anwendung zu einem Subskribenten. Der Begriff 'Subskription' definiert dabei, auf welche Weise passende Nachrichten zur Verarbeitung in Warteschlangen eingereiht werden.

Eine Subskription enthält Informationen zur Identität des Subskribenten und der Zielwarteschlange, auf der Veröffentlichungen eingereiht werden sollen. Sie enthält außerdem Informationen darüber, wie eine Veröffentlichung in der Zielwarteschlange eingereiht wird.

Weitere Informationen finden Sie unter „Subskribenten und Subskriptionen” auf Seite 71.

### Thema

Ein Thema ist eine Zeichenfolge, die den Gegenstand der Informationen beschreibt, die in einer Publish/Subscribe-Nachricht veröffentlicht werden.

Themen sind wichtig für die erfolgreiche Nachrichtenübermittlung in einem Publish/Subscribe-System. Anstatt eine bestimmte Zieladresse in die einzelnen Nachrichten einzufügen, ordnet ein Publisher jeder Nachricht ein Thema zu. Der Warteschlangenmanager gleicht das Thema mit einer Liste von Abonnenten ab, die das Thema abonniert haben, und stellt jedem dieser Abonnenten die Nachricht zu.

Weitere Informationen finden Sie unter „Themen” auf Seite 74.

## Nachrichten und Warteschlangen

Nachrichten und Warteschlangen sind die Basiskomponenten eines Message-Queuing-Systems.

### Was ist eine Nachricht?

Eine *Nachricht* ist eine Bytefolge, die Informationen für die Anwendungen bereitstellt, die diese Nachricht verwendet. Mithilfe von Nachrichten werden Informationen von einem Anwendungsprogramm an ein anderes oder zwischen den verschiedenen Teilen ein und derselben Anwendung übertragen. Die Anwendungen können dabei auf derselben oder auf verschiedenen Plattformen aktiv sein.

Eine IBM MQ-Nachricht setzt sich aus folgenden Komponenten zusammen:

- *Den Anwendungsdaten.*Inhalt und Struktur der Anwendungsdaten werden vom Anwendungsprogramm vorgegeben, das diese Daten nutzt.
- *Einem Nachrichtendeskriptor.*Der Nachrichtendeskriptor dient zur Kennzeichnung der Nachricht und enthält zusätzliche Steuerinformationen, z. B. den Nachrichtentyp und die Priorität, die der Nachricht von der sendenden Anwendung zugewiesen wurde.

  Das Format des Nachrichtendeskriptors wird von IBM MQ definiert. Eine umfassende Beschreibung des Nachrichtendeskriptors finden Sie im Abschnitt MQMD – Nachrichtendeskriptor.
- *Nachrichteneigenschaften.* Die Metadaten zur Nachricht. Der Inhalt der Nachrichteneigenschaften wird durch die Anwendungsprogramme vorgegeben, die diese Eigenschaften nutzen. Weitere Informationen finden Sie im Abschnitt Nachrichteneigenschaften.

## Nachrichtenlänge

Die maximale Nachrichtenlänge ist standardmäßig 4 MB; diesen Wert können Sie auf maximal 100 MB (1 MB = 1,048,576 Byte) erhöhen. Tatsächlich kann die Nachrichtenlänge durch folgende Faktoren begrenzt werden:

- Die für die Empfangswarteschlange definierte maximale Nachrichtenlänge
- Die für den Warteschlangenmanager definierte maximale Nachrichtenlänge
- Die von der Warteschlange definierte maximale Nachrichtenlänge
- Die von der sendenden oder empfangenden Anwendung definierte maximale Nachrichtenlänge
- Dem für die Nachricht verfügbaren Speicherplatz

In manchen Fällen sind mehrere Nachrichten erforderlich, um alle notwendigen Informationen an eine Anwendung zu übertragen.

## Wie werden Nachrichten von Anwendungen gesendet und empfangen?

Anwendungsprogramme senden und empfangen Nachrichten über **MQI-Aufrufe**.

Um beispielsweise eine Nachricht in eine Warteschlange einzureihen, sind folgende Schritte erforderlich:

1. Die Anwendung öffnet die betreffende Warteschlange, indem sie den MQ-Aufruf `MQOPEN` ausgibt.
2. Anschließend gibt die Anwendung den MQI-Aufruf `MQPUT` aus, um die Nachricht in die Warteschlange einzureihen.

Diese Nachricht kann von einer anderen Anwendung mit dem MQI-Aufruf `MQGET` aus der Warteschlange abgerufen werden.

Weitere Informationen zu MQI-Aufrufen finden Sie im Abschnitt MQI-Aufrufe.

## Was ist eine Warteschlange?

Eine *Warteschlange* ist eine Datenstruktur, in der Nachrichten gespeichert werden.

Jede Warteschlange ist einem *Warteschlangenmanager* zugeordnet, der damit Eigner der Warteschlange ist. Der Warteschlangenmanager ist zuständig für die Verwaltung aller ihm zugeordneten Warteschlangen; er muss außerdem alle Nachrichten, die er empfängt, in den entsprechenden Warteschlangen speichern. Die Nachrichten können entweder von den Anwendungsprogrammen oder vom Warteschlangenmanager (als Teil des normalen Warteschlangenmanagerbetriebs) in die Warteschlangen eingereiht werden.

## Vordefinierte und dynamische Warteschlangen

Warteschlangen werden anhand der Art und Weise ihrer Erstellung unterschieden:

- **Vordefinierte Warteschlangen** werden vom Administrator mit den entsprechenden MQSC- oder PCF-Befehlen erstellt.Vordefinierte Warteschlangen sind permanent; sie sind unabhängig von den Anwendungen vorhanden, von denen sie verwendet werden, und sind auch nach einem Neustart von IBM MQ noch vorhanden.
- **Dynamische Warteschlangen** werden erstellt, wenn eine Anwendung eine `MQOPEN`-Anforderung unter Angabe des Namens einer *Modellwarteschlange* ausgibt. Die Warteschlange wird auf Basis einer *Definition für eine Warteschlangenschablone*, einer sogenannten Modellwarteschlange, erstellt.Modellwarteschlangen können mit dem WebSphere MQ-Scriptbefehl DEFINE QMODEL erstellt werden. Die Attribute einer Modellwarteschlange (beispielsweise die Anzahl der Nachrichten, die maximal in der Warteschlange gespeichert werden können) werden von allen dynamischen Warteschlangen übernommen, die auf Basis dieser Modellwarteschlange erstellt werden.

  Modellwarteschlangen haben Attribute, die angeben, ob die dynamische Warteschlange permanent oder temporär sein soll. Permanente Warteschlangen sind auch nach dem Neustart einer Anwendung oder des Warteschlangenmanagers noch vorhanden, temporäre Warteschlangen hingegen nicht.

### Nachrichten aus Warteschlangen abrufen

Anwendungen mit der entsprechenden Berechtigung können Nachrichten anhand der folgenden Abrufalgorithmen aus einer Warteschlange abrufen:

- FIFI (First In/First Out).
- Nachrichtenpriorität (ist im Nachrichtendeskriptor festgelegt). Nachrichten mit derselben Priorität werden nach dem FIFI-Prinzip (First In/First Out) abgerufen.
- Programmanforderung für eine bestimmte Nachricht.

Der Algorithmus ist in der MQGET-Anforderung der Anwendung festgelegt.

## IBM MQ-Objekte

Die Eigenschaften von IBM MQ-Objekten werden durch Warteschlangenmanager festgelegt. Die Werte dieser Eigenschaften bestimmen, wie diese Objekte von IBM MQ verarbeitet werden. Objekte werden mit den von IBM MQ bereitgestellten Befehlen und Schnittstellen erstellt und verwaltet. In den Anwendungen werden Objekte über die MQI (Message Queue Interface) gesteuert. Programme verweisen auf Objekte anhand eines IBM MQ *Objektdeskriptors* (MQOD).

### Objektverwaltung

Die Objektverwaltung beinhaltet folgende Tasks:

- Warteschlangenmanager starten und stoppen
- Objekte, insbesondere Warteschlangen, für Anwendungen erstellen
- Objektattribute anzeigen und ändern
- Objekte löschen
- Die Arbeit mit Kanälen, um Kommunikationspfade zu Warteschlangenmanagern auf anderen (fernen) Systemen zu erstellen.
- *Warteschlangenmanager-Cluster* erstellen, um die Verwaltung zu vereinfachen und die Arbeitslast gleichmäßig zu verteilen.

Mit Ausnahme dynamischer Warteschlangen müssen Objekte im Warteschlangenmanager definiert sein, damit sie verwendet werden können.

Wenn Sie eine Objektverwaltungsoperation mit einem IBM MQ-Befehl ausführen, prüft der Warteschlangenmanager, ob Sie über die entsprechende Berechtigung zur Ausführung dieses Vorgangs verfügen. Ebenso überprüft der Warteschlangenmanager auch bei Anwendungen, die versuchen, ein Objekt mit einem MQOPEN-Aufruf zu öffnen, ob diese über die entsprechenden Berechtigungen verfügen, bevor der Zugriff auf das Objekt gestattet wird. Dabei werden die Prüfungen für den Namen des Objekts vorgenommen, das geöffnet werden soll.

Objekte können wie folgt definiert und verwaltet werden:

- Mit den in den Abschnitten Referenz zu Programmable Command Formats und Verwaltungstasks automatisieren beschriebenen PCF-Befehlen.
- Die im Abschnitt MQSC-Befehle beschriebenen MQSC-Befehle
- 

Die Operationen und Steuerkonsolen von IBM MQ for z/OS , beschrieben in IBM MQ for z/OS verwalten
- 

Über IBM MQ Explorer (nur Windows- und Linux for Intel-Systeme). Weitere Informationen finden Sie im Abschnitt Einführung in MQ Explorer.

Weitere Möglichkeiten für die Objektverwaltung sind:

- Steuerbefehle, die über die Tastatur eingeben werden. Siehe IBM MQ for Multiplatforms mit Steuerbefehlen verwalten.

- MQAI-Aufrufe (IBM MQ Administration Interface) in einem Programm. Siehe IBM MQ Administration Interface (MQAI).

ALW Für IBM MQ-Befehlsfolgen unter AIX, Linux, and Windows können Sie die MQSC-Funktion verwenden, um eine Reihe von Befehlen in einer Datei auszuführen. Weitere Informationen finden Sie unter IBM MQ mit MQSC-Befehlen verwalten.

IBM i Für häufig verwendete IBM MQ for IBM i-Befehlsfolgen können Sie CL-Programme schreiben. Weitere Informationen finden Sie im Abschnitt IBM MQ for IBM i mit CL-Befehlen verwalten.

z/OS Für häufig verwendete IBM MQ for z/OS-Befehlsfolgen können Sie auch Verwaltungsprogramme schreiben, die Nachrichten mit diesen Befehlen erstellen und in die Eingabewarteschlange für Systembefehle einreihen. Der Warteschlangenmanager verarbeitet die Nachrichten in dieser Warteschlange wie Befehle, die über die Befehlszeile oder die Betriebs- und Steueranzeigen eingegeben werden. Dieses Verfahren wird unter Programme für die Verwaltung von IBM MQ schreiben erläutert und in der mit IBM MQ for z/OS gelieferten Beispielanwendung 'Mail Manager' veranschaulicht. Eine Beschreibung dieses Beispiels finden Sie unter Beispielprogramme für IBM MQ for z/OS .

## Objektattribute

Die Eigenschaften eines Objektes werden über die Objektattribute definiert. Einige dieser Attribute können Sie angeben, andere hingegen können nur angezeigt werden.

Das Attribut **MaxMsgLength** einer Warteschlange gibt beispielsweise die maximal mögliche Länge für Nachrichten an, die in dieser Warteschlange eingereiht werden können. Das Attribut **DefinitionType** gibt an, wie die Warteschlange erstellt wurde; dieses Attribut kann nur angezeigt werden.

In IBM MQ gibt es für den Verweis auf Attribute zwei Möglichkeiten:

- Über den PCF-Namen des Attributs (beispielsweise **MaxMsgLength**).
- Über den WebSphere MQ-Scriptbefehlsnamen des Attributs (beispielsweise MAXMSGL).

## Gruppen mit gemeinsamer Warteschlange

z/OS

Warteschlangenmanager, die auf dieselbe Gruppe gemeinsam genutzter Warteschlangen zugreifen können, bilden eine sogenannte *Gruppe mit gemeinsamer Warteschlange*. Sie kommunizieren über eine Coupling-Facility miteinander, auf der die gemeinsam genutzten Warteschlangen gespeichert werden. Beachten Sie, dass eine Gruppe mit gemeinsamer Warteschlange kein striktes Objekt ist.

Eine gemeinsam genutzte Warteschlange ist eine bestimmte Art von lokaler Warteschlange, auf deren Nachrichten ein oder mehrere Warteschlangenmanager aus einer Gruppe mit gemeinsamer Warteschlange zugreifen können. Dies ist nicht zu verwechseln mit einer Warteschlange, die von mehreren Anwendungen gemeinsam genutzt wird, die denselben Warteschlangenmanager verwenden.

Die Namen von Gruppen mit gemeinsamer Warteschlange können bis zu vier Zeichen lang sein. Der Name muss in Ihrem Netz eindeutig sein und muss sich von allen WS-Managernamen unterscheiden.

**Wichtig:** Gemeinsam genutzte Warteschlangen und Gruppen mit gemeinsamer Warteschlange werden nur in IBM MQ for z/OS unterstützt.

Weitere Informationen finden Sie unter „Gemeinsam genutzte Warteschlangen und Gruppen mit gemeinsamer Warteschlange” auf Seite 181.

## Systemstandardwertobjekte

*Systemstandardobjekte* sind Objektdefinitionen, die bei der Erstellung eines Warteschlangenmanagers automatisch erstellt werden. Sie können diese Objektdefinitionen für die Verwendung in den Anwendungen Ihrer Installation kopieren und entsprechend ändern.

Standardobjektnamen beginnen mit SYSTEM; die lokale Standardschlange beispielsweise hat die Bezeichnung SYSTEM.DEFAULT.LOCAL.QUEUE, der Standardempfängerkanal die Bezeichnung SYSTEM.DEF.RECEIVER. Die Standardobjekte mit diesen Namen sind erforderlich; sie können nicht umbenannt werden.

Alle Attribute, die bei der Definition eines Objekts nicht angegeben werden, werden explizit aus dem entsprechenden Standardobjekt kopiert. Wenn Sie beispielsweise eine lokale Warteschlange definieren, werden die von Ihnen nicht angegebenen Attribute aus der Standardwarteschlange SYSTEM.DEFAULT.LOCAL.QUEUE übernommen.

Weitere Informationen finden Sie unter System-und Standardobjekte .

# Objekttypen

Bei vielen Verwaltungstasks werden verschiedene Arten von IBM MQ-*Objekten* bearbeitet.

Informationen zur Benennung von IBM MQ-Objekten finden Sie im Abschnitt „IBM MQ-Objekte benennen” auf Seite 39.

Informationen zu den auf einem Warteschlangenmanager erstellten Standardobjekten finden Sie im Abschnitt „Systemstandardwertobjekte” auf Seite 16.

Informationen zu den verschiedenen Typen von IBM MQ -Objekten finden Sie in den folgenden Abschnitten:

## Authentifizierungsdatenobjekte

Ein Authentifizierungsdatenobjekt stellt die Definitionen bereit, die für die Überprüfung auf widerrufene Zertifikate erforderlich sind.

Das Authentifizierungsdatenobjekt eines Warteschlangenmanagers ist Teil der TLS-Unterstützung (Transport Layer Security) von IBM MQ. Es stellt die Definitionen bereit, die für eine Überprüfung auf widerrufene Zertifikate erforderlich sind. Zertifikate, die nicht mehr als vertrauenswürdig eingestuft werden, werden von Zertifizierungsstellen widerrufen.

Sie können mit dem MQSC-Befehl **DEFINE AUTHINFO** ein Authentifizierungsdatenobjekt definieren. Weitere Informationen zu den Attributen von Authentifizierungsinformationsobjekten finden Sie unter **DEFINE AUTHINFO**.

Sie können folgende IBM MQ-Steuerbefehle für ein Authentifizierungsdatenobjekt verwenden:

- **setmqaut** (Berechtigung erteilen oder entziehen)
- **dspmqaut** (Vergabe der Objektberechtigung anzeigen)
- **dmpmqaut** (Speicherauszugsberechtigungen)
- **rcrmqobj** (Objekt erneut erstellen)
- **rcdmqimg** (Medienimage aufzeichnen)
- **dspmqfls** (Dateinamen anzeigen)

Eine Übersicht über TLS und die Verwendung der Authentifizierungsinformationsobjekte finden Sie unter TLS-Konzepte (Transport Layer Security) und TLS-Sicherheitsprotokolle in IBM MQ.

## Kanäle

Kanäle sind Objekte, die einen Kommunikationspfad von einem Warteschlangenmanager zu einem anderen bereitstellen.

Weitere Informationen finden Sie unter „Kanäle” auf Seite 32.

## Kommunikationsinformationsobjekte

IBM MQ Multicast bietet eine geringe Latenzzeit, eine hohe Ausgabefächerung und eine zuverlässige Multicasting-Nachrichtenübertragung. Für die Multicasting-Übertragung ist ein Kommunikationsinformationsobjekt (COMMINFO-Objekt) erforderlich.

Weitere Informationen finden Sie unter „IBM MQ Multicasting” auf Seite 115.

Bei COMMINFO-Objekten handelt es sich um IBM MQ-Objekte, die Attribute in Zusammenhang mit Multicast-Übertragungen enthalten. Weitere Informationen zu diesen Attributen finden Sie im Abschnitt DEFINE COMMINFO. Weitere Informationen zur Erstellung von COMMINFO-Objekten finden Sie im Abschnitt Erste Schritte mit Multicasting.

## Empfangsprogramme

*Empfangsprogramme* sind Prozesse, die Netzanforderungen von anderen Warteschlangenmanagern oder Clientanwendungen annehmen und die zugeordneten Kanäle starten.

*Listenerprozesse* können mit dem Steuerbefehl **runmqlsr** gestartet werden.

*Empfangsprogrammobjekte* sind IBM MQ-Objekte, mit denen Empfangsprogrammprozesse auf Warteschlangenmanagerebene gestartet und gestoppt werden können. Über die Definition der Attribute eines Empfangsprogramms können Sie:

- den Empfangsprogrammprozess definieren;
- angeben, ob der Empfangsprogrammprozess beim Starten bzw. Stoppen des Warteschlangenmanagers automatisch gestartet bzw. gestoppt werden soll.

**Wichtig:** z/OS Empfangsprogrammobjekte werden in IBM MQ for z/OS nicht unterstützt. Weitere Informationen zur Implementierung der Empfangsbereitschaft in IBM MQ for z/OS mithilfe des Kanalinitiators finden Sie im Abschnitt „Kanalinitiator unter z/OS” auf Seite 176.

## Namenslisten

Eine *Namensliste* ist ein IBM MQ-Objekt, das eine Liste mit Clusternamen, Warteschlangennamen oder Namen von Authentifizierungsdatenobjekten enthält. In einem Cluster kann dieses Objekt eine Liste der Cluster enthalten, für die der Warteschlangenmanager die Repositorys enthält.

Eine Namensliste ist ein IBM MQ-Objekt, das eine Liste anderer IBM MQ-Objekte enthält. Namenslisten werden in der Regel von Anwendungen wie beispielsweise Auslösemonitoren verwendet (in diesem Fall ist in diesem Objekt eine Gruppe von Warteschlangen angegeben). Namenslisten haben den Vorteil, dass sie unabhängig von Anwendungen verwaltet werden können; sie können aktualisiert werden, ohne dass eine der Anwendungen gestoppt werden muss, die dieses Objekt verwenden. Der Ausfall einer Anwendung hat keinerlei Auswirkungen auf die Namensliste - sie kann von den anderen Anwendungen weiterhin verwendet werden.

Namenslisten werden auch mit Warteschlangenmanagerclustern verwendet, um eine Liste von Clustern zu verwalten, auf die von mehreren IBM MQ -Objekten verwiesen wird.

Sie können Namenslisten mit den MQSC-Befehlen DEFINE NAMELIST und ALTER NAMELIST definieren und ändern.

**Anmerkung:** z/OS Alternativ können Sie unter z/OSdie Operationen und Steuerkonsolen von IBM MQ for z/OS verwenden.

Programme können über die MQI feststellen, welche Warteschlangen in diesen Namenslisten aufgeführt sind. Die Verwaltung der Namenslisten ist Aufgabe des Anwendungsentwicklers und des Systemadministrators.

Eine Liste der verfügbaren Namenslistenattribute finden Sie unter Attribute für Namenslisten.

## Prozessdefinitionen

Über Prozessdefinitionsobjekte können Anwendungen ohne Bedienereingriff gestartet werden; dazu werden die Attribute der Anwendung für den Warteschlangenmanager definiert.

Prozessdefinitionsobjekte definieren eine Anwendung, die auf ein Auslöserereignis auf einem IBM MQ-Warteschlangenmanager hin gestartet wird. Zu den Prozessdefinitionsattributen gehören die Anwendungs-ID, der Anwendungstyp sowie die anwendungsspezifischen Daten. Weitere Informationen finden Sie unter *Initialisierungswarteschlangen* im Abschnitt „Für bestimmte Zwecke von IBM MQ verwendete Warteschlangen” auf Seite 29.

Damit eine Anwendung ohne Bedienereingriff gestartet werden kann (siehe Abschnitt IBM MQ-Anwendungen mithilfe von Auslösten starten), müssen die Attribute der Anwendung dem Warteschlangenmanager bekannt sein. Diese Attribute werden in einem *Prozessdefinitionsobjekt* definiert.

Das Attribut **ProcessName** wird bei der Erstellung des Objekts festgelegt. Andere Attribute können Sie jedoch mithilfe von IBM MQ-Befehlen ändern.

**Anmerkung:** z/OS Alternativ können Sie unter z/OSdie Operationen und Steuerkonsolen von IBM MQ for z/OS verwenden.

Informationen zu den Werten aller Attribute finden Sie im Abschnitt MQINQ – Objektattribute abfragen.

Eine Liste der verfügbaren Prozessdefinitionsattribute finden Sie unter Attribute für Prozessdefinitionen.

## Warteschlangen

Eine IBM MQ-*Warteschlange* ist ein benanntes Objekt, in das Anwendungen Nachrichten einreihen und aus dem Anwendungen Nachrichten abrufen können.

Weitere Informationen finden Sie unter „Warteschlangen” auf Seite 21.

## Warteschlangenmanager

IBM MQ Warteschlangenmanager stellen Services zur Warteschlangensteuerung für Anwendungen bereit und verwalten die Warteschlangen, die zu ihnen gehören.

Weitere Informationen finden Sie unter „Warteschlangenmanager” auf Seite 30.

## Services

Über *Serviceobjekte* können die Programme definiert werden, die beim Starten oder Stoppen eines Warteschlangenmanagers ausgeführt werden sollen.

Es kann sich um folgende Programmtypen handeln:

**Server**
Für Serviceobjekte des Typs *Server* ist der Parameter SERVTYPE auf SERVER gesetzt. Bei einem Serviceobjekt des Typs SERVER handelt es sich um die Definition eines Programms, das beim Start des angegebenen Warteschlangenmanagers ausgeführt wird. Es kann immer nur jeweils eine Instanz eines Serverprozesses ausgeführt werden. Der Status eines Serverprozesses kann während der Ausführung mit dem WebSphere MQ-Scriptbefehl DISPLAY SVSTATUS überwacht werden. Serverserviceobjekte sind in der Regel Definitionen von Programmen wie beispielsweise Steuerroutinen für nicht zustellbare Nachrichten oder Auslösemonitore; die Programme, die ausgeführt werden können, sind allerdings nicht auf die von IBM MQ bereitgestellten Programme beschränkt. Darüber hinaus kann ein Serverserviceobjekt auch einen Befehl enthalten, mit dem das Programm nach der Beendigung des angegebenen Warteschlangenmanagers beendet wird.

**Befehle**
Ein *Befehl* ist ein Serviceobjekt, für das der Parameter SERVTYPE auf COMMAND gesetzt ist. Ein Befehlsserviceobjekt ist die Definition eines Programms, das beim Starten oder Stoppen des angegebenen Warteschlangenmanagers ausgeführt wird. Mehrere Instanzen eines Befehlsprozesses können gleichzeitig ausgeführt werden. Befehlsserviceobjekte unterscheiden sich von Serverserviceobjekten

darin, dass das Programm bei der Ausführung nicht vom Warteschlangenmanager überwacht wird. Befehlsserviceobjekte sind Definitionen von Programmen mit einer kurzen Lebensdauer, mit denen eine bestimmte Aufgabe wie beispielsweise das Starten einer oder mehrerer Tasks ausgeführt wird.

**Wichtig:** z/OS Serviceobjekte werden unter IBM MQ for z/OS nicht unterstützt.

Weitere Informationen hierzu finden Sie unter Mit Services arbeiten.

## Speicherklassen

z/OS

Eine Speicherklasse ordnet einer Seitengruppe eine oder mehrere Warteschlangen zu.

Folglich werden Nachrichten für die Warteschlange in der entsprechenden Seitengruppe gespeichert (gepuffert).

Speicherklassen werden nur in IBM MQ for z/OS unterstützt.

Weitere Informationen zu Speicherklassen finden Sie unter IBM MQ -Umgebung unter z/OSplanen.

## Topic-Objekte

Ein *Themenobjekt* ist ein IBM MQ-Objekt, mit dem Themen bestimmte, nicht standardmäßige Attribute zugewiesen werden können.

Ein *Thema* wird von einer Anwendung definiert, die für eine bestimmte *Themenzeichenfolge* eine Veröffentlichung oder Subskription durchführt. Eine Themenzeichenfolge kann eine Hierarchie von Themen angeben, indem die Themen durch einen Schrägstrich (/) voneinander getrennt werden. Dies kann durch eine *Themenstruktur* dargestellt werden. Wenn eine Anwendung beispielsweise Veröffentlichungen für die Themenzeichenfolgen `/Sport/American Football` und `/Sport/Soccer` durchführt, wird eine Themenstruktur mit dem übergeordneten Knoten `Sport` und den zwei untergeordneten Knoten `American Football` und `Soccer` erstellt.

Themen übernehmen ihre Attribute vom ersten übergeordneten Verwaltungsknoten, der in ihrer Themenstruktur gefunden wird. Wenn eine bestimmte Themenstruktur keine administrativen Themenknoten enthält, übernehmen alle Themen ihre Attribute vom Basisthemenobjekt SYSTEM.BASE.TOPIC.

Sie können jedem Knoten einer Themenstruktur ein Themenobjekt hinzufügen, indem Sie die Themenzeichenfolge des betreffenden Knotens im Attribut TOPICSTR des Themenobjekts angeben. Sie können auch weitere Attribute für den administrativen Themenknoten definieren. Weitere Informationen zu diesen Attributen finden Sie in den Abschnitten MQSC-Befehle bzw. Verwaltung mithilfe von PCF-Befehlen automatisieren. Jedes Themenobjekt übernimmt standardmäßig seine Attribute von dem übergeordneten administrativen Themenknoten, der ihm am nächsten liegt.

Themenobjekte können auch dazu verwendet werden, die vollständige Themenstruktur vor Anwendungsentwicklern zu verbergen. Wenn ein Themenobjekt mit dem Namen `FOOTBALL.US` für das Thema `/Sport/American Football`erstellt wird, kann eine Anwendung das Objekt mit dem Namen `FOOTBALL.US` anstelle der Zeichenfolge `/Sport/American Football` mit demselben Ergebnis veröffentlichen oder subskribieren.

Wenn Sie in einer Themenzeichenfolge für ein Themenobjekt das Zeichen #, +, / oder * eingeben, wird es wie ein normales Zeichen innerhalb der Zeichenfolge behandelt, d. h., es wird als Teil der Themenzeichenfolge betrachtet, die einem Themenobjekt zugeordnet ist.

Weitere Informationen zu Themenobjekten finden Sie im Abschnitt „Publish/Subscribe-Messaging” auf Seite 67.

**Zugehörige Konzepte**

„Einführung in die Nachrichtenwarteschlangensteuerung” auf Seite 5
Mit den IBM MQ-Produkten können Programme über ein Netz ungleicher Komponenten (Prozessoren, Betriebssysteme, Subsysteme und Kommunikationsprotokolle) unter Verwendung einer konsistenten Anwendungsprogrammierschnittstelle miteinander kommunizieren.

**Zugehörige Verweise**
Die MQSC-Befehle

## Warteschlangen

Einführung in IBM MQ-Warteschlangen und Warteschlangenattribute

Nachrichten werden in einer Warteschlange gespeichert. Wenn also die Anwendung, welche die Nachricht einreiht, eine Antwort auf ihre Nachricht erwartet, kann sie währenddessen andere Arbeiten erledigen. Anwendungen greifen über die im Abschnitt Übersicht über das Message Queue Interface beschriebene Message Queue Interface auf eine Warteschlange zu.

Bevor Nachrichten in eine Warteschlange eingereiht werden können, muss diese bereits erstellt sein. Eine Warteschlange gehört zu einem Warteschlangenmanager, der wiederum Eigner vieler Warteschlangen sein kann. Die Namen der einzelnen Warteschlangen müssen allerdings innerhalb des betreffenden Warteschlangenmanagers eindeutig sein.

Eine Warteschlange wird von einem Warteschlangenmanager verwaltet. In den meisten Fällen werden die einzelnen Warteschlangen von dem jeweiligen Warteschlangenmanager physisch verwaltet, ohne dass dies den Anwendungsprogrammen offensichtlich ist. Gemeinsam genutzte Warteschlangen in IBM MQ for z/OS können von jedem Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange verwaltet werden.

Warteschlangen können mithilfe von IBM MQ-Befehlen (MQSC), PCF-Befehlen oder plattformspezifischen Schnittstellen erstellt werden. Die Betriebs- und Steueranzeigen von IBM MQ for z/OS sind beispielsweise plattformspezifisch.

Lokale Warteschlangen für temporäre Jobs können *dynamisch* von der Anwendung aus erstellt werden. So können beispielsweise *Warteschlangen für zu beantwortende Nachrichten* erstellt werden (die nach Beendigung der Anwendung nicht mehr benötigt werden). Weitere Informationen finden Sie in „Dynamische und Modellwarteschlangen” auf Seite 26.

Bevor Sie eine Warteschlange verwenden können, müssen Sie sie zunächst öffnen und dabei angeben, wofür die Warteschlange vorgesehen ist. So können Warteschlangen beispielsweise für folgende Zwecke geöffnet werden:

- Ausschließlich zum Durchsuchen (nicht zum Abrufen) von Nachrichten
- Zum Abrufen von Nachrichten (mit gemeinsamem Zugriff mit anderen Programmen oder mit exklusivem Zugriff)
- Zum Einreihen von Nachrichten in die Warteschlange
- Zur Abfrage der Warteschlangenattribute
- Zur Definition der Warteschlangenattribute

Eine vollständige Liste der Optionen, die beim Öffnen einer Warteschlange angegeben werden können, finden Sie im Abschnitt MQOPEN - Objekt öffnen.

### Warteschlangenattribute

Einige Warteschlangenattribute werden bei der Definition der Warteschlange angegeben und können später nicht mehr geändert werden (z. B. der Warteschlangentyp). Andere Warteschlangenattribute können danach eingeteilt werden, von wem sie geändert werden können:

- Attribute, die vom Warteschlangenmanager während der Warteschlangenverarbeitung geändert werden können (z. B. die aktuelle Warteschlangenlänge)
- Attribute, die nur über Befehle geändert werden können (z. B. die Textbeschreibung der Warteschlange)
- Attribute, die von Anwendungen unter Verwendung des Aufrufs MQSET geändert werden können (z. B. ob Einreihvorgänge in die Warteschlange erlaubt sein sollen).

Die Werte aller Attribute können mit dem Aufruf MQINQ abgerufen werden.

Folgende Attribute werden für mehrere Warteschlangentypen verwendet:

**QName**
Name der Warteschlange

**QType**
Typ der Warteschlange

**QDesc**
Textbeschreibung der Warteschlange

**InhibitGet**
Gibt an, ob Programme Nachrichten aus der Warteschlange abrufen dürfen. Es können jedoch niemals Nachrichten aus fernen Warteschlangen abgerufen werden.

**InhibitPut**
Gibt an, ob Programme Nachrichten in die Warteschlange einreihen dürfen.

**DefPriority**
Gibt die Standardpriorität für in die Warteschlange eingereihte Nachrichten an.

**DefPersistence**
Die Standardpersistenz für Nachrichten, die in die Warteschlange eingereiht werden.

**Bereich**
Steuert, ob auch in einem Namensservice ein Eintrag für diese Warteschlange vorhanden ist.

z/OS Das Attribut **Scope** wird unter z/OS nicht unterstützt.

Eine umfassende Beschreibung dieser Attribute finden Sie im Abschnitt Attribute für Warteschlangen.

## Methoden zur Definition von Warteschlangen

Sie können Warteschlangen für IBM MQ mit dem MQSC-Befehl DEFINE oder dem PCF-Befehl Create Queue definieren. Mit den Befehlen werden Typ und Attribute der Warteschlange angegeben. Über die Attribute einer lokalen Warteschlange wird beispielsweise festgelegt, was geschieht, wenn Anwendungen in MQI-Aufrufen auf diese Warteschlange verweisen. Hier einige Beispiele für Attribute:

- Ob Anwendungen Nachrichten aus der Warteschlange abrufen können (GET-aktiviert).
- Ob Anwendungen Nachrichten in die Warteschlange einreihen können (PUT-aktiviert).
- Ob nur eine Anwendung oder aber mehrere Anwendungen auf die Warteschlange zugreifen können.
- Die maximale Anzahl an Nachrichten, die gleichzeitig in der Warteschlange gespeichert werden können (max. Warteschlangenlänge).
- Die Länge, die Nachrichten maximal haben dürfen, damit sie in die Warteschlange eingereiht werden können.

Es gibt auch verschiedene plattformspezifische Schnittstellen, die Sie verwenden können, um Warteschlangen zu definieren.

**Zugehörige Konzepte**

„Clusterwarteschlangen” auf Seite 63
Eine Clusterwarteschlange wird von einem Clusterwarteschlangenmanager anderen Warteschlangenmanagern im Cluster zur Verfügung gestellt.

„Warteschlangen für nicht zustellbare Nachrichten” auf Seite 53
Die Warteschlange für nicht zustellbare Nachrichten (oder Warteschlange für nicht zugestellte Nachrichten) ist die Warteschlange, an die Nachrichten gesendet werden, wenn sie nicht an ihr eigentliches Ziel weitergeleitet werden können. In der Regel hat jeder Warteschlangenmanager eine solche Warteschlange.

Verwaltung mithilfe von PCF-Befehlen automatisieren

In der IBM MQ Console mit Warteschlangen arbeiten

**Zugehörige Tasks**

IBM MQ mit MQSC-Befehlen verwalten

Warteschlangenmanager und Objekte mit MQ Explorer erstellen und konfigurieren

IBM i IBM MQ for IBM i mithilfe von CL-Befehlen verwalten

z/OS Quellen, aus denen Sie MQSC-und PCF-Befehle unter IBM MQ for z/OS ausgeben können

**Zugehörige Verweise**

„Vergleich zwischen gemeinsam genutzten Warteschlangen und Clusterwarteschlangen" auf Seite 63
Diese Informationen sollen Ihnen helfen, gemeinsam genutzte Warteschlangen und Clusterwarteschlangen miteinander zu vergleichen und dann zu entscheiden, welche für Ihr System besser geeignet sind.

**Zugehörige Informationen**

„Was ist eine gemeinsam genutzte Warteschlange?" auf Seite 181

## *Lokale Warteschlangen*

Übertragungs-, Initialisierungs-, Befehls-, Standard-, Kanal- und Ereigniswarteschlangen sowie Warteschlangen für nicht zustellbare Nachrichten sind verschiedene Typen lokaler Warteschlangen.

Eine Warteschlange gilt für ein Programm als *lokal*, wenn sie dem Warteschlangenmanager zugeordnet ist, mit dem das Programm verbunden ist. Nachrichten können aus lokalen Warteschlangen abgerufen und darin eingereiht werden.

Das Warteschlangendefinitionsobjekt enthält die Definitionsinformationen der Warteschlange sowie die in die Warteschlange eingereihten physischen Nachrichten.

Jeder Warteschlangenmanager kann über einige lokale Warteschlangen verfügen, die für spezielle Zwecke eingesetzt werden:

**Übertragungswarteschlangen**

Wenn eine Anwendung eine Nachricht an eine ferne Warteschlange sendet, speichert der lokale Warteschlangenmanager die Nachricht in einer speziellen lokalen Warteschlange, der sogenannten *Übertragungswarteschlange*. Anwendungen können Nachrichten direkt in eine Übertragungswarteschlange einreihen oder indirekt mithilfe der Definition einer fernen Warteschlange.

Wenn ein Warteschlangenmanager Nachrichten an einen fernen Warteschlangenmanager sendet, ermittelt er die Übertragungswarteschlange in der folgenden Reihenfolge:

1. Die Übertragungswarteschlange, die im Attribut XMITQ der lokalen Definition einer fernen Warteschlange angegeben ist.
2. Eine Übertragungswarteschlange mit demselben Namen wie dem des fernen Warteschlangenmanagers. Dies ist der Standardwert des Attributs XMITQ der lokalen Definition einer fernen Warteschlange.
3. Die Übertragungswarteschlange, die im Attribut DEFXMITQ des lokalen Warteschlangenmanagers angegeben ist.

Ein *Nachrichtenkanalagent* ist ein der Übertragungswarteschlange zugeordnetes Kanalprogramm, das die Nachricht an das nächste Ziel übermittelt. Bei dem nächsten Ziel handelt es sich um den Warteschlangenmanager, mit dem der Nachrichtenkanal verbunden ist. Dieser Warteschlangenmanager muss nicht zwangsläufig das endgültige Ziel der Nachricht darstellen. Nach ihrer Übermittlung an das nächste Ziel wird die Nachricht aus der Übertragungswarteschlange gelöscht. Möglicherweise muss die Nachricht auf ihrem Weg zum Zielort viele Warteschlangenmanager durchlaufen. Auf jedem Warteschlangenmanager entlang der Route muss eine Übertragungswarteschlange definiert werden. Diese enthalten jeweils Nachrichten, die auf die Übertragung zum nächsten Ziel warten. Eine normale Übertragungswarteschlange enthält Nachrichten für das nächste Ziel, die jedoch nicht alle denselben endgültigen Zielort haben müssen. In einer Clusterübertragungswarteschlange sind Nachrichten für mehrere Ziele enthalten. Die Korrelations-ID (`correlID`) der einzelnen Nachrichten gibt den Kanal an, in den die Nachricht zur Übertragung an das nächste Ziel gestellt wird.

An einem Warteschlangenmanager können mehrere Übertragungswarteschlangen definiert werden. Sie können auch mehrere Übertragungswarteschlangen für dasselbe Ziel definieren, die jeweils für eine andere Berechtigungsklasse verwendet werden. So ist es unter Umständen sinnvoll, für kurze und lange Nachrichten an dasselbe Ziel unterschiedliche Übertragungswarteschlangen zu erstellen. Die

Nachrichten können dann über verschiedene Nachrichtenkanäle übertragen werden, sodass die kürzeren Nachrichten nicht von den längeren aufgehalten werden. Alle Nachrichten an Clusterwarteschlangen oder Cluster-Topics werden standardmäßig in eine einzige Clusterübertragungswarteschlange namens `SYSTEM.CLUSTER.TRANSMIT.QUEUE` gestellt. Sie haben die Möglichkeit, die Standardeinstellung zu ändern und die Nachrichtenübertragung an verschiedene Cluster-Warteschlangenmanager auf mehrere Clusterübertragungswarteschlangen aufzuteilen. Wenn Sie das Warteschlangenmanager-Attribut `DEFCLXQ` auf `CHANNEL` setzen, erstellt jeder Clustersenderkanal eine separate Clusterübertragungswarteschlange. Eine weitere Möglichkeit besteht darin, manuell Clusterübertragungswarteschlangen zu definieren, die von den Clustersenderkanälen verwendet werden sollen.

Mithilfe von Übertragungswarteschlangen kann ein Nachrichtenkanalagent veranlasst werden, Nachrichten weiterzuleiten. Informationen hierzu finden Sie im Abschnitt IBM MQ-Anwendungen mithilfe von Auslösern starten.

z/OS Wenn Sie unter IBM MQ for z/OS eine gruppeninterne Warteschlangensteuerung verwenden, ist für die Übertragungswarteschlange ein *Agent für die gruppeninterne Warteschlangensteuerung* zuständig. Bei Einsatz der gruppeninternen Warteschlangensteuerung in IBM MQ for z/OS wird eine gemeinsam genutzte Übertragungswarteschlange verwendet.

**Initialisierungswarteschlangen**

Eine *Initialisierungswarteschlange* ist eine lokale Warteschlange, in die der Warteschlangenmanager eine Auslösenachricht einreiht, wenn in einer Anwendungswarteschlange ein Auslöserereignis eintritt.

Ein Auslöserereignis ist ein Ereignis, das ein Programm dazu veranlassen soll, die Verarbeitung einer Warteschlange zu starten. Bei einem solchen Ereignis könnte es sich beispielsweise um den Eingang von mehr als 10 Nachrichten handeln. Weitere Informationen zur Funktionsweise von Auslösern finden Sie im Abschnitt IBM MQ-Anwendungen mithilfe von Auslösern starten.

**Warteschlange für nicht zustellbare Nachrichten**

Eine *Warteschlange für nicht zustellbare Nachrichten* ist eine lokale Warteschlange, in die der Warteschlangenmanager Nachrichten einreiht, die nicht zugestellt werden können.

Wenn der Warteschlangenmanager eine Nachricht in die Warteschlange für nicht zustellbare Nachrichten stellt, fügt er ihr einen Header hinzu. In den Headerinformationen ist unter anderem auch angegeben, warum die Nachricht vom Warteschlangenmanager in die Warteschlange für nicht zustellbare Nachrichten gestellt wurde. Darüber hinaus sind der Zielort der ursprünglichen Nachricht sowie der Zeitpunkt (Datum und Uhrzeit) aufgeführt, an dem der Warteschlangenmanager die Nachricht in die Warteschlange für nicht zustellbare Nachrichten eingereiht hat.

Auch Anwendungen können die Warteschlange für Nachrichten verwenden, die sie nicht zustellen können. Weitere Informationen finden Sie im Abschnitt Verwendung der Warteschlange für nicht zustellbare Nachrichten.

**Systembefehlswarteschlange**

Eine *Systembefehlswarteschlange* ist eine Warteschlange, an die entsprechend berechtigte Anwendungen IBM MQ-Befehle senden können. Je nachdem, welche Befehle auf Ihrer Plattform unterstützt werden, empfangen diese Warteschlangen PCF-, MQSC- und CL-Befehle, die dann vom Warteschlangenmanager bearbeitet werden können.

z/OS Unter IBM MQ for z/OS heißt die Warteschlange `SYSTEM.COMMAND.INPUT` ; Auf anderen Plattformen heißt sie `SYSTEM.ADMIN.COMMAND.QUEUE`. Welche Befehle akzeptiert werden, hängt von der Plattform ab. Details hierzu finden Sie im Abschnitt Programmierbare Befehlsformate-Referenz.

**Systemstandardwarteschlangen**

In den *Systemstandardwarteschlangen* sind die ursprünglichen Definitionen der Warteschlangen für Ihr System enthalten. Wenn Sie eine Warteschlangendefinition erstellen, wird die Definition vom Warteschlangenmanager aus der entsprechenden Systemstandardwarteschlange kopiert. Die Vorgehensweise beim Erstellen einer Warteschlangendefinition ist anders als beim Erstellen einer dynamischen Warteschlange. Die Definition der dynamischen Warteschlange basiert auf der als Schablone für die dynamische Warteschlange ausgewählten Modellwarteschlange.

**Ereigniswarteschlangen**
In *Ereigniswarteschlangen* sind Ereignisnachrichten enthalten. Diese Nachrichten werden vom Warteschlangenmanager oder einem Kanal gemeldet.

## *Ferne Warteschlangen*

Eine Warteschlange ist für ein Programm *fern*, wenn sie zu einem Warteschlangenmanager gehörig ist, mit dem das Programm nicht verbunden ist.

Wenn eine Kommunikationsverbindung eingerichtet ist, kann ein Programm eine Nachricht an eine ferne Warteschlange senden. Nicht möglich ist es dagegen, dass ein Programm eine Nachricht aus einer fernen Warteschlange abruft.

In dem bei der Definition einer fernen Warteschlange erstellten Warteschlangendefinitionsobjekt sind nur die Informationen enthalten, die erforderlich sind, damit der lokale Warteschlangenmanager die Warteschlange, an welche die Nachricht gehen soll, ausfindig machen kann. Dieses Objekt wird auch als *lokale Definition einer fernen Warteschlange* bezeichnet. Alle Attribute der fernen Warteschlange sind auf dem Warteschlangenmanager gespeichert, der Eigner der Warteschlange ist, da es sich für diesen Warteschlangenmanager um eine lokale Warteschlange handelt.

Beim Öffnen einer fernen Warteschlange ist zur Identifikation der Warteschlange eine der folgenden Angaben erforderlich:

- Der Name der lokalen Definition der fernen Warteschlange. Vom Blickpunkt einer Anwendung aus besteht kein Unterschied zum Öffnen einer lokalen Warteschlange. Für die Anwendung ist es nicht relevant, ob es sich um eine lokale oder eine ferne Warteschlange handelt.

  Verwenden Sie den Befehl DEFINE QREMOTE , um auf allen Plattformen außer IBM ieine lokale Definition einer fernen Warteschlange zu erstellen.

  IBM i Verwenden Sie unter IBM i den Befehl CRTMQMQ.
- Der Name des fernen Warteschlangenmanagers und der Name, unter dem dieser ferne Warteschlangenmanager die Warteschlange kennt.

Lokale Definitionen ferner Warteschlangen umfassen zusätzlich zu den im Abschnitt „Warteschlangenattribute” auf Seite 21 beschriebenen allgemeinen Attributen drei weitere Attribute. Bei diesen drei Attributen handelt es sich um die folgenden:

**RemoteQName**
Der Name, unter dem der Warteschlangenmanager, zu dem die Warteschlange gehört, sie kennt.

**RemoteQMgrName**
Der Name des Warteschlangenmanagers, zu dem die Warteschlange gehört.

**XmitQName**
Der Name der lokalen Übertragungswarteschlange, die zur Weiterleitung von Nachrichten an andere Warteschlangenmanager verwendet wird.

Weitere Informationen zu diesen Attributen finden Sie im Abschnitt Attribute für Warteschlangen.

Wenn Sie den Aufruf MQINQ für die lokale Definition einer fernen Warteschlange verwenden, gibt der Warteschlangenmanager nur die Attribute der lokalen Definition zurück, also die Namen der fernen Warteschlange, des fernen Warteschlangenmanagers und der Übertragungswarteschlange. Die Attribute der entsprechenden lokalen Warteschlange im fernen System werden nicht zurückgegeben.

Weitere Informationen finden Sie auch im Abschnitt Übertragungswarteschlangen.

### *Aliaswarteschlangen*

Eine *Aliaswarteschlange* ist ein IBM MQ-Objekt, mit dessen Hilfe der Zugriff auf eine andere Warteschlange oder ein Thema möglich ist. Somit können mehrere Programme mit derselben Warteschlange arbeiten, indem sie für den Zugriff jeweils einen anderen Namen verwenden.

Die aus der Auflösung eines Aliasnamens resultierende Warteschlange, die sogenannte Basiswarteschlange, kann einer der folgenden Warteschlangentypen sein, wie von der Plattform unterstützt:

- Eine lokale Warteschlange
- die lokale Definition einer fernen Warteschlange
- z/OS Eine gemeinsam genutzte Warteschlange, ein lokaler Warteschlangentyp, der nur in IBM MQ for z/OS verfügbar ist.
- eine vordefinierte Warteschlange
- eine dynamische Warteschlange

Ein Aliasname kann auch in ein Thema aufgelöst werden. Wenn eine Anwendung gerade Nachrichten in eine Warteschlange einreiht, kann der Warteschlangenname zu einem Alias für ein Thema gemacht und die Anwendung auf diese Weise zu Veröffentlichungen zu dem Thema gebracht werden. Hierfür muss der Anwendungscode nicht geändert werden.

**Anmerkung:** Ein Alias kann nicht direkt in ein anderes Alias auf demselben Warteschlangenmanager aufgelöst werden.

Ein Beispiel für die Verwendung von Aliaswarteschlangen: Ein Systemadministrator erteilt für den Basiswarteschlangennamen (also die Warteschlange, in die der Aliasname aufgelöst wird) und den Aliaswarteschlangennamen verschiedene Zugriffsberechtigungen. Es kann dann passieren, dass Programme oder Benutzer berechtigt sind, die Aliaswarteschlange zu verwenden, nicht jedoch die Basiswarteschlange.

Alternativ dazu kann die Berechtigung so definiert werden, dass Einreihoperationen für die Basiswarteschlange erlaubt, für den Aliasnamen dagegen nicht zulässig sind.

Bei manchen Anwendungen bietet sich Systemadministratoren durch die Verwendung von Aliaswarteschlangen die Möglichkeit, ganz einfach die Definition eines Aliaswarteschlangenobjekts zu ändern, ohne dass eine Änderung der Anwendung erforderlich ist.

Wenn Programme versuchen, den Aliasnamen zu verwenden, nimmt IBM MQ Berechtigungsprüfungen für diesen Namen vor. Dabei wird nicht geprüft, ob das Programm auch zum Zugriff auf den Namen berechtigt ist, in den der Aliasname aufgelöst wird. Es kann also sein, dass ein Programm zum Zugriff auf einen Aliaswarteschlangennamen berechtigt ist, jedoch über keine Berechtigung für den aufgelösten Warteschlangennamen verfügt.

Zusätzlich zu den im Abschnitt „Warteschlangen” auf Seite 21 beschriebenen allgemeinen Warteschlangenattributen gibt es zu Aliaswarteschlangen noch das Attribut **BaseQName**. Hierbei handelt es sich um den Basiswarteschlangennamen, in den der Aliasname aufgelöst wird. Eine detailliertere Beschreibung dieser Attribute finden Sie im Abschnitt BaseQName (MQCHAR48).

Die Aliaswarteschlangenattribute *InhibitGet* und **InhibitPut** (siehe Abschnitt „Warteschlangen” auf Seite 21) gehören zum Aliasnamen. Wird beispielsweise der Aliaswarteschlangenname ALIAS1 in den Basiswarteschlangennamen BASE aufgelöst, gelten Beschränkungen für ALIAS1 nur für ALIAS1 und nicht für BASE. Beschränkungen für BASE dagegen gelten auch für ALIAS1.

Auch die Attribute *DefPriority* und **DefPersistence** gehören zum Aliasnamen. Somit können Sie beispielsweise verschiedenen Aliassen derselben Basiswarteschlange unterschiedliche Standardprioritäten zuordnen. Diese Prioritäten können außerdem geändert werden, ohne dass eine Änderung der Anwendungen, welche die Aliasse verwenden, erforderlich ist.

### *Dynamische und Modellwarteschlangen*

In diesem Abschnitt erhalten Sie einen Einblick in dynamische Warteschlangen, Sie finden hier Erläuterungen zu den Eigenschaften temporärer und permanenter dynamischer Warteschlangen, zur Verwendung dynamischer Warteschlangen und zu dabei besonders zu berücksichtigenden Aspekten sowie zu Modellwarteschlangen.

Wenn ein Anwendungsprogramm einen Aufruf MQOPEN absetzt, um eine Modellwarteschlange zu öffnen, erstellt der Warteschlangenmanager mit den Attributen der Modellwarteschlange dynamisch eine Instanz einer lokalen Warteschlange. Je nachdem, welcher Wert im Feld *DefinitionType* der Modellwarteschlange angegeben ist, erstellt der Warteschlangenmanager eine temporäre oder eine permanente dynamische Warteschlange (siehe Dynamische Warteschlangen erstellen).

**Eigenschaften temporärer dynamischer Warteschlangen**

*Temporäre dynamische Warteschlangen* weisen folgende Eigenschaften auf:

- z/OS Die Warteschlangen können nicht gemeinsam genutzt werden, es ist also kein Zugriff von Warteschlangenmanagern aus einer Gruppe mit gemeinsamer Warteschlange möglich.

  Beachten Sie, dass Gruppen mit gemeinsamer Warteschlange nur in IBM MQ for z/OS verfügbar sind.
- Sie enthalten nur nicht persistente Nachrichten.
- Sie sind nicht wiederherstellbar.
- Beim Start des Warteschlangenmanagers werden sie gelöscht.
- Sie werden auch gelöscht, wenn die Anwendung, die den Aufruf MQOPEN zum Erstellen der Warteschlange ausgegeben hat, die Warteschlange schließt oder selbst beendet wird.
  - Eventuell vorhandene festgeschriebene Nachrichten in der Warteschlange werden gelöscht.
  - Stehen zu diesem Zeitpunkt nicht festgeschriebene Aufrufe MQGET, MQPUT oder MQPUT1 für die Warteschlange aus, wird die Warteschlange als logisch gelöscht markiert und wird erst physisch gelöscht (nach Festschreibung dieser Aufrufe), wenn der Vorgang zum Schließen der Warteschlange verarbeitet oder die Anwendung beendet wird.
  - Wird die Warteschlange zu diesem Zeitpunkt gerade verwendet (von der erstellenden oder einer anderen Anwendung), wird sie als logisch gelöscht markiert und erst physisch gelöscht, wenn sie von der letzten Anwendung, welche die Warteschlange verwendet, geschlossen wird.
  - Zugriffsversuche auf eine logisch gelöschte Warteschlange schlagen mit dem Ursachencode MQRC_Q_DELETED fehl (es sei denn, der Zugriff erfolgt zum Schließen der Warteschlange).
  - MQCO_NONE, MQCO_DELETE und MQCO_DELETE_PURGE werden alle als MQCO_NONE behandelt, wenn sie in einem Aufruf MQCLOSE zu dem entsprechenden Aufruf MQOPEN angegeben sind, mit dem die Warteschlange erstellt wurde.

**Eigenschaften permanenter dynamischer Warteschlangen**

*Permanente dynamische Warteschlangen* weisen folgende Eigenschaften auf:

- Sie enthalten persistente und nicht persistente Nachrichten.
- Im Falle eines Systemfehlers sind sie wiederherstellbar.
- Sie werden gelöscht, wenn sie von einer Anwendung mit der Option MQCO_DELETE oder MQCO_DELETE_PURGE erfolgreich geschlossen werden. (Bei dieser Anwendung muss es sich nicht zwangsläufig um die Anwendung handeln, die den Aufruf MQOPEN zum Erstellen der Warteschlange ausgegeben hat.)
  - Falls sich noch (festgeschriebene oder nicht festgeschriebene) Nachrichten in der Warteschlange befinden, schlägt eine Schließanforderung mit der Option MQCO_DELETE fehl. Eine Schließanforderung mit der Option MQCO_DELETE_PURGE kann selbst dann erfolgreich ausgeführt werden, wenn sich festgeschriebene Nachrichten in der Warteschlange befinden (wobei die Nachrichten im Rahmen des Schließvorgangs gelöscht werden), sie schlägt jedoch fehl, wenn für die Warteschlange nicht festgeschriebene Aufrufe MQGET, MQPUT oder MQPUT1 ausstehen.
  - Wenn die Löschanforderung erfolgreich ist, die Warteschlange jedoch gerade (von der erstellenden oder einer anderen Anwendung) verwendet wird, wird sie als logisch gelöscht markiert und erst physisch gelöscht, wenn sie von der letzten Anwendung, die die Warteschlange verwendet, geschlossen wird.
- Die Warteschlangen werden nicht gelöscht, wenn sie von einer Anwendung geschlossen werden, die nicht zum Löschen der Warteschlange berechtigt ist, es sei denn, diese Anwendung hat den

Aufruf MQOPEN zum Erstellen der Warteschlange ausgegeben. Für die Benutzer-ID (bzw. bei Angabe von MQOO_ALTERNATE_USER_AUTHORITY für die alternative Benutzer-ID), mit welcher der entsprechende Aufruf MQOPEN validiert wurde, werden Berechtigungsprüfungen vorgenommen.

- Sie können wie normale Warteschlangen gelöscht werden.

**Verwendung dynamischer Warteschlangen**

Dynamische Warteschlangen können in folgenden Fällen eingesetzt werden:

- Für Anwendungen, nach deren Beendigung keine Warteschlangen beibehalten werden müssen.
- Für Anwendungen, bei denen Antworten auf Nachrichten von einer anderen Anwendung verarbeitet werden müssen. Solche Anwendungen können durch Öffnen einer Modellwarteschlange dynamisch eine Empfangswarteschlange für Antworten erstellen. Eine Clientanwendung kann beispielsweise folgende Aktionen ausführen:

  1. Eine dynamische Warteschlange erstellen.
  2. Im Feld **ReplyToQ** der Nachrichtendeskriptorstruktur der Anforderungsnachricht deren Namen angeben.
  3. Die Anforderung in eine von einem Server verarbeitete Warteschlange stellen.

Der Server kann die Antwortnachricht dann in die Empfangswarteschlange für Antworten einreihen. Der Client wiederum kann nun die Antwort verarbeiten und die Empfangswarteschlange für Antworten mit der Löschoption schließen.

**Bei der Verwendung dynamischer Warteschlangen zu berücksichtigende Aspekte**

Bei der Verwendung dynamischer Warteschlangen sind folgende Punkte zu beachten:

- In einem Client-Server-Modell müssen die einzelnen Clients jeweils eine eigene dynamische Empfangswarteschlange für Antworten erstellen und verwenden. Falls eine dynamische Empfangswarteschlange für Antworten von mehreren Clients gemeinsam genutzt wird, kann es sein, dass die Empfangswarteschlange für Antworten erst mit Verzögerung gelöscht wird, da für die Warteschlange nicht festgeschriebene Aktivitäten ausstehen oder die Warteschlange gerade von einem anderen Client verwendet wird. Außerdem wird die Warteschlange möglicherweise als logisch gelöscht markiert. Nachfolgende API-Anforderungen (außer MQCLOSE) können dann nicht mehr auf die Warteschlange zugreifen.
- Müssen in Ihrer Anwendungsumgebung dynamische Warteschlangen von mehreren Anwendungen gemeinsam genutzt werden, ist darauf zu achten, dass die Warteschlange erst geschlossen wird (mit der Löschoption), wenn die gesamte Aktivität für die Warteschlange festgeschrieben wurde. Diese Aktion sollte der letzte Benutzer ausführen. Auf diese Weise wird gewährleistet, dass die Warteschlange ohne Verzögerungen gelöscht wird. Außerdem wird der Zeitraum, in dem die Warteschlange nicht zugänglich ist, da sie als logisch gelöscht markiert ist, auf ein Minimum reduziert.

## Modellwarteschlangen

Eine *Modellwarteschlange* ist eine Schablone einer Warteschlangendefinition, die beim Erstellen einer dynamischen Warteschlange verwendet wird.

Von einem IBM MQ-Programm aus kann dynamisch eine lokale Warteschlange erstellt werden, wobei die als Schablone für die Warteschlangenattribute zu verwendende Modellwarteschlange angegeben wird. Zu diesem Zeitpunkt können einige Attribute der neuen Warteschlange geändert werden. Eine Änderung des Attributs **DefinitionType** ist jedoch nicht möglich. Falls Sie also beispielsweise eine permanente Warteschlange benötigen, wählen Sie eine Modellwarteschlange aus, für die als Definitionstyp 'permanent' angegeben ist. Manche Dialoganwendungen können für Antworten auf ihre Anfragen dynamische Warteschlangen verwenden, da diese Warteschlangen nach Verarbeitung der Antworten vermutlich nicht beibehalten werden müssen.

Den Namen einer Modellwarteschlange geben Sie im *Objektdeskriptor* (MQOD) des Aufrufs MQOPEN an. Unter Verwendung der Attribute der Modellwarteschlange wird vom Warteschlangenmanager dynamisch eine lokale Warteschlange erstellt.

Sie können einen (vollständigen) Namen für die dynamische Warteschlange angeben oder aber nur den Namensstamm (z. B. ABC), dem dann vom Warteschlangenmanager ein eindeutiger Teil hinzugefügt wird, Sie können aber auch die Zuordnung eines eindeutigen Namens komplett dem Warteschlangenmanager überlassen. Falls der Warteschlangenmanager den Namen zuordnet, stellt er ihn in die MQOD-Struktur.

Es ist nicht möglich, direkt für eine Modellwarteschlange einen Aufruf MQPUT1 auszugeben, für die dynamische Warteschlange, die durch Öffnen einer Modellwarteschlange erstellt wurde, kann der Aufruf MQPUT1 dagegen ausgegeben werden.

MQSET und MQINQ können nicht an einer Modellwarteschlange ausgeführt werden. Wenn Sie eine Modellwarteschlange mit MQOO_INQUIRE oder MQOO_SET öffnen, werden an der dynamisch erstellten Warteschlange weitere MQINQ- und MQSET-Aufrufe ausgeführt.

Bei den Attributen einer Modellwarteschlange handelt es sich um eine Untergruppe der Attribute einer lokalen Warteschlange. Eine detailliertere Beschreibung finden Sie im Abschnitt Attribute für Warteschlangen.

### *Für bestimmte Zwecke von IBM MQ verwendete Warteschlangen*

In IBM MQ werden einige lokale Warteschlangen für bestimmte Aspekte des Betriebsablaufs verwendet.

Diese Warteschlangen müssen erst definiert werden, damit sie von IBM MQ verwendet werden können.

**Initialisierungswarteschlangen**

Initialisierungswarteschlangen werden für die Auslösefunktion (Triggering) eingesetzt. Wenn ein Auslöserereignis eintritt, reiht der Warteschlangenmanager eine Auslösenachricht in eine Initialisierungswarteschlange ein. Ein Auslöserereignis ist eine logische Kombination an Bedingungen, die von einem Warteschlangenmanager erkannt wird. So kann beispielsweise ein Auslöserereignis generiert werden, wenn die Anzahl der Nachrichten in einer Warteschlange die vorgegebene Warteschlangengröße erreicht hat. Bei Eintreten dieses Ereignisses reiht der Warteschlangenmanager eine Auslösenachricht in die angegebene Initialisierungswarteschlange ein. Diese Auslösenachricht wird von einem *Auslösemonitor* abgerufen, bei dem es sich um eine besondere Anwendung für die Überwachung einer Initialisierungswarteschlange handelt. Der Auslösemonitor startet daraufhin das in der Auslösenachricht angegebene Anwendungsprogramm.

Soll die Auslösefunktion in einem Warteschlangenmanager verwendet werden, muss mindestens eine Initialisierungswarteschlange für ihn definiert sein. Weitere Informationen finden Sie in den Abschnitten Objekte für Auslösefunktion verwalten, runmqtrm und IBM MQ-Anwendungen mithilfe von Auslösern starten

**Übertragungswarteschlangen**

In Übertragungswarteschlangen werden kurzfristig Nachrichten gespeichert, die an einen fernen Warteschlangenmanager gesendet werden sollen. Sie müssen zumindest eine Übertragungswarteschlange für jeden fernen Warteschlangenmanager definieren, an den der lokale Warteschlangenmanager direkt Nachrichten senden soll. Diese Warteschlangen werden auch bei der Fernverwaltung verwendet (siehe Fernverwaltung von einem lokalen Warteschlangenmanager aus). Informationen zur Verwendung von Übertragungswarteschlangen in der verteilten Steuerung von Warteschlangen finden Sie im Abschnitt IBM MQ-Verfahren für verteilte Steuerung von Warteschlangen.

Für jeden Warteschlangenmanager kann eine Standard-Übertragungs-WS definiert werden. Wenn ein Warteschlangenmanager, der zu keinem Cluster gehört, eine Nachricht in eine ferne Warteschlange einreiht, wird standardmäßig die Standard-Übertragungs-WS verwendet. Wenn eine Übertragungswarteschlange mit demselben Namen wie dem des Ziel-Warteschlangenmanagers vorhanden ist, wird die Nachricht in diese Übertragungswarteschlange gestellt. Ist eine Definition eines Warteschlangenmanager-Aliasnamens vorhanden, deren Parameter **RQMNAME** mit dem Zielwarteschlangenmanager übereinstimmt, und ist der Parameter **XMITQ** angegeben, wird die Nachricht in die in **XMITQ** benannte Übertragungswarteschlange eingereiht. Wenn kein **XMITQ**-Parameter vorhanden ist, wird die Nachricht in die in der Nachricht angegebene lokale Warteschlange gestellt.

**Clusterübertragungswarteschlangen**

Zu jedem Warteschlangenmanager in einem Cluster gibt es eine Clusterübertragungswarteschlange `SYSTEM.CLUSTER.TRANSMIT.QUEUE` und eine Modell-Clusterübertragungswarteschlange `SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE`. Definitionen dieser Warteschlangen werden standardmä-

ßig bei der Definition eines Warteschlangenmanagers erstellt. Wenn das Warteschlangenmanager-Attribut **DEFCLXQ** auf CHANNEL gesetzt ist, wird automatisch eine permanente dynamische Cluster-Übertragungs-WS für jeden erstellten Clustersenderkanal erstellt. Die Warteschlangen heißen SYSTEM.CLUSTER.TRANSMIT. *ChannelName*. Cluster-Übertragungs-WS können auch manuell definiert werden.

Ein Warteschlangenmanager, der zu einem Cluster gehört, sendet Nachrichten in einer dieser Warteschlangen an andere Warteschlangenmanager im selben Cluster.

Bei der Namensauflösung hat eine Cluster-Übertragungs-WS Vorrang vor der standardmäßigen Übertragungswarteschlange und eine spezifische Cluster-Übertragungs-WS hat Vorrang vor SYSTEM.CLUSTER.TRANSMIT.QUEUE.

**Warteschlangen für nicht zustellbare Nachrichten**
In Warteschlangen für nicht zustellbare Nachrichten werden Nachrichten gespeichert, die nicht an ihr vorgegebenes Ziel gesendet werden können. Eine Nachricht kann nicht weitergeleitet werden, wenn beispielsweise die Zielwarteschlange voll ist. Die bereitgestellte Warteschlange für nicht zustellbare Nachrichten hat den Namen SYSTEM.DEAD.LETTER.QUEUE.

Bei einer verteilten Steuerung von Warteschlangen muss für jeden beteiligten Warteschlangenmanager eine Warteschlange für nicht zustellbare Nachrichten definiert werden.

**Befehlswarteschlangen**
Die Befehlswarteschlange SYSTEM.ADMIN.COMMAND.QUEUE ist eine lokale Warteschlange, an die entsprechend berechtigte Anwendungen WebSphere MQ-Scriptbefehle zur Verarbeitung senden können. Diese Befehle werden anschließend von einer IBM MQ-Komponente, dem sogenannten Befehlsserver, abgerufen. Der Befehlsserver überprüft die Befehle, sendet die gültigen Befehle weiter zur Verarbeitung durch den Warteschlangenmanager und gibt Antworten an die entsprechende Empfangswarteschlange für Antworten zurück.

Die Befehlswarteschlange wird automatisch bei der Erstellung eines Warteschlangenmanagers erstellt.

**Empfangswarteschlangen für Antworten**
Wenn eine Anwendung eine Anforderungsnachricht sendet, kann die Anwendung, die diese Nachricht enthält, eine Antwortnachricht an die Anwendung zurückgeben, von der die Nachricht stammt. Diese Nachricht wird in eine sogenannte Empfangswarteschlange für Antworten eingereiht, bei der es sich in der Regel um eine lokale Warteschlange der Anwendung handelt, die die Nachricht sendet. Der Name der Empfangswarteschlange wird von der sendenden Anwendung als Teil des Nachrichtendeskriptors angegeben.

**Ereigniswarteschlangen**
Mithilfe von Instrumentierungsereignissen können Warteschlangenmanager unabhängig von MQI-Anwendungen überwacht werden.

Bei Eintreten eines Instrumentierungsereignisses reiht der Warteschlangenmanager eine Ereignisnachricht in eine Ereigniswarteschlange ein. Diese Nachricht kann von einer Überwachungsanwendung gelesen werden, die entweder den Administrator informieren oder aber eine Korrekturmaßnahme einleiten kann, wenn das Ereignis auf ein Problem hindeutet.

**Anmerkung:** Auslöserereignisse und Instrumentierungsereignisse sollten nicht verwechselt werden. Auslöserereignisse treten unter anderen Bedingungen auf und generieren auch keine Ereignisnachrichten.

Weitere Informationen zu Instrumentierungsereignissen finden Sie im Abschnitt Instrumentierungsereignisse.

## Warteschlangenmanager

Dieser Abschnitt enthält eine Einführung in *Warteschlangenmanager* und die Services für die Warteschlangensteuerung, die sie für die Anwendungen bereitstellen.

Damit ein Programm die Services eines Warteschlangenmanagers nutzen kann, muss es eine Verbindung zu diesem Warteschlangenmanager herstellen. Diese Verbindung kann vom Programm direkt (über den

MQCONN- oder MQCONNX-Aufruf) oder indirekt (abhängig von der Plattform und der Umgebung, auf der bzw. in der das Programm aktiv ist) hergestellt werden.

Ein IBM MQ -Warteschlangenmanager stellt die folgenden Aktionen sicher:

- Objektattribute entsprechend den empfangenen Befehlen geändert werden;
- besondere Ereignisse wie Auslöser- oder Instrumentierungsereignisse generiert werden, wenn die entsprechenden Bedingungen erfüllt sind;
- Nachrichten entsprechend dem MQPUT-Aufruf einer Anwendung in die richtige Warteschlange eingereiht werden. Ist dies nicht möglich, wird die Anwendung unter Angabe des entsprechenden Ursachencodes darüber informiert.

Jede Warteschlange gehört nur zu einem Warteschlangenmanager und wird als *lokale Warteschlange* dieses Warteschlangenmanagers bezeichnet. Der Warteschlangenmanager, mit dem die Anwendung verbunden ist, wird als *lokaler Warteschlangenmanager* dieser Anwendung bezeichnet. Für die Anwendung sind die Warteschlangen ihres lokalen Warteschlangenmanagers lokale Warteschlangen.

Eine *ferne Warteschlange* ist eine Warteschlange, die zu einem anderen Warteschlangenmanager gehört. Als *fern* werden alle Warteschlangenmanager bezeichnet, bei denen es sich nicht um den lokalen Warteschlangenmanager handelt. Ein ferner Warteschlangenmanager kann sich auf einer fernen Maschine im Netz oder auf derselben Maschine wie der lokale Warteschlangenmanager befinden. IBM MQ unterstützt das Vorhandensein mehrerer Warteschlangenmanager auf ein und derselben Maschine.

In einigen MQI-Aufrufen können Warteschlangenmanagerobjekte verwendet werden. So können Sie beispielsweise mit dem MQI-Aufruf MQINQ die Attribute des Warteschlangenmanager-Objekts abfragen.

## Attribute von Warteschlangenmanagern

Jedem Warteschlangenmanager sind eine Reihe von Attributen (oder Eigenschaften) zugeordnet, die diesen Warteschlangenmanager definieren.Einige dieser Attribute werden bei der Erstellung eines Warteschlangenmanagers festgelegt; andere Attribute hingegen können über IBM MQ-Befehle geändert werden. Mit Ausnahme der für die TLS-Verschlüsselung (Transport Layer Security) verwendeten Attribute können Sie die Werte aller Attribute mit dem MQINQ-Aufruf abfragen.

Zu den festgelegten Attributen gehören:

- Der Name des Warteschlangenmanagers.
- Die Plattform, auf der der Warteschlangenmanager aktiv ist (z. B. Windows).
- Die Systemsteuerbefehle, die der Warteschlangenmanager unterstützt.
- Die Priorität, die den vom Warteschlangenmanager verarbeiteten Nachrichten maximal zugeordnet werden kann.
- Der Name der Warteschlange, an die Programme IBM MQ-Befehle senden können.
- Die maximale Länge der Nachrichten, die der Warteschlangenmanager verarbeiten kann z/OS (nur in IBM MQ for z/OS festgelegt)
- Ob der Warteschlangenmanager beim Einreihen und Abrufen von Nachrichten Synchronisationspunkte unterstützt.

Zu den Attribute, die *geändert* werden können, gehören die folgenden:

- Eine Beschreibung des Warteschlangenmanagers im Textformat.
- Die ID des Zeichensatzes, den der Warteschlangenmanager bei der Verarbeitung von MQI-Aufrufen für Zeichenfolgen verwendet.
- Das Zeitintervall, mit dem der Warteschlangenmanager die Anzahl von Auslösenachrichten einschränkt.
- z/OS Das Zeitintervall, mit dem der Warteschlangenmanager vorgibt, wie oft Warteschlangen auf abgelaufene Nachrichten durchsucht werden sollen (nur in IBM MQ for z/OS).
- Der Name der Warteschlange für nicht zustellbare Nachrichten, die dem Warteschlangenmanager zugeordnet ist.

- Der Name der Übertragungswarteschlange des Warteschlangenmanagers.
- Die maximale Anzahl geöffneter Kennungen für jede Verbindung.
- Die Aktivierung und Inaktivierung verschiedener Kategorien von Ereignisberichten.
- Die maximale Anzahl nicht festgeschriebener Nachrichten innerhalb einer Arbeitseinheit.

### Warteschlangenmanager und Workload-Management

Sie können einen Warteschlangenmanager-Cluster einrichten, für den mehrere Definitionen ein und derselben Warteschlange vorhanden sind (bei den Warteschlangenmanagern im Cluster kann es sich beispielsweise um Klone handeln). Nachrichten für eine bestimmte Warteschlange können von jedem Warteschlangenmanager verarbeitet werden, der über eine Instanz dieser Warteschlange verfügt. Mithilfe eines Workload-Management-Algorithmus wird der Warteschlangenmanager ermittelt, der die Nachricht verarbeiten soll. Auf diese Weise wird die Arbeitslast auf die einzelnen Warteschlangenmanager verteilt. Weitere Informationen hierzu finden Sie im Abschnitt Der Cluster-Workload-Management-Algorithmus.

## Kanäle

Ein *Kanal* ist eine logische Kommunikationsverbindung, die von verteilten Warteschlangenmanagern verwendet wird, zwischen einem IBM MQ MQI client und einem IBM MQ-Server oder zwischen zwei IBM MQ-Servern.

Kanäle werden für die Übertragung von Nachrichten zwischen Warteschlangenmanagern verwendet und schirmen Anwendungen von den verwendeten Kommunikationsprotokollen ab. Die Warteschlangenmanager können sich dabei auf demselben oder einem anderen System auf derselben Plattform oder auf verschiedenen Plattformen befinden. Die gesendeten Nachrichten können aus vielen Quellen stammen:

- Vom Benutzer geschriebene Anwendungsprogramme, die Daten von einem Knoten an einen anderen übertragen
- Vom Benutzer geschriebene Verwaltungsanwendungen, die PCF-Befehle oder die MQAI verwenden
- Die IBM MQ Explorer.
- Warteschlangenmanager, die Instrumentierungsereignisnachrichten an einen anderen Warteschlangenmanager senden
- Warteschlangenmanager, die Fernverwaltungsbefehle an einen anderen Warteschlangenmanager senden (z. B. mit MQSC-Befehlen oder über die administrative REST API)

Ein Kanal verfügt über zwei Definitionen, nämlich je eine an jedem Ende der Verbindung. Damit Warteschlangenmanager miteinander kommunizieren können, müssen Sie einen Kanal auf der Seite des Warteschlangenmanagers definieren, der Nachrichten senden soll, sowie einen Kanal auf der Seite des Warteschlangenmanagers, der diese Nachrichten empfangen soll. An beiden Enden muss derselbe *Kanalname* verwendet werden und die *Kanaltypen* müssen kompatibel sein.

In IBM MQ gibt es drei Kanalkategorien mit jeweils verschiedenen Kanaltypen:

- Unidirektionale Nachrichtenkanäle, über die Nachrichten von einem Warteschlangenmanager an einen anderen gesendet werden.
- Bidirektionale MQI-Kanäle, über die MQI-Aufrufe von einem IBM MQ MQI client an einen Warteschlangenmanager und Antworten von einem Warteschlangenmanager an einen IBM MQ-Client übertragen werden.
- Bidirektionale AMQP-Kanäle, die einen AMQP-Client mit einem Warteschlangenmanager auf einer Servermaschine verbinden. IBM MQ verwendet AMQP-Kanäle, um AMQP-Aufrufe und -Antworten zwischen AMQP-Anwendungen und Warteschlangenmanagern zu übertragen.

### Nachrichtenkanäle

Über Nachrichtenkanäle werden Nachrichten zwischen den Warteschlangenmanagern übertragen. In der Client/Server-Umgebung sind keine Nachrichtenkanäle erforderlich.

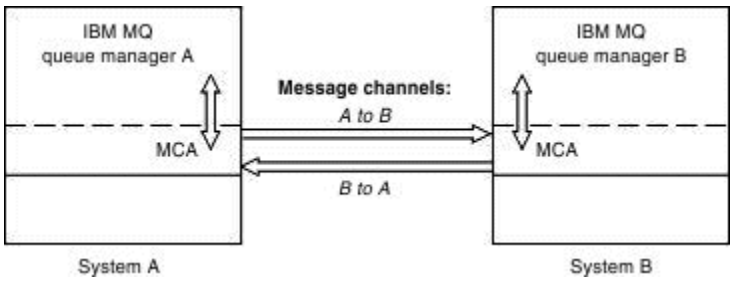



*Abbildung 2. Nachrichtenkanäle zwischen zwei Warteschlangenmanagern*

Ein Nachrichtenkanal ist eine unidirektionale Verbindung, Wenn ein ferner Warteschlangenmanager auf Nachrichten antworten soll, die von einem lokalen Warteschlangenmanager gesendet werden, müssen Sie einen zweiten Kanal einrichten, um Antworten zurück an den lokalen Warteschlangenmanager zu senden.

Ein Nachrichtenkanal verbindet zwei Warteschlangenmanager über *Nachrichtenkanalagenten*. An jedem Ende eines Kanals gibt es einen Nachrichtenkanalagenten. Sie können einem Nachrichtenkanalagenten ermöglichen, Nachrichten unter Verwendung mehrerer Threads zu übertragen. Dieser Prozess ist als *Pipelining* bekannt. Mithilfe von Pipelining kann der Nachrichtenkanalagent Nachrichten effizienter übertragen, was die Kanalleistung verbessert. Weitere Informationen zum Pipelining finden Sie im Abschnitt Attribute von Kanälen.

Weitere Informationen zu Kanälen finden Sie in den Abschnitten Kanalexitaufrufe und Datenstrukturen und „Komponenten der verteilten Steuerung von Warteschlangen” auf Seite 50.

## MQI-Kanäle

Ein MQI-Kanal (Message Queue Interface) verbindet einen IBM MQ MQI client mit einem Warteschlangenmanager auf einer Servermaschine und wird eingerichtet, wenn Sie einen `MQCONN` -oder `MQCONNX` -Aufruf von einer IBM MQ MQI client -Anwendung absetzen.

MQI-Kanäle sind bidirektionale Verbindungen, die nur für die Übertragung von MQI-Aufrufen und -Antworten verwendet werden (z. B. `MQPUT`-Aufrufe, die Nachrichtendaten enthalten, und `MQGET`-Aufrufe, mit denen Nachrichtendaten zurückgegeben werden). Es gibt verschiedene Möglichkeiten, Kanaldefinitionen zu erstellen und zu verwenden (siehe MQI-Kanäle definieren).

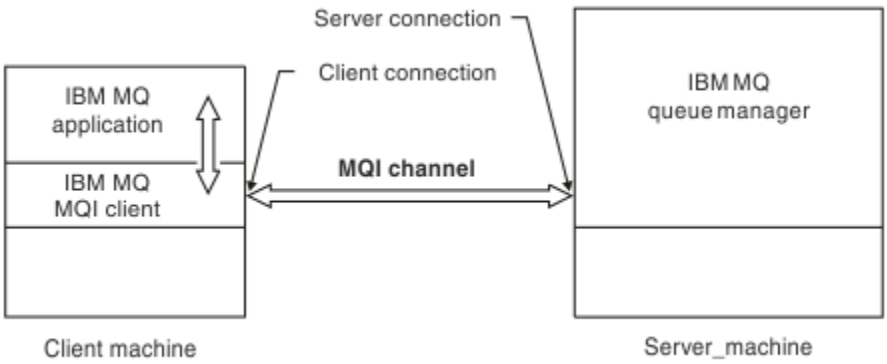



*Abbildung 3. Clientverbindung und Serververbindung über einen MQI-Kanal*

z/OS Mit einem MQI-Kanal kann ein Client mit einem einzelnen Warteschlangenmanager oder mit einem Warteschlangenmanager, der zu einer Gruppe mit gemeinsamer Warteschlange gehört, verbunden werden (siehe Client mit einer Gruppe mit gemeinsamer Warteschlange verbinden).

Es gibt zwei Kanaltypen für MQI-Kanaldefinitionen. Sie beschreiben den bidirektionalen MQI-Kanal.

**Clientverbindungskanal**
Dieser Typ ist für den IBM MQ MQI client.

**Serververbindungskanal**
Dieser Typ gilt für den Server, auf dem der Warteschlangenmanager ausgeführt wird, mit dem die IBM MQ -Anwendung in einer IBM MQ MQI client -Umgebung kommunizieren soll.

## AMQP-Kanäle


Es gibt nur einen AMQP-Kanaltyp.

Über den Kanal wird eine AMQP-Messaging-Anwendung mit einem Warteschlangenmanager verbunden, damit die Anwendung Nachrichten mit IBM MQ-Anwendungen austauschen kann. Ein AMQP-Kanal ermöglicht es Ihnen, mithilfe von MQ Light eine Anwendung zu entwickeln, die dann als Unternehmensanwendung eingesetzt werden kann, um die auf Unternehmen abgestimmten Funktionen, die von IBM MQ bereitgestellt werden, zu nutzen.

## Clientverbindungskanäle

*Clientverbindungskanäle* sind Objekte, die einen Kommunikationspfad von einem IBM MQ MQI client zu einem Warteschlangenmanager bereitstellen.

Clientkommunikationskanäle werden bei der verteilten Steuerung von Warteschlangen für die Übertragung von Nachrichten zwischen einem Warteschlangenmanager und einem Client verwendet. Sie schirmen Anwendungen von den verwendeten Kommunikationsprotokollen ab. Der Client kann sich auf derselben Plattform wie der Warteschlangenmanager oder auf einer anderen Plattform befinden.

## Kanal- definitionen

Beschreibungen der einzelnen Kanaltypen finden Sie im Abschnitt „Kanal- definitionen” auf Seite 34.

**Zugehörige Konzepte**

„Verteilte Warteschlangen und Cluster” auf Seite 46
Durch die verteilte Steuerung von Warteschlangen können Nachrichten von einem Warteschlangenmanager an einen anderen gesendet werden. Der empfangende Warteschlangenmanager kann sich auf demselben System oder auf einem anderen System ganz in der Nähe oder am anderen Ende der Welt befinden. Er kann auf derselben Plattform wie der lokale Warteschlangenmanager oder jeder anderen von IBM MQ unterstützten Plattform aktiv sein. In einer Umgebung mit verteilter Steuerung von Warteschlangen können Sie alle Verbindungen manuell definieren oder Sie können einen Cluster erstellen und IBM MQ den größten Teil der Verbindungskonfiguration überlassen.

Übersicht über Message Queue Interface

**Zugehörige Tasks**

Ferne IBM MQ-Objekte verwalten

MQI-Kanäle stoppen

Verbindungen zwischen dem Server und dem Client konfigurieren

**Zugehörige Verweise**

Kanalexitaufrufe und Datenstrukturen

„Kommunikation” auf Seite 39
IBM MQ MQI clients kommunizieren über MQI-Kanäle mit dem Server.

## *Kanal- definitionen*

Tabellen zur Beschreibung der verschiedenen Arten von Nachrichtenkanälen und MQI-Kanälen, die von IBM MQ verwendet werden.

Bei den Erläuterungen zu Nachrichtenkanälen wird das Wort Kanal gleichbedeutend mit dem Begriff Kanaldefinition verwendet. Aus dem Kontext geht im Allgemeinen hervor, ob von einem vollständigen Kanal (mit zwei Enden) oder von einer Kanaldefinition (die sich nur auf ein Kanalende bezieht) die Rede ist.

## Nachrichtenkanäle

Nachrichtenkanaldefinitionen können zu einer der folgenden Arten gehören:

| Art der Nachrichtenkanaldefinition | Beschreibung |
|---|---|
| Sender | Ein Senderkanal ist ein Nachrichtenkanal, über den ein Warteschlangenmanager Nachrichten an andere Warteschlangenmanager sendet. Damit Sie Nachrichten über einen Senderkanal senden können, müssen Sie auf dem anderen Warteschlangenmanager auch einen Empfangskanal mit demselben Namen wie der Senderkanal erstellen. Bei Verwendung eines Callback-Mechanismus kann der Senderkanal auch mit Requester-Kanälen verwendet werden. |
| Server | Ein Serverkanal ist ein Nachrichtenkanal, über den ein Warteschlangenmanager Nachrichten an andere Warteschlangenmanager sendet. Damit Sie Nachrichten über einen Serverkanal senden können, müssen Sie auf dem anderen Warteschlangenmanager auch einen Empfangskanal mit demselben Namen wie der Serverkanal erstellen. Sie können auch Serverkanäle in Verbindung mit Requester-Kanälen verwenden. In diesem Fall fordert die Requesterkanaldefinition am anderen Ende des Kanals die Serverkanaldefinition zum Start auf. Der Server sendet Nachrichten an den Requester. Auch der Server kann die Datenübertragung initiieren, sofern ihm der Verbindungsname des Partnerkanals bekannt ist. |
| Empfänger | Ein Empfängerkanal ist ein Nachrichtenkanal, über den ein Warteschlangenmanager Nachrichten von anderen Warteschlangenmanagern empfängt. Damit Sie Nachrichten über einen Empfangskanal empfangen können, müssen Sie auf dem anderen Warteschlangenmanager auch einen Senderkanal oder Serverkanal mit demselben Namen wie der Empfangskanal erstellen. |

| Art der Nachrichtenkanaldefinition | Beschreibung |
| --- | --- |
| Requester | Ein Requesterkanal ist ein Nachrichtenkanal, über den ein Warteschlangenmanager Nachrichten von anderen Warteschlangenmanagern empfängt. Ein Requesterkanal kann eine Startanforderung an den am fernen Ende definierten Partnerkanal senden. Wenn der Partnerkanal ein Serverkanal ist, akzeptiert der Serverkanal die Startanforderung und beginnt damit, Nachrichten aus der Übertragungswarteschlange, die in der Kanaldefinition des Servers angegeben ist, an den Requesterkanal zu senden. Wenn der Partnerkanal ein Senderkanal ist, akzeptiert der Senderkanal die Startanforderung, schließt aber dann die Verbindung mit dem Requester. Dann startet der Senderkanal, vereinbart eine Sitzung mit dem Partner-Requesterkanal und beginnt damit, Nachrichten aus der Übertragungswarteschlange zu senden, die in der Kanaldefinition des Senders angegeben ist. In diesem Fall wird im Wesentlichen ein Callback-Mechanismus bereitgestellt, indem der Requesterkanal eine Rückrufanforderung an den Senderkanal sendet. |
| Clustersender | Die Definition eines Clustersenderkanals (CLUSSDR) legt die Senderseite des Kanals fest, über den Clusterwarteschlangenmanager clusterspezifische Daten an eines der vollständigen Repositorys senden können. Der Clustersenderkanal wird dazu verwendet, das Repository über Änderungen am Status des Warteschlangenmanagers zu benachrichtigen, z. B. über das Hinzufügen oder Entfernen einer Warteschlange. Darüber hinaus wird dieser Kanal auch zur Nachrichtenübertragung verwendet. Die Warteschlangenmanager mit dem vollständigen Repository wiederum verfügen über Clustersenderkanäle, die auf den jeweils anderen Repository-Warteschlangenmanager verweisen. Über diese Kanäle informieren sie sich gegenseitig über Änderungen am Clusterstatus. Es ist unerheblich, auf welches vollständige Repository die Kanaldefinition 'CLUSSDR' eines Warteschlangenmanagers verweist. Nachdem der erste Kontakt hergestellt wurde, werden weitere Clusterwarteschlangenmanagerobjekte automatisch nach Bedarf definiert, sodass der Warteschlangenmanager Clusterinformationen an jedes vollständige Repository und Nachrichten an jeden Warteschlangenmanager senden kann. |

| Art der Nachrichtenkanaldefinition | Beschreibung |
|---|---|
| Clusterempfänger | Die Definition eines Clusterempfängerkanals (CLUSRCVR) legt die Empfängerseite des Kanals fest, über den Clusterwarteschlangenmanager Nachrichten von anderen Warteschlangenmanagern im Cluster empfangen können. Darüber hinaus können über Clusterempfängerkanäle auch Clusterdaten für die Repositorys übertragen werden. Indem der Warteschlangenmanager den Clusterempfängerkanal definiert, teilt er den anderen Warteschlangenmanagern dadurch mit, dass er für den Empfang von Nachrichten verfügbar ist. Für jeden Clusterwarteschlangenmanager ist mindestens ein Clusterempfängerkanal erforderlich. |

Für jeden Kanal müssen beide Kanalenden definiert werden, sodass für jedes Kanalende eine eigene Definition vorhanden ist. Die beiden Kanalenden müssen vom Typ her miteinander kompatibel sein.

Folgende Kombinationen von Kanaldefinitionen sind möglich:

- Sender-Empfänger
- Server-Empfänger
- Requester-Server
- Requester-Sender (Callback)
- Clustersender-Clusterempfänger

## Nachrichtenkanalagenten

Jede Kanaldefinition, die Sie erstellen, gehört zu einem bestimmten Warteschlangenmanager. Ein Warteschlangenmanager kann über mehrere Kanäle desselben Typs oder verschiedener Typen verfügen. An den beiden Kanalenden ist jeweils ein Programm installiert, der Nachrichtenkanalagent (MCA). Am einen Ende holt der aufrufende MCA Nachrichten aus der Übertragungswarteschlange und sendet diese über den Kanal. Am anderen Ende des Kanals empfängt der Responder-MCA die Nachrichten und leitet sie an den fernen Warteschlangenmanager weiter.

Ein aufrufender MCA kann einem Sende-, einem Server- oder einem Requesterkanal zugeordnet sein. Ein Responder-MCA kann jeder Nachrichtenkanalart zugeordnet werden.

In IBM MQ werden die folgenden Kombinationen von Kanaltypen an den beiden Verbindungsenden unterstützt:

| **Aufrufer** | | **Richtung des Nachrichtenflusses** | **Responder** | |
|---|---|---|---|---|
| **Kanaltyp** | **Empfangsprogramm erforderlich?** | | **Empfangsprogramm erforderlich?** | **Kanaltyp** |
| Sender | Nein | Aufrufer an Responder | Ja | Empfänger |
| Server | Nein | Aufrufer an Responder | Ja | Empfänger |
| Server | Nein | Aufrufer an Responder | Ja | Requester |
| Requester | Nein | Responder an Aufrufer | Ja | Server |

| Aufrufer | | Richtung des Nachrichtenflusses | Responder | |
|---|---|---|---|---|
| **Kanaltyp** | **Empfangsprogramm erforderlich?** | | **Empfangsprogramm erforderlich?** | **Kanaltyp** |
| Requester | Ja | Responder an Aufrufer | Ja | Sender |

## MQI-Kanäle

MQI-Kanäle können zu einem der folgenden Kanaltypen gehören:

| MQI-Kanaltyp | Beschreibung |
|---|---|
| Serververbindung | Ein Serververbindungskanal ist ein bidirektionaler MQI-Kanal, der eine Verbindung zwischen einem IBM MQ-Client und einem IBM MQ-Server herstellt. Dabei stellt der Serververbindungskanal das serverseitige Ende des Kanals dar. |
| Clientverbindung | Ein Clientverbindungskanal ist ein bidirektionaler MQI-Kanal, der eine Verbindung zwischen einem IBM MQ-Client und einem IBM MQ-Server herstellt. Im IBM MQ Explorer werden Clientverbindungen auch dazu verwendet, eine Verbindung zu fernen Warteschlangenmanagern herzustellen. Dabei stellt der Clientverbindungskanal das clientseitige Ende des Kanals dar. Beim Erstellen eines Clientverbindungskanals wird auf dem Computer, auf dem sich der Warteschlangenmanager befindet, eine Datei erstellt. Die Clientverbindungsdatei müssen Sie dann auf den IBM MQ-Clientcomputer kopieren. |

Multi

### *Unterstützung mehrerer Threads — Pipelining*

Sie haben die Option, einem Nachrichtenkanalagenten (MCA) die Übertragung von Nachrichten über mehrere Threads zu gestatten. Dieser Prozess, als *Pipelining* bezeichnet, ermöglicht dem MCA, Nachrichten effizienter mit weniger Wartestatus zu übertragen, wodurch sich die Kanalleistung verbessern lässt. Für jeden MCA können maximal zwei Threads ausgeführt werden.

Pipelining wird über den Parameter *PipeLineLength* in der Datei "qm.ini" gesteuert. Dieser Parameter wird der Zeilengruppe Channelshinzugefügt.

**Anmerkung:** Pipelining ist nur für TCP/IP-Kanäle effektiv.

Bei der Verwendung von Pipelining muss in der Konfiguration der Warteschlangenmanager an beiden Kanalenden der Parameter *PipeLineLength* auf einen größeren Wert als 1 festgelegt sein.

## Überlegungen zu Kanalexits

Pipelining kann aus folgenden Gründen bei einigen Exitprogrammen zu Ausfällen führen:

- Exits wurden nicht seriell aufgerufen.
- Exits werden von unterschiedlichen Threads abwechselnd aufgerufen.

Überprüfen Sie die Struktur Ihres Exitprogramms, bevor Sie Pipelining nutzen:

- Exits müssen auf allen Ausführungsstufen wiedereintrittsfähig sein.
- Wenn Sie MQI-Aufrufe verwenden, achten Sie darauf, nicht die gleiche MQI-Kennung zu verwenden, wenn der Exit aus verschiedenen Threads aufgerufen wird.

Ziehen Sie einen Nachrichtenexit in Betracht, der eine Warteschlange öffnet und bei allen nachfolgenden Aufrufen des Exits dessen Kennung für MQPUT-Aufrufe angibt. Dies ist im Pipelining-Modus nicht möglich, da der Exit aus verschiedenen Threads aufgerufen wird. Um dies zu vermeiden, halten Sie eine Warteschlangenkennung für jeden Thread bereit, und überprüfen Sie bei jedem Aufruf des Exits die Thread-ID.

## Kommunikation

IBM MQ MQI clients kommunizieren über MQI-Kanäle mit dem Server.

Auf der IBM MQ MQI client- und -Serverseite der Verbindung muss jeweils eine Kanaldefinition erstellt werden. Informationen zum Erstellen von Kanaldefinitionen finden Sie im Abschnitt MQI-Kanäle definieren.

In der folgenden Tabelle sind die möglichen Übertragungsprotokolle aufgeführt:

*Tabelle 1. Übertragungsprotokolle für MQI-Kanäle*

| Clientplattform | LU 6.2 | TCP/IP | NetBIOS | SPX |
|---|---|---|---|---|
| IBM i | | Ja | | |
| Linux AIX and Linux-Systeme | Ja[1] | Ja | | |
| Windows | Ja | Ja | Ja | Ja |

**Anmerkung:**

1. Linux LU6.2 wird auf folgenden Plattformen nicht unterstützt:
   - Linux (POWER-Plattform)
   - Linux (x86-64-Plattform)
   - Linux (zSeries s390x-Plattform)

In Übertragungsprotokolle - Kombination von IBM MQ MQI client- und Serverplattformen sind die möglichen Kombinationen von IBM MQ MQI client- und Serverplattformen mit diesen Übertragungsprotokollen aufgeführt.

Eine IBM MQ-Anwendung auf einem IBM MQ MQI client kann alle MQI-Aufrufe so verwenden, als ob ein lokaler Warteschlangenmanager verwendet wird. Der **MQCONN**- oder **MQCONNX**-Aufruf ordnet die IBM MQ-Anwendung dem ausgewählten Warteschlangenmanager zu und erstellt auf diese Weise eine *Verbindungskennung*. Alle weiteren Aufrufe, die diese Verbindungskennung verwenden, werden daraufhin von dem verbundenen Warteschlangenmanager verarbeitet. Im Unterschied zur Kommunikation der Warteschlangenmanager untereinander, die verbindungs- und zeitunabhängig erfolgt, ist für die IBM MQ MQI clientkommunikation eine aktive Verbindung zwischen Client und Server erforderlich.

Das Übertragungsprotokoll wird über die Verwendung der Kanaldefinition festgelegt und hat keine Auswirkung auf die Anwendung. So kann beispielsweise eine Windows-Anwendung zu einem Warteschlangenmanager eine Verbindung über TCP/IP und zu einem anderen über NetBIOS herstellen.

### Leistungsaspekte

Das von Ihnen verwendete Übertragungsprotokoll kann sich auf die Leistung des IBM MQ-Client/Server-Systems auswirken. In bestimmten Situationen, in denen die Übertragung langsam ist, können Sie die IBM MQ -Kanalkomprimierung verwenden.

# IBM MQ-Objekte benennen

Die Namenskonvention für die Benennung von IBM MQ-Objekten hängt vom jeweiligen Objekt ab. Auch für die Namen der Maschinen und die Benutzer-IDs für IBM MQ gelten einige Einschränkungen.

Jede Instanz eines Warteschlangenmanagers hat einen eigenen Namen. Dieser Name muss im Netz bzw. unter den verbundenen Warteschlangenmanagern eindeutig sein, damit ein Warteschlangenmanager eindeutig den Ziel-Warteschlangenmanager angeben kann, an den eine Nachricht gesendet werden soll.

Bei den anderen Objekttypen ist jedem Objekt ein Name zugewiesen, über den es identifiziert werden kann. Diese Namen müssen innerhalb eines Warteschlangenmanagers und innerhalb eines Objekttyps eindeutig sein. So können beispielsweise eine Warteschlange und ein Prozess denselben Namen haben, nicht jedoch zwei Warteschlangen.

In IBM MQ können Namen eine Länge von bis zu 48 Zeichen haben; ausgenommen hiervon sind *Kanalnamen*, die maximal eine Länge von 20 Zeichen haben können. Weitere Informationen zur Benennung von IBM MQ-Objekten finden Sie im Abschnitt „Regeln für die Benennung von IBM MQ-Objekten" auf Seite 40.

Für die Namen der Maschinen und die Benutzer-IDs für IBM MQ gelten ebenfalls einige Einschränkungen:

- Achten Sie darauf, dass der Maschinenname keine Leerzeichen enthält. Maschinennamen mit Leerzeichen werden von IBM MQ nicht unterstützt. Wenn Sie IBM MQ auf eine Maschine mit einem solchen Namen installieren, können Sie keine Warteschlangenmanager erstellen.
- Bei IBM MQ-Berechtigungen dürfen die Namen der Benutzer-IDs und -Gruppen nicht länger als 20 Zeichen sein (Leerzeichen sind nicht zulässig).
- Windows Von IBM MQ for Windows-Servern wird die Verbindung eines IBM MQ MQI clients nicht unterstützt, wenn der Client unter einer Benutzer-ID ausgeführt wird, die ein kommerzielles A (@) enthält (z. B. abc@d).

**Zugehörige Konzepte**

„IBM MQ-Dateinamen" auf Seite 44
Die einzelnen Warteschlangenmanager, Warteschlangen, Prozessdefinitionen, Namenslisten, Kanäle, Clientverbindungskanäle, Empfangsprogramme, Services und Authentifizierungsdatenobjekte in IBM MQ werden jeweils durch eine Datei dargestellt. Da Objektnamen nicht notwendigerweise auch gültige Dateinamen sind, wandelt der Warteschlangenmanager die Objektnamen bei Bedarf in gültige Dateinamen um.

**Zugehörige Verweise**

„Regeln für die Benennung von IBM MQ-Objekten" auf Seite 40
Für die Namen von IBM MQ-Objekten gelten Längeneinschränkungen. Außerdem wird bei diesen Namen die Groß-/Kleinschreibung beachtet. Nicht alle Zeichen werden für jeden Objekttyp unterstützt, und bei vielen Objekten gelten Regeln hinsichtlich der Eindeutigkeit der Namen.

## Regeln für die Benennung von IBM MQ-Objekten

Für die Namen von IBM MQ-Objekten gelten Längeneinschränkungen. Außerdem wird bei diesen Namen die Groß-/Kleinschreibung beachtet. Nicht alle Zeichen werden für jeden Objekttyp unterstützt, und bei vielen Objekten gelten Regeln hinsichtlich der Eindeutigkeit der Namen.

Es gibt viele verschiedene Arten von IBM MQ-Objekten, wobei Objekte unterschiedlicher Typen gleiche Namen haben können, da sie in getrennten Namensbereichen verwaltet werden. Eine lokale Warteschlange und ein Senderkanal können zum Beispiel den gleichen Namen haben. Ein Objekt darf jedoch nicht den Namen eines anderen Objekts des gleichen Namensbereichs haben. Eine lokale Warteschlange darf zum Beispiel nicht den gleichen Namen wie eine Modellwarteschlange haben, und ein Senderkanal darf nicht den gleichen Namen wie ein Empfängerkanal haben.

Die folgenden IBM MQ-Objekte liegen in separaten Namensbereichen vor:

- Authentifizierungsdaten
- Kanal
- Clientkanal
- Empfangsprogramm
- Namensliste
- Prozess

- Warteschlange
- Service
- Speicherklasse
- Abonnement
- Thema

## Zeichenlänge von Objektnamen

Im Allgemeinen dürfen die Namen von IBM MQ-Objekten bis zu 48 Zeichen lang sein. Diese Regel gilt für die folgenden Objekte:

- Authentifizierungsdaten
- Cluster
- Empfangsprogramm
- Namensliste
- Prozessdefinition
- Warteschlange
- Warteschlangenmanager
- Service
- Abonnement
- Thema

Allerdings gibt es hierzu auch Einschränkungen:

1. z/OS Unter z/OS müssen die Namen von Warteschlangenmanagern mindestens 4 Zeichen lang sein. Sie dürfen nur Großbuchstaben und numerische Zeichen enthalten.
2. Die Namen von Kanalobjekten und Clientverbindungskanälen dürfen maximal 20 Zeichen lang sein. Weitere Informationen zu Kanälen finden Sie im Abschnitt Kanäle definieren.
3. Themenzeichenfolgen dürfen maximal 10240 Byte lang sein. Bei allen Namen von IBM MQ-Objekten wird die Groß-/Kleinschreibung beachtet.
4. Subskriptionsnamen dürfen maximal 10240 Byte lang sein und können Leerzeichen enthalten.
5. Die Namen von Speicherklassen dürfen maximal 8 Zeichen lang sein.
6. Die Namen von Coupling-Facility-(CF-)Strukturen dürfen maximal 12 Zeichen lang sein.

## Zeichen in Objektnamen

Für die Namen von IBM MQ-Objekten sind folgende Zeichen gültig:

| Zeichen | Einschränkungen |
|---|---|
| Großbuchstaben (A - Z) | • -- |

| Zeichen | Einschränkungen |
| --- | --- |
| Kleinbuchstaben (a - z) | • In MQSC-Scripts müssen Namen mit Kleinbuchstaben in einfache Anführungszeichen eingeschlossen werden, da die Kleinbuchstaben ansonsten in Großbuchstaben umgewandelt werden.<br>• Systeme, auf denen EBCDIC Katakana verwendet wird, unterstützen in Objektnamen keine Kleinbuchstaben.<br>• z/OS Unter z/OS gelten für Kleinbuchstaben gewisse Einschränkungen. Die Namen von Warteschlangenmanagern dürfen zum Beispiel keine Kleinbuchstaben enthalten.<br>• IBM i Auf IBM i-Systemen müssen in CL-Befehlen Namen mit Kleinbuchstaben in einfache Anführungszeichen eingeschlossen werden, da die Kleinbuchstaben ansonsten in Großbuchstaben umgewandelt werden. |
| Numerische Zeichen (0 - 9) | • -- |
| Punkt (.) | • -- |
| Unterstrich (_) | • Multi --<br>• z/OS Vermeiden Sie Namen mit führenden oder abschließenden Unterstrichen, da diese von den Bedien- und Steuerkonsolen von IBM MQ for z/OS nicht verarbeitet werden können. |
| Schrägstrich (/) | • Windows Unter Windows darf das erste Zeichen eines Warteschlangenmanagernamens kein Schrägstrich sein.<br>• IBM i Auf IBM i-Systemen müssen in CL-Befehlen Namen mit Schrägstrichen in einfache Anführungszeichen eingeschlossen werden.<br>• z/OS -- |
| Prozentzeichen (%) | • ALW --<br>• z/OS Wenn Sie RACF als externen Sicherheitsmanager für IBM MQ for z/OS verwenden, geben Sie in Objekten nicht das Prozentzeichen an, da Objektnamen mit Prozentzeichen bei Verwenden der generischen Profile von RACF nicht in Sicherheitsprüfungen eingeschlossen werden.<br>• IBM i Auf IBM i-Systemen müssen in CL-Befehlen Namen mit Prozentzeichen in einfache Anführungszeichen eingeschlossen werden. |

Darüber hinaus gelten für die in Objektnamen verwendeten Zeichen einige generelle Regeln:

1. Führende oder eingebettete Leerzeichen sind nicht erlaubt.
2. Landessprachliche Zeichen sind nicht erlaubt.
3. Namen, die kürzer als die Feldlänge sind, können am Ende mit Leerzeichen aufgefüllt werden. Alle zu kurzen Namen, die vom Warteschlangenmanager zurückgegeben werden, sind am Ende mit Leerzeichen aufgefüllt.

## Warteschlangennamen

Der Name einer Warteschlange besteht aus zwei Teilen:

- Der Name eines Warteschlangenmanagers
- Aus dem lokalen Namen der Warteschlange, wie er dem Warteschlangenmanager bekannt ist

Jeder Teil des Warteschlangennamens ist 48 Zeichen lang.

Bei der Angabe einer lokalen Warteschlange können Sie den Namen des Warteschlangenmanagers weglassen (durch Ersetzen seines Namens durch Leerzeichen oder durch eine führende Null). Alle Warteschlangennamen, die von IBM MQ an ein Programm zurückgegeben werden, enthalten jedoch den Namen des Warteschlangenmanagers.

z/OS Eine gemeinsam genutzte Warteschlange, die für jeden Warteschlangenmanager der jeweiligen Gruppe mit gemeinsamer Warteschlange zugänglich ist, darf nicht den gleichen Namen haben wie eine nicht gemeinsam genutzte lokale Warteschlange der gleichen Gruppe mit gemeinsamer Warteschlange. Dadurch wird verhindert, dass eine Anwendung versehentlich eine gemeinsam genutzte Warteschlange öffnet, wenn sie eigentlich eine lokale Warteschlange öffnen sollte, und umgekehrt. Gemeinsam genutzte Warteschlangen und Gruppen mit gemeinsamer Warteschlange sind nur in IBM MQ for z/OS verfügbar.

Zur Angabe einer fernen Warteschlange muss ein Programm im vollständigen Warteschlangennamen den Namen des Warteschlangenmanagers angeben (es sei denn, es liegt eine lokale Definition der fernen Warteschlange vor).

Der von einer Anwendung angegebene Warteschlangenname kann der Name (oder der Aliasname) einer lokalen Warteschlange oder der Name einer lokalen Definition einer fernen Warteschlange sein. Die Anwendung selbst muss nicht wissen, um welche Art von Warteschlange es sich handelt, es sei denn, sie erwartet eine Nachricht aus dieser Warteschlange (in diesem Fall muss es sich um eine lokale Warteschlange handeln). Wenn die Anwendung das Warteschlangenobjekt öffnet, löst der Aufruf MQOPEN den Namen auf, wodurch festgestellt wird, an welcher Warteschlange die nachfolgenden Operationen durchgeführt werden sollen. Dieser Mechanismus ist insofern signifikant, als dass die Anwendung inhärent nicht von bestimmten Warteschlangen abhängig ist, die an bestimmten Orten im Netz der Warteschlangenmanager definiert sind. Falls ein Systemadministrator die Warteschlangen im Netz umstellt und deren Definitionen ändert, hat dies den Vorteil, dass die Anwendungen, die auf diese Warteschlangen zugreifen, nicht geändert werden müssen.

## Reservierte Objektnamen

Objektnamen, die mit SYSTEM. beginnen, sind für vom Warteschlangenmanager definierte Objekte reserviert. Mit den Befehlen **Alter**, **Define** und **Replace** können Sie diese Objektdefinitionen an Ihre Installation anpassen. Eine vollständige Liste der für IBM MQ definierten Namen finden Sie im Abschnitt Warteschlangennamen.

z/OS Unter IBM MQ for z/OS ist der Name der Coupling-Facility-Anwendungsstruktur CSQSYSAPPL reserviert.

**Zugehörige Konzepte**

Installationsname unter AIX, Linux, and Windows

## IBM MQ-Dateinamen

Die einzelnen Warteschlangenmanager, Warteschlangen, Prozessdefinitionen, Namenslisten, Kanäle, Clientverbindungskanäle, Empfangsprogramme, Services und Authentifizierungsdatenobjekte in IBM MQ werden jeweils durch eine Datei dargestellt. Da Objektnamen nicht notwendigerweise auch gültige Dateinamen sind, wandelt der Warteschlangenmanager die Objektnamen bei Bedarf in gültige Dateinamen um.

Der Standardpfad eines Warteschlangenmanager-Verzeichnisses setzt sich wie folgt zusammen:

- Ein Präfix, das in den IBM MQ-Konfigurationsdaten definiert wird:
  - Linux AIX Unter AIX and Linux ist `/var/mqm` das Standardpräfix. Es wird in der Zeilengruppe `DefaultPrefix` der Konfigurationsdatei 'mqs.ini' konfiguriert.
  - Windows Auf Windows-32-Bit-Systemen ist `C:\Programme (x86)\IBM\WebSphere MQ` das Standardpräfix. Auf Windows-64-Bit-Systemen ist `C:\Programme\IBM\MQ` das Standardpräfix. Sowohl bei 32-Bit- als auch bei 64-Bit-Installationen befinden sich die Datenverzeichnisse unter `C:\ProgramData\IBM\MQ`. Es wird in der Zeilengruppe `DefaultPrefix` der Konfigurationsdatei 'mqs.ini' konfiguriert.

  Sofern verfügbar, kann das Präfix über die IBM MQ-Eigenschaftenseite in IBM MQ Explorer geändert werden. Andernfalls müssen Sie die Konfigurationsdatei `mqs.ini` manuell bearbeiten.
- Der Name des Warteschlangenmanagers wird in einen gültigen Verzeichnisnamen umgesetzt. Der Warteschlangenmanager

  ```
  queue.manager
  ```

  beispielsweise wird dargestellt als

  ```
  queue!manager
  ```

  Dieser Vorgang wird als *Namensumsetzung* bezeichnet.

In IBM MQ können Sie Warteschlangenmanagern Namen mit bis zu 48 Zeichen zuweisen.

Beispielsweise kann ein Warteschlangenmanager den folgenden Namen erhalten:

```
QUEUE.MANAGER.ACCOUNTING.SERVICES
```

Jeder Warteschlangenmanager wird jedoch anhand einer Datei dargestellt; für Dateinamen bestehen sowohl Längenbeschränkungen als auch Vorgaben für die Zeichen, die verwendet werden können. Daher werden die Namen der Dateien für Objekte automatisch entsprechend den Anforderungen des Dateisystems umgewandelt.

Für die Umsetzung von Warteschlangenmanager-Namen gelten die folgenden Regeln:

1. Umsetzung einzelner Zeichen:
   - Von . !
   - Von / in &
2. Ist das Ergebnis immer noch kein gültiger Name:
   a. Begrenzen Sie die Länge des Namens auf acht Zeichen.
   b. Hängen Sie ein dreistelliges numerisches Suffix an.

Wird beispielsweise das Standardpräfix verwendet und lautet der Warteschlangenmanager-Name `queue.manager`, gilt Folgendes:

- Windows Unter Windows mit NTFS oder FAT32 wird der Warteschlangenmanagername wie folgt umgesetzt:

```
C:\Programme\IBM\MQ\qmgrs\queue!manager
```

- Windows Unter Windows mit FAT wird der Warteschlangenmanagername wie folgt umgesetzt:

```
C:\Programme\IBM\MQ\qmgrs\queue!ma
```

- Linux AIX Unter AIX and Linux wird der Warteschlangenmanagername wie folgt umgesetzt:

```
/var/mqm/qmgrs/queue!manager
```

Der Umsetzungsalgorithmus unterscheidet auch in Dateisystemen, bei denen die Groß-/Kleinschreibung keine Rolle spielt, zwischen Dateinamen, die sich nur in der Groß-/Kleinschreibung unterscheiden.

### Objektnamensumsetzung

Bei Objektnamen handelt es sich nicht notwendigerweise um gültige Dateisystemnamen. Daher müssen Sie Objektnamen unter Umständen umwandeln. Die Vorgehensweise unterscheidet sich von der Umsetzung von Warteschlangenmanager-Namen, da auf jeder Maschine nur wenige Warteschlangenmanager-Namen, jedoch unter Umständen eine große Anzahl von Objekten pro Warteschlangenmanager vorhanden sind. Warteschlangen, Prozessdefinitionen, Namenslisten, Kanäle, Clientverbindungskanäle, Empfangsprogramme, Services und Authentifizierungsdatenobjekte sind im Dateisystem dargestellt.

Die vom Umsetzungsprozess generierten neuen Namen sind keine direkte Entsprechung des ursprünglichen Objektnamens. Mit dem Befehl **dspmqfls** können die ursprünglichen Namen umgesetzter Objektnamen angezeigt werden.

**Zugehörige Verweise**

**dspmqfls** (Dateinamen anzeigen)

**Zugehörige Informationen**

Zeilengruppe 'AllQueueManagers' in der Datei 'mqs.ini'

## IBM i Objektnamen unter IBM i

Einem Warteschlangenmanager ist eine Warteschlangenmanagerbibliothek mit einem eindeutigen Namen zugeordnet. Warteschlangenmanagernamen und Objektnamen müssen gegebenenfalls umgewandelt werden, um den Anforderungen des IBM i Integrated File System (IFS) zu entsprechen.

Wenn ein Warteschlangenmanager erstellt wird, ordnet IBM MQ ihm eine Warteschlangenmanagerbibliothek zu. Diese Warteschlangenmanagerbibliothek erhält einen eindeutigen Namen aus maximal zehn Zeichen, der überwiegend auf dem benutzerdefinierten Namen des Warteschlangenmanagers basiert. Sowohl der Warteschlangenmanager als auch die Warteschlangenmanagerbibliothek werden in einem Verzeichnis abgelegt, das ebenfalls auf dem Namen des Warteschlangenmanagers basiert und das Präfix `/QIBM/UserData/mqm` besitzt. Es folgt ein Beispiel für einen Warteschlangenmanager, eine Warteschlangenmanagerbibliothek und ein Verzeichnis:

| | |
|---|---|
| Name des Warteschlangenmanagers | ORANGE |
| Name der Warteschlangenmanagerbibliothek | QMORANGE |
| Directory | `/QIBM/UserData/mqm/ORANGE` |

Alle Namen von Warteschlangenmanagern und Warteschlangenmanagerbibliotheken werden in Zeilengruppen in der Datei `/QIBM/UserData/mqm/mqs.ini` geschrieben.

### IBM MQ-IFS-Verzeichnisse und -Dateien

Das integrierte Dateisystem (IFS) von IBM i wird von IBM MQ ausgiebig zum Speichern von Daten verwendet. Weitere Informationen zum integrierten Dateisystem finden Sie in der *Einführung zum integrierten Dateisystem*.

Jedes IBM MQ-Objekt, z. B. ein Kanal oder Warteschlangenmanager, wird durch eine Datei dargestellt. Da Objektnamen nicht notwendigerweise auch gültige Dateinamen sind, wandelt der Warteschlangenmanager die Objektnamen bei Bedarf in gültige Dateinamen um.

Der Pfad zu einem Warteschlangenmanager-Verzeichnis wird aus folgenden Teilen gebildet:

- Einem Präfix, das in der Konfigurationsdatei des Warteschlangenmanagers `qm.ini` definiert ist. Das Standardpräfix lautet `/QIBM/UserData/mqm`.
- Einem Literal (`qmgrs`).
- Einem codierten Warteschlangenmanager-Namen, bei dem es sich um den Warteschlangenmanager-Namen handelt, der in einen gültigen Verzeichnisnamen umgesetzt wurde. Beispielsweise wird der Warteschlangenmanager `queue/manager` durch `queue&manager` dargestellt.

Dieser Prozess wird als Namensumsetzung bezeichnet.

### Warteschlangenmanager-Namensumsetzung im integrierten Dateisystem

In IBM MQ können Sie Warteschlangenmanagern Namen mit bis zu 48 Zeichen zuweisen.

Sie können einen Warteschlangenmanager beispielsweise `QUEUE/MANAGER/ACCOUNTING/SERVICES` nennen. Ebenso wie für jeden Warteschlangenmanager eine Bibliothek erstellt wird, wird jeder Warteschlangenmanager auch durch eine Datei dargestellt. Aufgrund unterschiedlicher Codepunkte in EBCDIC gibt es Einschränkungen hinsichtlich der Zeichen, die im Namen verwendet werden können. Dies hat zur Folge, dass Namen von Dateien des integrierten Dateisystems, die Objekte darstellen, automatisch umgesetzt werden, damit die Anforderungen des Dateisystems erfüllt werden.

Wenn Sie das Beispiel eines Warteschlangenmanagers mit dem Namen `queue/manager`verwenden, das Zeichen / in &umwandeln und das Standardpräfix annehmen, wird der Warteschlangenmanagername in IBM MQ for IBM i zu `/QIBM/UserData/mqm/qmgrs/queue&manager`.

### Objektnamensumsetzung

Objektnamen sind nicht zwangsläufig auch gültige Dateisystemnamen, sodass sie möglicherweise umgesetzt werden müssen. Die zu diesem Zweck verwendete Methode unterscheidet sich von der für Warteschlangenmanager-Namen, weil es trotz der Tatsache, dass es für jedes System nur wenige Warteschlangenmanager-Namen gibt, für jeden Warteschlangenmanager eine große Zahl von anderen Objekten geben kann. Es werden nur Prozessdefinitionen, Warteschlangen und Namenslisten im Dateisystem dargestellt; für Kanäle gelten diese Hinweise nicht.

Die vom Umsetzungsprozess generierten neuen Namen sind keine direkte Entsprechung des ursprünglichen Objektnamens. Mit dem Befehl DSPMQMOBJN können Sie die umgesetzten Namen für IBM MQ-Objekte anzeigen.

## Verteilte Warteschlangen und Cluster

Durch die verteilte Steuerung von Warteschlangen können Nachrichten von einem Warteschlangenmanager an einen anderen gesendet werden. Der empfangende Warteschlangenmanager kann sich auf demselben System oder auf einem anderen System ganz in der Nähe oder am anderen Ende der Welt befinden. Er kann auf derselben Plattform wie der lokale Warteschlangenmanager oder jeder anderen von IBM MQ unterstützten Plattform aktiv sein. In einer Umgebung mit verteilter Steuerung von Warteschlangen können Sie alle Verbindungen manuell definieren oder Sie können einen Cluster erstellen und IBM MQ den größten Teil der Verbindungskonfiguration überlassen.

## Verteilte Steuerung von Warteschlangen

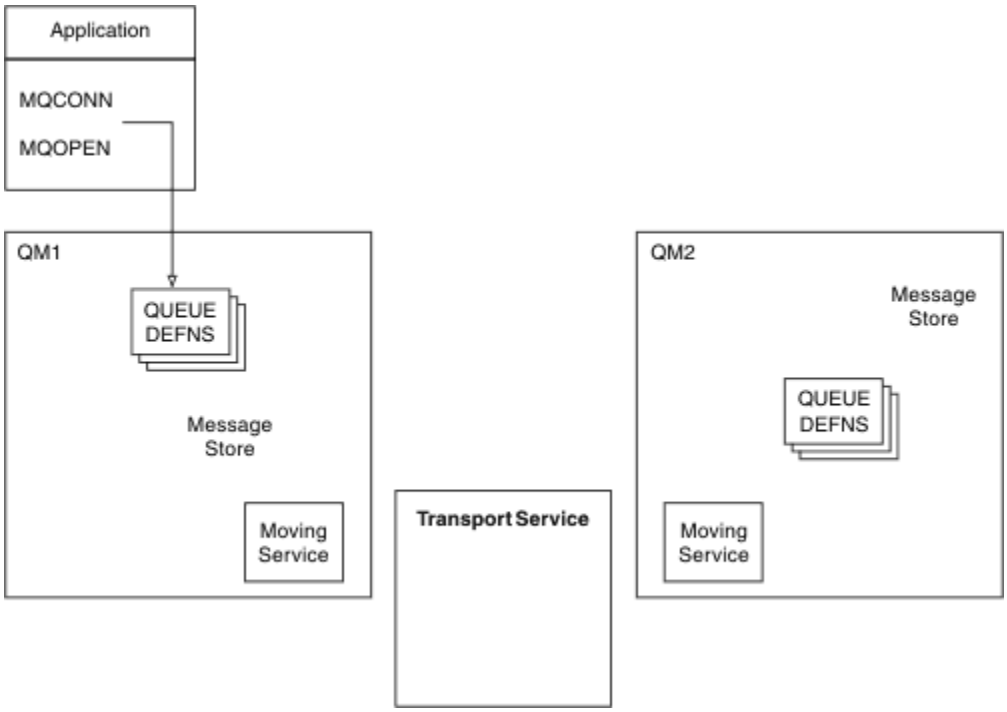



*Abbildung 4. Übersicht über die Komponenten der verteilten Steuerung von Warteschlangen*

In der vorherigen Abbildung:

- Eine Anwendung stellt mit dem Aufruf MQCONN eine Verbindung mit einem Warteschlangenmanager her. Danach öffnet die Anwendung mit dem Aufruf MQOPEN eine Warteschlange, sodass sie Nachrichten darin einreihen kann.
- Jeder Warteschlangenmanager besitzt für jede seiner Warteschlangen eine Definition. Dies können Definitionen *lokaler Warteschlangen* (von diesem Warteschlangenmanager bereitgestellt) und Definitionen *ferner Warteschlangen* (von anderen Warteschlangenmanagern bereitgestellt) sein.
- Wenn die Nachrichten für eine ferne Warteschlange bestimmt sind, behält der lokale Warteschlangenmanager sie in einer *Übertragungswarteschlange*, die die Nachrichten in einem Nachrichtenspeicher speichert, bis sie an den fernen Warteschlangenmanager weitergeleitet werden können.
- Jeder Warteschlangenmanager enthält eine Datenübertragungssoftware, den sogenannten *Übertragungsservice*, über den der Warteschlangenmanager mit anderen Warteschlangenmanagern kommuniziert.
- Der *Transportservice* ist vom Warteschlangenmanager unabhängig, sodass es sich um jeden der folgenden Services (je nach Plattform) handeln kann:
  - Systems Network Architecture Advanced Program-to Program Communication (SNA APPC)
  - Transmission Control Protocol/Internet Protocol (TCP/IP)
  - Network Basic Input/Output System (NetBIOS)
  - Sequenced Packet Exchange (SPX)

**Komponenten, die zum Senden einer Nachricht erforderlich sind**

Um eine Nachricht an einen fernen Warteschlangenmanager senden zu können, benötigt der lokale Warteschlangenmanager Definitionen für eine *Übertragungswarteschlange* und einen *Kanal*. Ein Kanal ist eine einseitige Übertragungsverbindung zwischen zwei Warteschlangenmanagern. In ihm können Nachrichten, die für beliebig viele Warteschlangen des fernen Warteschlangenmanagers bestimmt sind, übertragen werden.

Es gibt für jedes Ende eines Kanals eine separate Definition, in der es zum Beispiel als Sendeseite oder Empfangsseite definiert ist. Ein einfacher Kanal besteht aus einer *Sender*kanaldefinition beim lokalen

Warteschlangenmanager und einer *Empfänger*kanaldefinition beim fernen Warteschlangenmanager. Diese beiden Definitionen müssen den gleichen Namen haben und gemeinsam einen einzigen Kanal bilden.

Die Software, die das Senden und Empfangen von Nachrichten ausführt, wird als *Nachrichtenkanalagent* bezeichnet. An jedem Ende eines Kanals befindet sich ein *Nachrichtenkanalagent*.

Jeder Warteschlangenmanager muss eine *Warteschlange für nicht zustellbare Nachrichten* (auch *Warteschlange für nicht zugestellte Nachrichten* genannt) besitzen. In diese Warteschlange werden Nachrichten gestellt, die nicht an ihre Zieladresse übermittelt werden können.

Die folgende Abbildung zeigt die Beziehung zwischen Warteschlangenmanagern, Übertragungswarteschlangen, Kanäle und Nachrichtenkanalagenten:

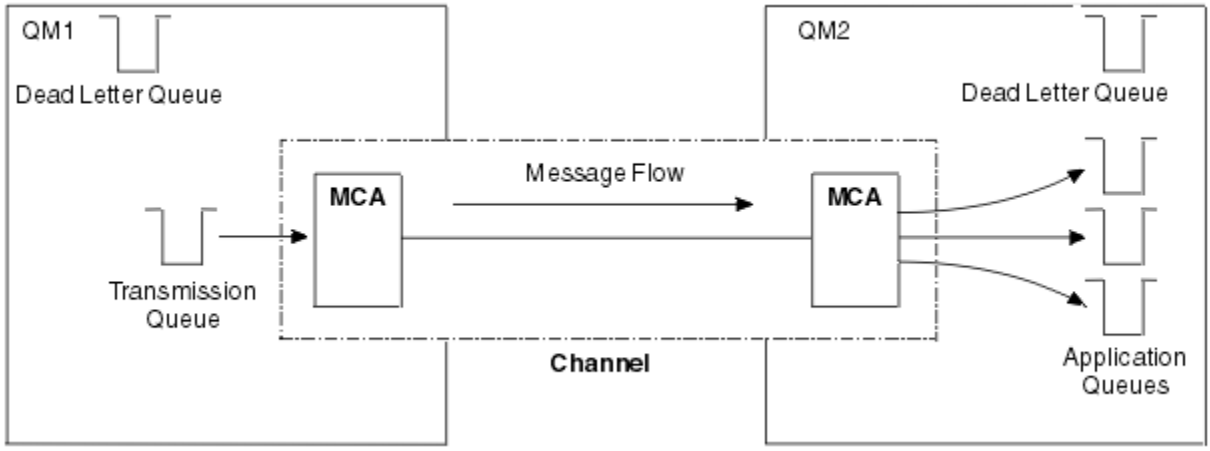



*Abbildung 5. Nachrichten senden*

**Komponenten, die zur Rückgabe einer Nachricht erforderlich sind**

Wenn Ihre Anwendung erfordert, dass Nachrichten vom fernen Warteschlangenmanager zurückgegeben werden, müssen Sie einen weiteren Kanal definieren, der, wie in der Abbildung gezeigt, in entgegengesetzter Richtung zwischen den Warteschlangenmanagern verläuft:

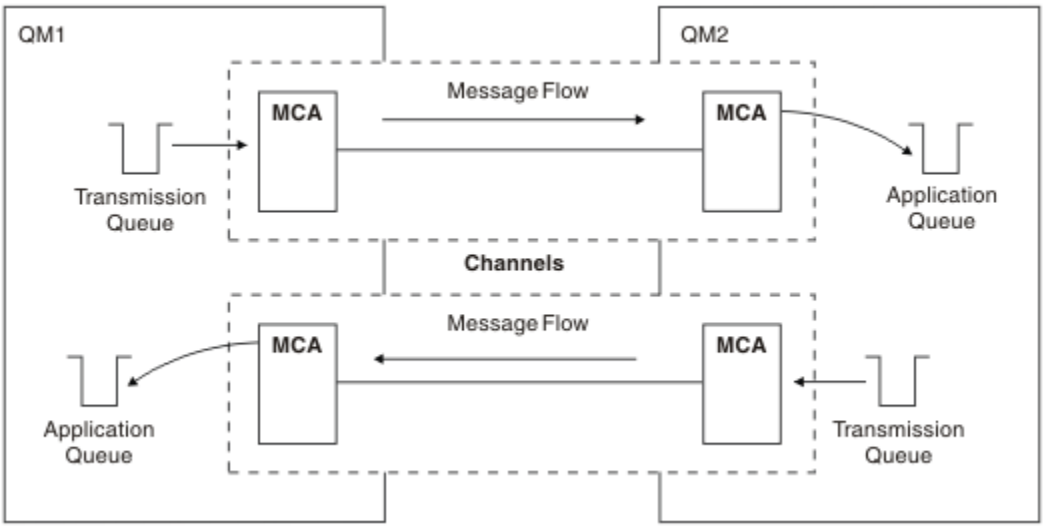



*Abbildung 6. Nachrichten in beiden Richtungen senden*

## Cluster

Anstatt die Verbindungen in einer Umgebung mit verteilter Steuerung von Warteschlangen manuell zu definieren, können Sie auch mehrere Warteschlangenmanager zu einem Cluster zusammenfassen. In diesem Fall können die Warteschlangenmanager die von ihnen bereitgestellten Warteschlangen auch

anderen Warteschlangenmanagern im Cluster zur Verfügung stellen, ohne dass für jedes Ziel explizite Kanaldefinitionen, Definitionen ferner Warteschlangen oder Übertragungswarteschlangen vorhanden sein müssen. Jeder Warteschlangenmanager in einem Cluster besitzt eine einzelne Übertragungswarteschlange, um Nachrichten an einen anderen Warteschlangenmanager im Cluster zu übertragen. Sie müssen dann für jeden Warteschlangenmanager nur einen Clusterempfängerkanal und einen Clustersenderkanal definieren; sämtliche weiteren Kanäle werden automatisch vom Cluster verwaltet.

Die Verbindung eines IBM MQ-Clients mit einem Warteschlangenmanager eines Clusters erfolgt auf die gleiche Weise wie bei jedem anderen Warteschlangenmanager. Wie bei der manuell konfigurierten verteilten Steuerung von Warteschlangen wird der Aufruf MQPUT verwendet, um eine Nachricht in eine Warteschlange eines anderen Warteschlangenmanagers zu stellen. Mit dem Aufruf MQGET können Nachrichten aus einer lokalen Warteschlange abgerufen werden.

Warteschlangenmanager auf Plattformen, die Cluster unterstützen, müssen jedoch kein Teil eines Clusters sein. Neben oder anstatt der Verwendung von Clustern können Sie nach wie vor manuell Techniken für die verteilte Steuerung von Warteschlangen konfigurieren und verwenden.

**Vorteile bei der Verwendung von Clustern**

Cluster bieten zwei wichtige Vorteile:

- Cluster vereinfachen die Verwaltung von IBM MQ-Netzen, für die normalerweise unzählige Objektdefinitionen für Kanäle, Übertragungswarteschlangen und ferne Warteschlangen konfiguriert werden müssen. Besonders gravierend ist dieser Aufwand in großen Netzen, die sich häufig ändern und in denen die Verbindung vieler Warteschlangenmanager erforderlich ist. Bei einer solchen Architektur ist die Konfiguration und Verwaltung ohne Cluster besonders anspruchsvoll.
- In Clustern kann die Arbeitslast der Nachrichtenübertragung über mehrere Warteschlangen und Warteschlangenmanager verteilt werden. Eine solche Verteilung sieht die Verteilung der Arbeitslast einer einzelnen Warteschlange auf identische Instanzen dieser Warteschlange auf mehreren Warteschlangenmanagern vor. Durch diese Verteilung erreichen Sie zum Einen eine größere Fehlersicherheit des Systems, zum Anderen verbessern Sie aber auch die Skalierleistung besonders aktiver Nachrichtenflüsse im System. In einer solchen Umgebung verfügt jede Instanz der verteilten Warteschlangen über konsumierende Anwendungen, die die Nachrichten verarbeiten. Weitere Informationen finden Sie im Abschnitt Cluster für Workload-Management verwenden.

**Weiterleitung von Nachrichten in einem Cluster**

Sie können sich einen Cluster wie ein Netz aus Warteschlangen vorstellen, das von einem gewissenhaften Systemadministrator verwaltet wird. Sobald Sie eine Clusterwarteschlange definieren, erstellt der Systemadministrator automatisch auf den anderen Warteschlangenmanagern die benötigten Definitionen für die ferne Warteschlange.

Definitionen für Übertragungswarteschlangen müssen nicht erstellt werden, da IBM MQ auf jedem Warteschlangenmanager des Clusters eine Übertragungswarteschlange bereitstellt. Über diese Übertragungswarteschlange können Nachrichten an jeden anderen Warteschlangenmanager des Clusters übertragen werden. Sie müssen sich nicht auf die Verwendung nur einer einzelnen Übertragungswarteschlange beschränken. Ein Warteschlangenmanager kann mehrere Übertragungswarteschlangen verwenden, um die Nachrichten, die an die einzelnen Warteschlangenmanager in einem Cluster gesendet werden, voneinander zu trennen. In der Regel verwendet ein Warteschlangenmanager eine einzelne Clusterübertragungswarteschlange. Durch die Änderung des Warteschlangenmanagerattributs DEFCLXQ können Sie festlegen, dass ein Warteschlangenmanager für jeden Warteschlangenmanager in einem Cluster eine andere Clusterübertragungswarteschlange verwenden soll. Cluster-Übertragungs-WS können auch manuell definiert werden.

Alle Warteschlangenmanager, die einem Cluster beitreten, sind sich über diese Art der Zusammenarbeit einig. Sie geben Informationen über sich selbst und die von ihnen bereitgestellten Warteschlangen nach außen und erhalten ihrerseits Informationen über die anderen Mitglieder des Clusters.

Um sicherzustellen, dass keine Informationen verloren gehen, wenn ein Warteschlangenmanager ausfällt, sollten Sie zwei Warteschlangenmanager im Cluster mit *vollständigem Repository* definieren. Diese Warteschlangenmanager speichern einen vollständigen Satz aller Informationen über die Warteschlangenmanager und Warteschlangen im Cluster. Alle anderen Warteschlangenmanager im Cluster speichern nur

Informationen über diejenigen Warteschlangenmanager und Warteschlangen, mit denen sie Nachrichten austauschen. Diese Warteschlangenmanager werden als Warteschlangenmanager mit *Teilrepositorys* bezeichnet. Weitere Informationen finden Sie in „Cluster-Repository" auf Seite 61.

Für den Anschluss an einen Cluster benötigt ein Warteschlangenmanager zwei Kanäle, einen Clustersenderkanal und einen Clusterempfängerkanal:

- Ein Clustersenderkanal ist ein einem Senderkanal entsprechender Kommunikationskanal. Den Clustersenderkanal müssen Sie auf jedem Warteschlangenmanager manuell erstellen, um den Warteschlangenmanager mit einem vollständigen Repository des Clusters zu verbinden.
- Ein Clusterempfängerkanal ist ein einem Empfängerkanal entsprechender Kommunikationskanal. Sie müssen manuell einen Clusterempfängerkanal erstellen. Dieser Kanal wird als Mechanismus für den Empfang der Clusterkommunikation durch den Warteschlangenmanager genutzt.

Alle anderen Kanäle, die zusätzlich für die Kommunikation zwischen diesem Warteschlangenmanager und den anderen Mitgliedern des Clusters erforderlich sind, werden dann automatisch erstellt.

Die folgende Abbildung zeigt die Komponenten eines Clusters mit dem Namen CLUSTER:

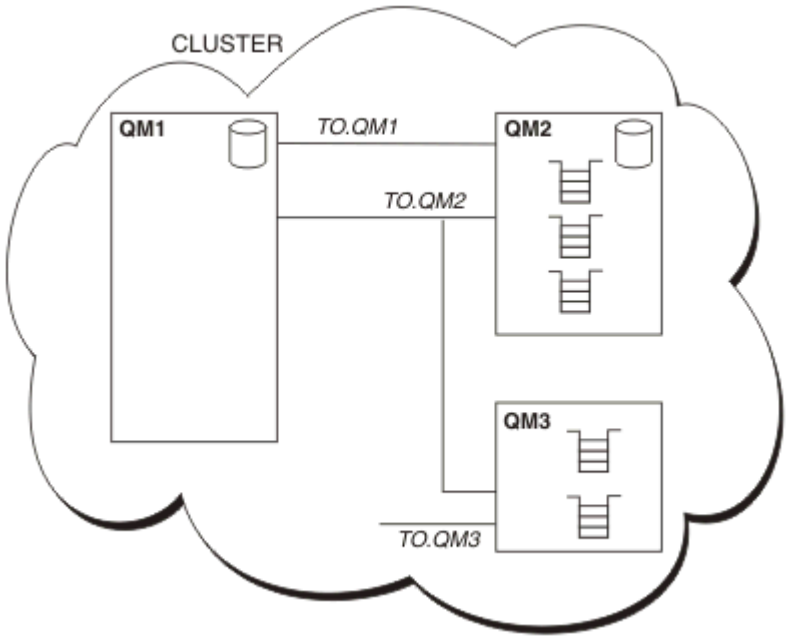



*Abbildung 7. Warteschlangenmanagercluster*

- CLUSTER enthält die drei Warteschlangenmanager QM1, QM2 und QM3.
- QM1 und QM2 verwalten vollständige Repositorys mit Informationen über die Warteschlangenmanager und Warteschlangen im Cluster.
- QM2 und QM3 verwalten einige Clusterwarteschlangen, d. h. Warteschlangen, auf die jeder andere Warteschlangenmanager im Cluster zugreifen kann.
- Jeder Warteschlangenmanager besitzt einen Clusterempfängerkanal mit dem Namen TO.qmgr, über den er Nachrichten empfangen kann.
- Außerdem besitzt jeder Warteschlangenmanager einen Clustersenderkanal, über den er Informationen an einen der Repository-Warteschlangenmanager senden kann.
- QM1 und QM3 senden Informationen an das Repository von QM2 und QM2 sendet Informationen an das Repository von QM1.

## Komponenten der verteilten Steuerung von Warteschlangen

Die verteilte Steuerung von Warteschlangen besteht aus folgenden Komponenten: Nachrichtenkanälen, Nachrichtenkanalagenten, Übertragungswarteschlangen, Kanalinitiatoren, Empfangsprogrammen und Ka-

nalexitprogrammen. Bei der Definition der beiden Enden eines Nachrichtenkanals kann es sich um verschiedene Typen handeln.

Nachrichtenkanäle sind die Kanäle, über die Nachrichten von einem Warteschlangenmanager an einen anderen übertragen werden. Verwechseln Sie Nachrichtenkanäle nicht mit MQI-Kanälen. Es gibt zwei Typen von MQI-Kanälen, Serververbindung (SVRCONN) und Clientverbindung (CLNTCONN). Weitere Informationen finden Sie im Abschnitt Kanäle.

Bei der Definition der beiden Enden eines Nachrichtenkanals kann es sich um einen der folgenden Typen handeln:

- Sender (SDR)
- Empfänger (RCVR)
- Server (SVR)
- Requester (RQSTR)
- Clustersender (CLUSSDR)
- Clusterempfänger (CLUSRCVR)

Ein Nachrichtenkanal wird definiert, indem einer dieser Typen an einem Ende und ein kompatibler Typ am anderen Ende definiert werden. Folgende Kombinationen sind möglich:

- Sender-Empfänger
- Requester-Server
- Requester-Sender (Callback)
- Server-Empfänger
- Clustersender-Clusterempfänger

Detaillierte Anweisungen zum Erstellen eines Sender-Empfänger-Kanals finden Sie im Abschnitt Kanäle definieren. Beispiele für die erforderlichen Parameter zur Konfiguration von Sender-Empfänger-Kanälen finden Sie im betreffenden Abschnitt Beispielkonfigurationsinformationen für Ihre Plattform. Informationen zu den Parametern, die zur Definition eines Kanals eines beliebigen Typs benötigt werden, finden Sie in der Beschreibung von DEFINE CHANNEL.

## Sender-Empfänger-Kanäle

Ein Sender in einem System startet den Kanal, damit er Nachrichten an das andere System senden kann. Der Sender fordert den Empfänger am anderen Ende des Kanals zum Starten auf. Der Sender sendet Nachrichten aus seiner Übertragungswarteschlange an den Empfänger. Der Empfänger stellt die Nachrichten in die Zielwarteschlange. Abbildung 8 auf Seite 51 zeigt diesen Vorgang.

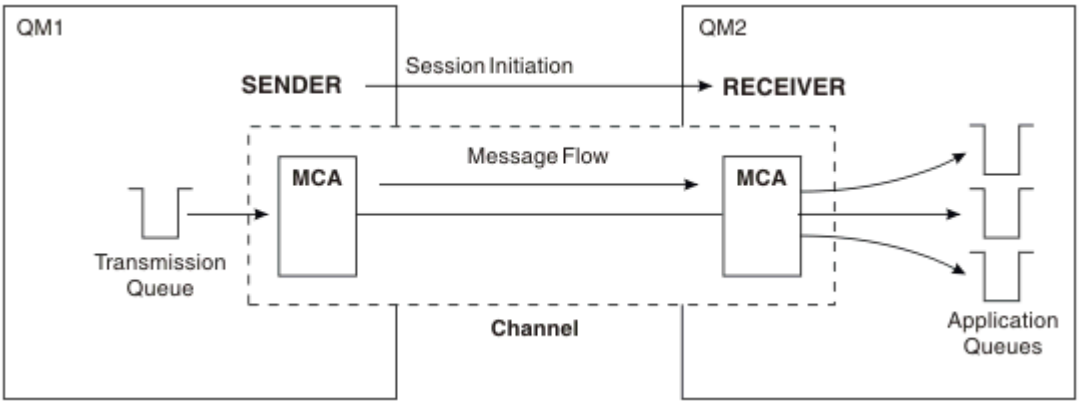



*Abbildung 8. Sender-Empfänger-Kanal*

## Anforderer-Server-Kanäle

Ein Requester in einem System startet den Kanal, damit er Nachrichten vom anderen System empfangen kann. Der Requester fordert den Server am anderen Ende des Kanals zum Starten auf. Der Server sendet Nachrichten aus der Übertragungswarteschlange, die in seiner Kanaldefinition angegeben ist, an den Requester.

Ein Serverkanal kann auch selbst die Kommunikation einleiten und Nachrichten an einen Requester senden. Dies gilt nur für *vollständig qualifizierte* Server, d. h. für Serverkanäle, in deren Kanaldefinition der Verbindungsname des Partners angegeben ist. Ein vollständig qualifizierter Server kann entweder von einem Requester gestartet werden oder selbst eine Kommunikation mit einem Requester einleiten.

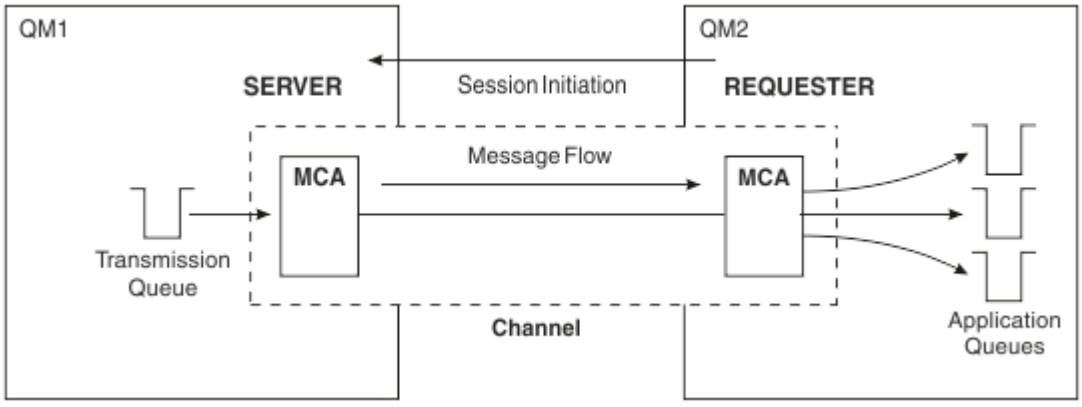



*Abbildung 9. Requester-Server-Kanal*

## Anforderer-Sender-Kanäle

Der Requester startet den Kanal und der Sender beendet den Aufruf. Der Sender startet die Kommunikation anschließend erneut anhand der Informationen in seiner Kanaldefinition (bekannt als *Callback*). Er sendet Nachrichten aus der Übertragungswarteschlange an den Requester.

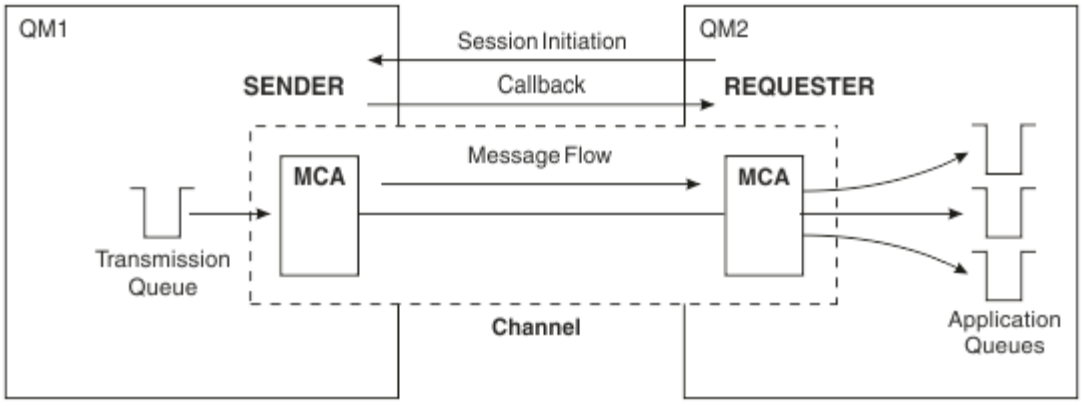



*Abbildung 10. Requester-Sender-Kanal*

## Server-Empfänger-Kanäle

Dies entspricht einem Sender-Empfänger-Kanal, gilt aber nur für *vollständig qualifizierte* Server, d. h. für Serverkanäle, in deren Kanaldefinition der Verbindungsname des Partners angegeben ist. Der Start des Kanals muss auf der Serverseite der Verbindung eingeleitet werden. Dieser Vorgang wird in Abbildung 8 auf Seite 51 dargestellt.

## Clustersenderkanäle

In einem Cluster besitzt jeder Warteschlangenmanager einen Clustersenderkanal, über den er Clusterinformationen an eines der vollständigen Warteschlangenmanager-Repositorys senden kann. Warteschlangenmanager können über Clustersenderkanäle auch Nachrichten an andere Warteschlangenmanager senden.

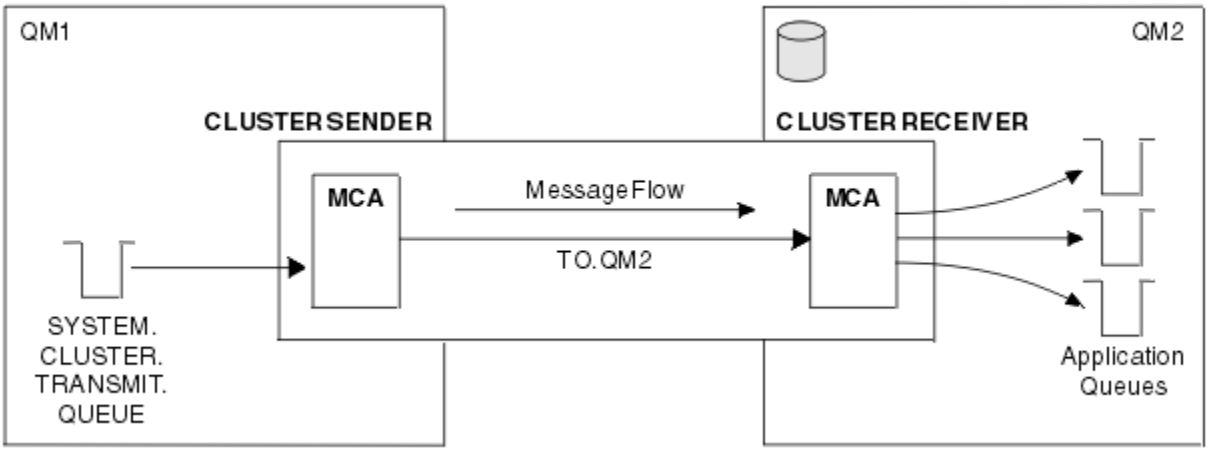



*Abbildung 11. Ein Clustersenderkanal*

## Clusterempfängerkanäle

In einem Cluster besitzt jeder Warteschlangenmanager einen Clusterempfängerkanal, über den er Nachrichten sowie Informationen über den Cluster empfangen kann. Dieser Vorgang wird in Abbildung 11 auf Seite 53 dargestellt.

# Warteschlangen für nicht zustellbare Nachrichten

Die Warteschlange für nicht zustellbare Nachrichten (oder Warteschlange für nicht zugestellte Nachrichten) ist die Warteschlange, an die Nachrichten gesendet werden, wenn sie nicht an ihr eigentliches Ziel weitergeleitet werden können. In der Regel hat jeder Warteschlangenmanager eine solche Warteschlange.

Eine *Warteschlange für nicht zustellbare Nachrichten* (DLQ), manchmal auch als *Warteschlange für nicht zugestellte Nachrichten*bezeichnet, ist eine Haltewarteschlange für Nachrichten, die nicht an ihre Zielwarteschlangen zugestellt werden können, z. B. weil die Warteschlange nicht vorhanden oder voll ist. Warteschlangen für nicht zustellbare Nachrichten werden auch auf der Sendeseite eines Kanals bei Datenkonvertierungsfehlern verwendet. Jeder Warteschlangenmanager im Netz hat üblicherweise eine lokale Warteschlange, die als Warteschlange für nicht zustellbare Nachrichten verwendet wird, sodass Nachrichten, die ihrem Ziel momentan nicht zugestellt werden können, für den späteren Abruf gespeichert werden können.

Nachrichten können von Warteschlangenmanagern, Nachrichtenkanalagenten (MCA; Message Channel Agent) und Anwendungen in die Warteschlange für nicht zustellbare Nachrichten eingereiht werden. Alle Nachrichten in der DLQ müssen mit einer *Headerstruktur einer nicht zustellbaren Nachricht* (MQDLH) als Präfix vorangesetzt werden. Das Feld *Reason* der MQDLH-Struktur enthält einen Ursachencode, der angibt, warum sich die Nachricht in der DLQ befindet.

Sie sollten jedem Warteschlangenmanager eine Warteschlange für nicht zustellbare Nachrichten zuweisen. Wenn keine vorhanden ist und der Nachrichtenkanalagent eine Nachricht nicht einreihen kann, bleibt sie in der Übertragungswarteschlange stehen und der Kanal wird gestoppt. Auch werden schnelle, nicht persistente Nachrichten (siehe Schnelle, nicht persistente Nachrichten), die nicht zugestellt werden können, verworfen, wenn auf dem Zielsystem keine Warteschlange für nicht zustellbare Nachrichten vorhanden ist.

Allerdings kann sich die Verwendung von Warteschlangen für nicht zustellbare Nachrichten auf die Reihenfolge auswirken, in der Nachrichten zugestellt werden, sodass Sie möglicherweise darauf verzichten möchten.

**Zugehörige Tasks**

Mit dead-letter-Warteschlangen arbeiten

Fehlerbehebung bei nicht zugestellten Nachrichten

**Zugehörige Verweise**

runmqdlq (Steuerroutine der Warteschlange für nicht zustellbare Nachrichten ausführen)

## Definitionen ferner Warteschlangen

Definitionen ferner Warteschlangen sind Definitionen für Warteschlangen, die Eigentum eines anderen Warteschlangenmanagers sind.

Abrufen können Anwendungen Nachrichten zwar nur aus lokalen Warteschlangen, aber Einreihen können Sie Nachrichten sowohl in lokale als auch in ferne Warteschlangen. Deshalb kann ein Warteschlangenmanager neben einer Definition für jede seiner lokalen Warteschlangen auch *Definitionen ferner Warteschlangen* besitzen. Der Vorteil von Definitionen ferner Warteschlangen besteht darin, dass sie es einer Anwendung ermöglichen, eine Nachricht in eine ferne Warteschlange zu stellen, ohne den Namen der fernen Warteschlange oder des fernen Warteschlangenmanagers bzw. den Namen der Übertragungswarteschlange kennen zu müssen. Definitionen ferner Warteschlangen bieten Standortunabhängigkeit.

Es gibt weitere Verwendungsmöglichkeiten für Definitionen ferner Warteschlangen, die später beschrieben werden.

## Ferne Warteschlangenmanager abrufen

Sie verfügen möglicherweise nicht immer über einen separaten Kanal zwischen jedem Quellen- und Ziel-Warteschlangenmanager. Es gibt verschiedene Möglichkeiten, eine Verbindung zwischen den beiden herzustellen, einschließlich Multihopping, gemeinsame Nutzung von Kanälen, Nutzung verschiedener Kanäle und Clustering.

### Multihopping

Wenn keine direkte Kommunikationsverbindung zwischen dem Quellen-Warteschlangenmanager und dem Ziel-Warteschlangenmanager besteht, ist es möglich, einen oder mehrere *zwischengeschaltete Warteschlangenmanager* auf dem Weg zum Ziel-Warteschlangenmanager zu durchlaufen. Dies ist als *Multihopping* bekannt.

Sie müssen Kanäle zwischen allen Warteschlangenmanagern und Übertragungswarteschlangen auf den zwischengeschalteten Warteschlangenmanagern definieren. Dies wird in Abbildung 12 auf Seite 55 gezeigt.

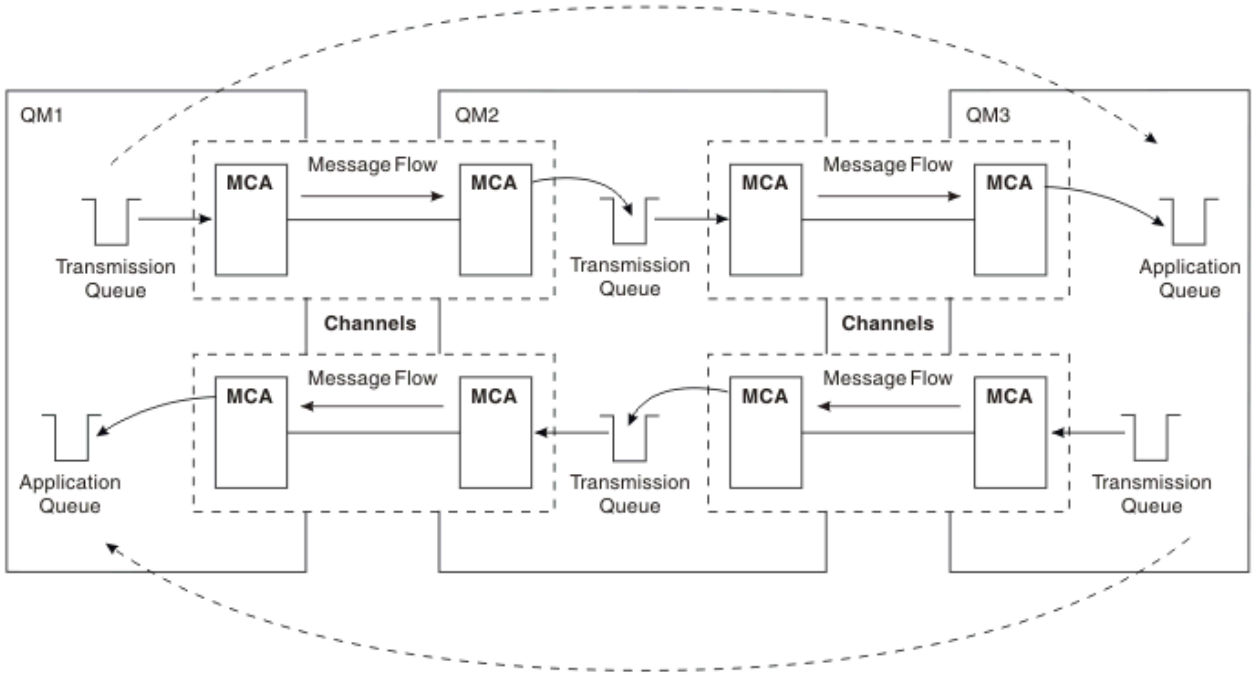



*Abbildung 12. Übertragung über zwischengeschaltete Warteschlangenmanager*

## Gemeinsame Nutzung von Kanälen

Als Anwendungsentwickler können Sie wählen, ob Sie Ihre Anwendungen zwingen, neben dem Namen der Warteschlange auch den Namen des fernen Warteschlangenmanagers anzugeben, oder ob Sie für jede ferne Warteschlange eine *Definition einer fernen Warteschlange* erstellen. Diese Definition enthält den Namen des fernen Warteschlangenmanagers, den Namen der Warteschlange und den Namen der Übertragungswarteschlange. In jedem Fall werden alle Nachrichten von allen Anwendungen, die Warteschlangen an denselben fernen Standort adressieren, über dieselbe Übertragungswarteschlange gesendet. Dies wird in Abbildung 13 auf Seite 55 gezeigt.

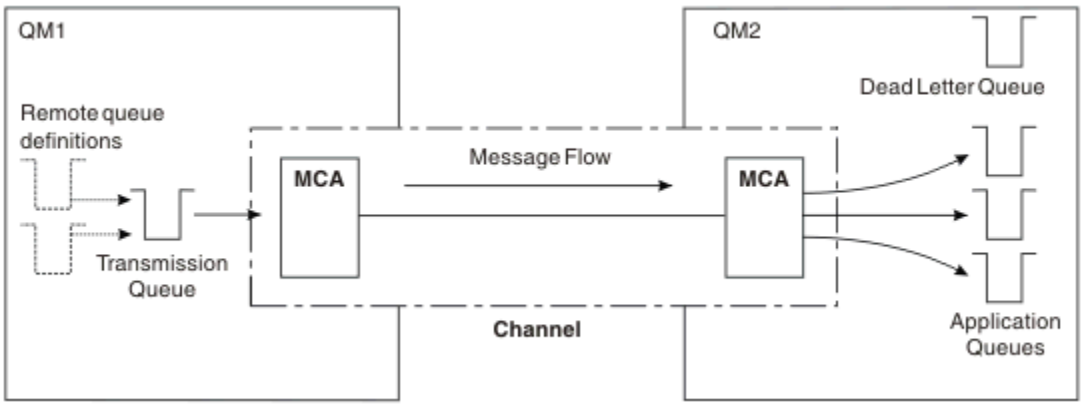



*Abbildung 13. Gemeinsame Nutzung einer Übertragungswarteschlange*

Abbildung 13 auf Seite 55 zeigt, dass für Nachrichten von mehreren Anwendungen an mehrere ferne Warteschlangen derselbe Kanal verwendet werden kann.

## Nutzung verschiedener Kanäle

Wenn Nachrichten unterschiedlichen Typs zwischen zwei Warteschlangenmanagern ausgetauscht werden müssen, können Sie mehrere Kanäle zwischen den beiden definieren. Es kann Situationen geben, in

denen Sie alternative Kanäle benötigen, vielleicht aus Sicherheitsgründen oder um Zustellungsgeschwindigkeit durch reine Massenübertragung von Nachrichten auszugleichen.

Um einen zweiten Kanal zu konfigurieren, müssen Sie einen weiteren Kanal und eine weitere Übertragungswarteschlange definieren sowie eine Definition einer fernen Warteschlange erstellen, in der Sie den Standort und den Namen der Übertragungswarteschlange angeben. Ihre Anwendung kann anschließend jeden der beiden Kanäle verwenden, aber die Nachrichten werden weiterhin an dieselbe Zielwarteschlange zugestellt. Dies wird in Abbildung 14 auf Seite 56 gezeigt.

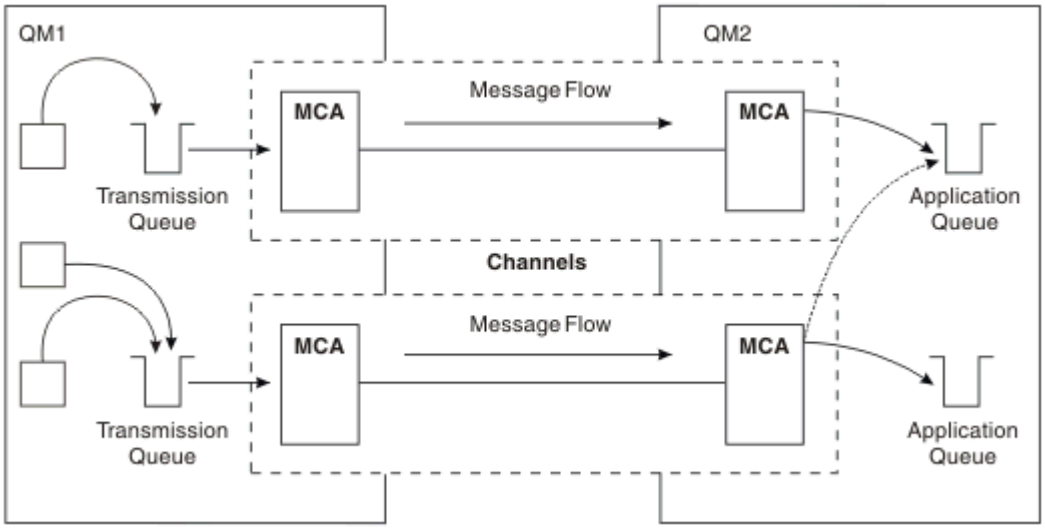



*Abbildung 14. Nutzung mehrerer Kanäle*

Wenn Sie eine Übertragungswarteschlange in Definitionen ferner Warteschlangen angeben, darf der Standort (d. h. der Ziel-Warteschlangenmanager) **nicht** in den Anwendungen selbst angegeben werden. Denn wenn er dort angegeben ist, verwendet der Warteschlangenmanager nicht die Definitionen ferner Warteschlangen. Definitionen ferner Warteschlangen bieten Standortunabhängigkeit. Anwendungen können Nachrichten in eine *logische* Warteschlange stellen, ohne zu wissen, wo sich die Warteschlange befindet, und Sie können die *physische* Warteschlange wechseln, ohne Ihre Anwendung ändern zu müssen.

### Clustering verwenden

Jeder Warteschlangenmanager in einem Cluster definiert einen Clusterempfängerkanal. Wenn ein Warteschlangenmanager eine Nachricht an einen anderen Warteschlangenmanager senden möchte, definiert er automatisch den zugehörigen Clustersenderkanal. Wenn es beispielsweise mehrere Instanzen einer Warteschlange in einem Cluster gibt, kann der Clustersenderkanal für jeden der Warteschlangenmanager, die die Warteschlange betreiben, definiert werden. IBM MQ verwendet einen Workload-Management-Algorithmus mit einer Umlaufroutine, um einen verfügbaren Warteschlangenmanager zur Weiterleitung einer Nachricht auszuwählen. Weitere Informationen finden Sie im Abschnitt Cluster.

## Adressinformation

Wenn eine Anwendung Nachrichten einreiht, die für einen fernen Warteschlangenmanager bestimmt sind, fügt der lokale Warteschlangenmanager einen Übertragungsheader hinzu, bevor er die Nachrichten in die Übertragungswarteschlange stellt. Dieser Header enthält die Namen der Zielwarteschlange und des Warteschlangenmanagers, also die *Adressinformationen*.

In einer Umgebung mit nur einem Warteschlangenmanager wird die Adresse einer Zielwarteschlange erstellt, wenn eine Anwendung eine Warteschlange zum Einreihen von Nachrichten öffnet. Da sich die Zielwarteschlange auf demselben Warteschlangenmanager befindet, werden keine Adressinformationen benötigt.

In einer Umgebung mit verteilter Steuerung von Warteschlangen muss der Warteschlangenmanager nicht nur den Namen der Zielwarteschlange kennen, sondern auch deren Standort (d. h. den Namen des Warteschlangenmanagers) und die Route zu diesem Standort (d. h. die Übertragungswarteschlange).

Diese Adressinformationen sind im Übertragungsheader enthalten. Der empfangende Kanal entfernt den Übertragungsheader und lokalisiert anhand der darin enthaltenen Informationen die Zielwarteschlange.

Sie können vermeiden, dass in Anwendungen der Name des Zielwarteschlangenmanagers angegeben werden muss, indem Sie eine Definition einer fernen Warteschlange verwenden. In dieser Definition werden der Name der fernen Warteschlange, der Name des fernen Warteschlangenmanagers, für den Nachrichten bestimmt sind, und der Name der Übertragungswarteschlange für den Transport der Nachrichten angegeben.

## Was sind Aliasnamen?

Aliasnamen dienen dazu, eine Servicequalität für Nachrichten bereitzustellen. Mithilfe des Warteschlangenmanager-Aliasnamens kann ein Systemadministrator den Namen eines Ziel-Warteschlangenmanagers ändern, ohne dass Sie daraufhin Ihre Anwendungen ändern müssen. Außerdem erhält der Systemadministrator die Möglichkeit, die Route zu einem Zielwarteschlangenmanager zu ändern oder eine Route einzurichten, die durch mehrere andere Warteschlangenmanager führt (Multihopping). Der Aliasname für Empfangswarteschlangen für Antworten bietet eine Servicequalität für Antworten.

Warteschlangenmanager-Aliasnamen und Aliasnamen für Empfangswarteschlangen für Antworten werden mithilfe einer Definition einer fernen Warteschlange, die ein leeres RNAME-Feld enthält, erstellt. Solche Definitionen definieren keine realen Warteschlangen; sie werden vom Warteschlangenmanager verwendet, um Namen von physischen Warteschlangen, Warteschlangenmanager-Namen und Namen von Übertragungswarteschlangen aufzulösen.

Aliasnamensdefinitionen sind durch ein leeres RNAME-Feld gekennzeichnet.

### Auflösung des Warteschlangennamens

Eine Auflösung von Warteschlangennamen findet auf jedem Warteschlangenmanager bei jedem Öffnen einer Warteschlange statt. Sie hat den Zweck, die Zielwarteschlange, den Ziel-Warteschlangenmanager (kann ein lokaler sein) und die Route zu diesem Warteschlangenmanager (kann null sein) zu ermitteln. Der aufgelöste Name besteht aus drei Teilen: den Namen des Warteschlangenmanagers, der Warteschlange und, falls es sich um einen fernen Warteschlangenmanager handelt, der Übertragungswarteschlange.

Wenn eine Definition einer fernen Warteschlange vorhanden ist, wird nicht auf Aliasnamensdefinitionen verwiesen. Der in der Anwendung angegebene Warteschlangenname wird in die Namen der Zielwarteschlange, des fernen Warteschlangenmanagers und der Übertragungswarteschlange, die in der Definition einer fernen Warteschlange angegeben sind, aufgelöst. Ausführlichere Informationen zur Auflösung von Warteschlangennamen finden Sie im Abschnitt Auflösung von Warteschlangennamen.

Wenn keine Definition einer fernen Warteschlange vorhanden und ein Warteschlangenmanager-Name angegeben ist (oder vom Namensservice aufgelöst wurde), überprüft der Warteschlangenmanager, ob für den übergebenen Warteschlangenmanager-Namen eine zugehörige Warteschlangenmanager-Aliasnamensdefinition vorhanden ist. Wenn ja, wird der Warteschlangenmanager-Name anhand der darin enthaltenen Informationen in den Namen des Zielwarteschlangenmanagers aufgelöst. Mithilfe der Warteschlangenmanager-Aliasnamensdefinition kann auch die Übertragungswarteschlange für den Zielwarteschlangenmanager ermittelt werden.

Wenn der aufgelöste Warteschlangenname nicht der Name einer lokalen Warteschlange ist, werden sowohl der Warteschlangenmanager-Name als auch der Warteschlangenname in den Übertragungsheader jeder Nachricht aufgenommen, die von der Anwendung in die Übertragungswarteschlange gestellt wird.

Der Name der verwendeten Übertragungswarteschlange entspricht normalerweise dem aufgelösten Warteschlangenmanager-Namen, sofern er nicht durch eine Definition einer fernen Warteschlange oder eine Warteschlangenmanager-Aliasnamensdefinition geändert wird. Wenn Sie keine Übertragungswarteschlange, aber eine Standardübertragungswarteschlange definiert haben, wird diese verwendet.

z/OS Namen von Warteschlangenmanagern, die unter z/OS ausgeführt werden, sind auf eine Länge von vier Zeichen begrenzt.

## WS-Manager-Aliasdefinitionen

Warteschlangenmanager-Aliasnamensdefinitionen werden wirksam, wenn eine Anwendung, die eine Warteschlange zum Einreihen einer Nachricht öffnet, den Warteschlangennamen **und** den Warteschlangenmanager-Namen angibt.

Warteschlangenmanager-Aliasnamensdefinitionen werden in drei Situationen verwendet:

- Beim Senden von Nachrichten, um den Warteschlangenmanager-Namen neu zuzuordnen
- Beim Senden von Nachrichten, um die Übertragungswarteschlange zu ändern oder anzugeben
- Beim Empfangen von Nachrichten, um zu ermitteln, ob der lokale Warteschlangenmanager das gewünschte Ziel für die Nachrichten ist

### Abgehende Nachrichten - Neuzuordnung des Warteschlangenmanager-Namens

Mithilfe von Warteschlangenmanager-Aliasnamensdefinitionen kann der in einem MQOPEN-Aufruf angegebene Warteschlangenmanager-Name neu zugeordnet werden. Ein Beispiel: In einem MQOPEN-Aufruf wird der Warteschlangenname THISQ und der Warteschlangenmanager-Name YOURQM angegeben. Der lokale Warteschlangenmanager verfügt über folgende Warteschlangenmanager-Aliasnamensdefinition:

```
DEFINE QREMOTE (YOURQM) RQMNAME(REALQM)
```

Daraus geht hervor, dass tatsächlich der Warteschlangenmanager REALQM verwendet werden soll, wenn eine Anwendung Nachrichten an den Warteschlangenmanager YOURQM übergibt. Handelt es sich bei REALQM um den lokalen Warteschlangenmanager, stellt dieser die Nachrichten in die Warteschlange THISQ, eine lokale Warteschlange. Trägt der lokale Warteschlangenmanager nicht die Bezeichnung REALQM, wird die Nachricht an ein Übertragungswarteschlange namens REALQM weitergeleitet. Der Warteschlangenmanager ändert den Übertragungsheader von YOURQM in REALQM.

### Abgehende Nachrichten - Änderung oder Angabe der Übertragungswarteschlange

Abbildung 15 auf Seite 58 zeigt ein Szenario, in dem bei Warteschlangenmanager QM1 Nachrichten ankommen, in deren Übertragungsheadern Namen von Warteschlangen auf Warteschlangenmanager QM3 angegeben sind. In diesem Szenario ist QM3 durch Multihopping über QM2 erreichbar.

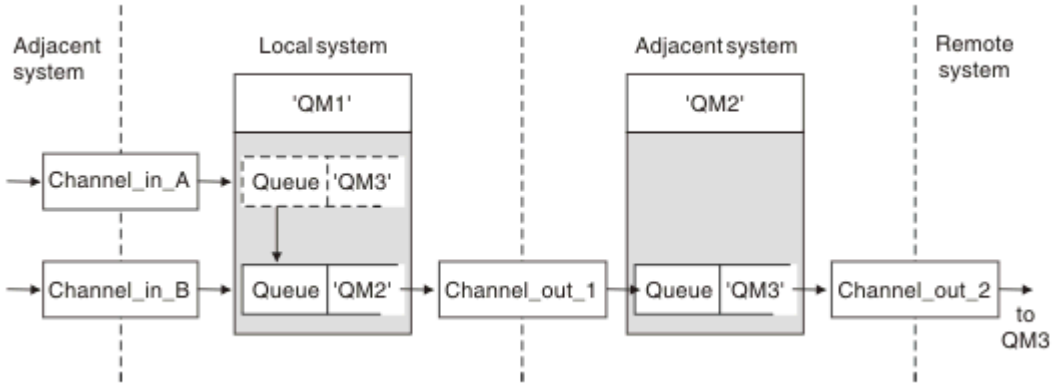



*Abbildung 15. Aliasname des Warteschlangenmanagers*

Alle Nachrichten für QM3 werden auf QM1 mit einem Warteschlangenmanager-Aliasnamen erfasst. Der Warteschlangenmanager-Aliasname lautet QM3 und enthält die Definition QM3 über die Übertragungswarteschlange QM2. Das folgende Beispiel zeigt die entsprechende Definition:

```
DEFINE QREMOTE (QM3) RNAME(' ') RQMNAME(QM3) XMITQ(QM2)
```

Der Warteschlangenmanager stellt die Nachrichten in Übertragungswarteschlange QM2, ändert jedoch den Header der Übertragungswarteschlange nicht, da sich der Name des Zielwarteschlangenmanagers QM3 nicht ändert.

Alle an QM1 ankommenden Nachrichten, in deren Übertragungsheader der Name einer Warteschlange von QM2 angegeben ist, werden auch in die Übertragungswarteschlange QM2 eingereiht. Auf diese Weise werden Nachrichten mit unterschiedlichen Zielen in einer gemeinsamen Übertragungswarteschlange auf einem geeigneten Nachbarsystem gesammelt, um sie dann an ihre jeweiligen Ziele zu übertragen.

### Eingehende Nachrichten - Ermittlung des Ziels

Ein empfangender Nachrichtenkanalagent öffnet die Warteschlange, die im Übertragungsheader angegeben ist. Wenn eine Warteschlangenmanager-Aliasnamensdefinition mit demselben Namen wie dem des angegebenen Warteschlangenmanagers vorhanden ist, wird der im Übertragungsheader empfangene Name des Warteschlangenmanagers aus der Definition ersetzt.

Für diesen Prozess gibt es zwei Anwendungsfälle:

- Weiterleitung von Nachrichten an einen anderen Warteschlangenmanager
- Änderung des Warteschlangenmanager-Namens in den Namen des lokalen Warteschlangenmanagers

## Aliasnamendefinitionen für Antwortwarteschlange

In einer Aliasnamensdefinition für eine Empfangswarteschlange für Antworten werden alternative Namen für die Antwortinformationen im Nachrichtendeskriptor angegeben. Dies bietet den Vorteil, dass Sie den Namen einer Warteschlange oder eines Warteschlangenmanagers ändern können, ohne Ihre Anwendungen ändern zu müssen.

### Auflösung des Warteschlangennamens

Wenn eine Anwendung auf eine Nachricht antwortet, ermittelt sie anhand der Daten im *Nachrichtendeskriptor* der empfangenen Nachricht die Warteschlange, an die die Antwort gehen soll. Die sendende Anwendung gibt an, wohin Antworten gesendet werden sollen, indem sie entsprechende Informationen an ihre Nachrichten anhängt. Dieses Konzept muss im Rahmen des Anwendungsentwurfs koordiniert werden.

Die Auflösung des Warteschlangennamens finden auf der Sendeseite Ihrer Anwendung statt, bevor die Nachricht in eine Warteschlangen gestellt wird. Die Auflösung des Warteschlangennamens finden daher vor der Interaktion mit der fernen Anwendung statt, an die die Nachricht gesendet wird. Dies ist die einzige Situation, in der die Namensauflösung zu einem Zeitpunkt erfolgt, an dem eine Warteschlange nicht geöffnet ist.

### Auflösung des Warteschlangennamens mithilfe eines Warteschlangenmanager-Aliasnamens

Normalerweise gibt eine Anwendung eine Empfangswarteschlange für Antworten an und lässt das Feld für den Namen des Managers der Empfangswarteschlange für Antworten leer. Der Warteschlangenmanager trägt seinen Namen zum Zeitpunkt des Einreihens der Nachricht ein. Dieses Verfahren funktioniert gut, es sei denn, für Antworten soll ein anderer Kanal verwendet werden, also beispielsweise ein Kanal, der die Übertragungswarteschlange QM1_relief verwendet, und nicht der Standardrückgabekanal, der die Übertragungswarteschlange QM1 verwendet. In dieser Situation handelt es sich bei den Warteschlangenmanager-Namen, die in den Übertragungswarteschlangenheadern angegeben sind, nicht um “echte” Warteschlangenmanager-Namen, sondern um Namen, die mithilfe von Warteschlangenmanager-Aliasnamensdefinitionen neu festgelegt werden. Um Antworten über alternative Routen zurückgeben zu können, müssen auch Daten der Empfangswarteschlange für Antworten mithilfe von Aliasnamensdefinitionen für Empfangswarteschlangen für Antworten zugeordnet werden.

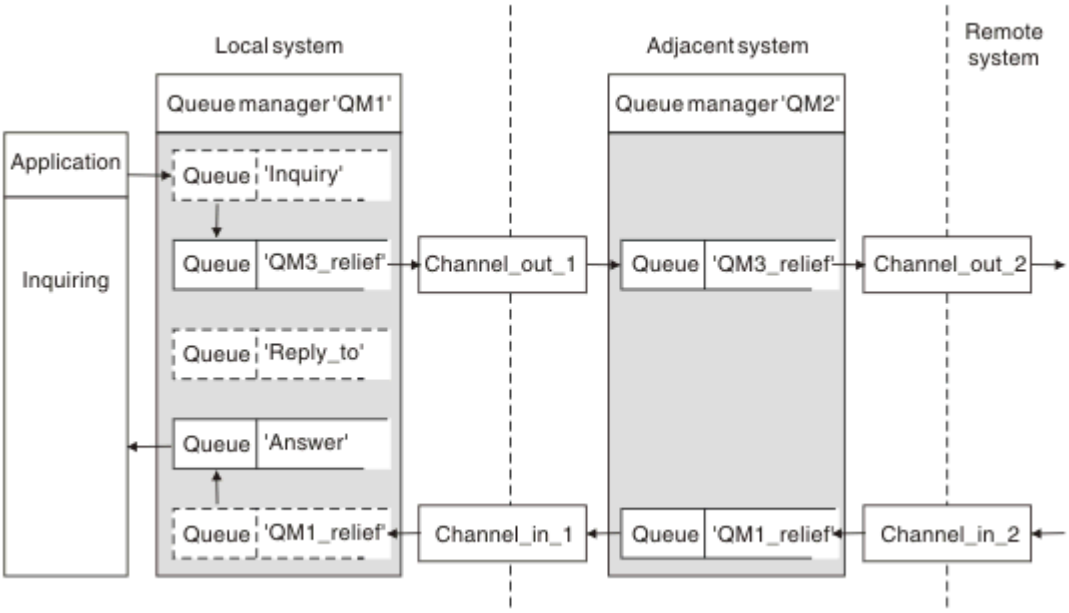



*Abbildung 16. Aliasnamen für Empfangswarteschlange für Antworten zum Ändern der Antwortposition verwenden*

Das Beispiel in Abbildung 16 auf Seite 60 zeigt Folgendes:

1. Die Anwendung reiht mit dem Aufruf MQPUT eine Nachricht ein und gibt folgende Informationen im Nachrichtendeskriptor an:

   ```
   ReplyToQ='Reply_to'
   ReplyToQMgr=' '
   ```

   Die Angabe für ReplyToQMgr muss ein Leerzeichen sein, damit der Aliasname der Empfangswarteschlange für Antworten verwendet wird.
2. Sie erstellen eine Aliasdefinition `Reply_to` für die Empfangswarteschlange für Antworten, in welcher der Name `Answer` und der Name des Warteschlangenmanagers `QM1_relief` enthalten sind.

   ```
   DEFINE QREMOTE ('Reply_to') RNAME ('Answer')
   RQMNAME ('QM1_relief')
   ```
3. Die Nachrichten werden mit einem Nachrichtendeskriptor gesendet, in dem `ReplyToQ='Answer'` und `ReplyToQMgr='QM1_relief'` angegeben ist.
4. Die Anwendungsspezifikation muss die Information enthalten, dass Antworten in der Warteschlange `Answer` und nicht in `Reply_to` zu finden sind.

Um Vorbereitungen für den Empfang der Antworten zu treffen, müssen Sie den parallelen Rückgabekanal erstellen, indem Sie Folgendes definieren:

- Auf QM2 die Übertragungswarteschlange `QM1_relief`

  ```
  DEFINE QLOCAL ('QM1_relief') USAGE(XMITQ)
  ```
- Auf QM1 den Warteschlangenmanager-Aliasnamen `QM1_relief`

  ```
  DEFINE QREMOTE ('QM1_relief') RNAME() RQMNAME(QM1)
  ```

Dieser Warteschlangenmanager-Aliasname beendet die Kette aus parallelen Rückgabekanälen und erfasst die Nachrichten für QM1.

Wenn Sie glauben, dass Sie dieses Verfahren irgendwann in der Zukunft einsetzen werden, stellen Sie sicher, dass der Aliasname von Anfang an in Anwendungen verwendet wird. Im Moment handelt es sich um einen normalen Aliasnamen für die Empfangswarteschlange für Antworten, der aber später in einen Warteschlangenmanager-Aliasnamen geändert werden kann.

### Name der Empfangswarteschlange für Antworten

Bei der Benennung von Empfangswarteschlangen für Antworten ist sorgfältig vorzugehen. Dass eine Anwendung den Namen einer Empfangswarteschlange für Antworten in der Nachricht angibt, hat den Grund, dass sie auf diese Weise die Warteschlange angeben kann, an die die Antworten gesendet werden sollen. Wenn Sie eine Aliasnamensdefinition für eine Empfangswarteschlange für Antworten mit diesem Namen erstellen, darf die eigentliche Empfangswarteschlange für Antworten (d. h. die Definition einer lokalen Warteschlange) nicht denselben Namen haben. Deshalb muss die Aliasnamensdefinition der Empfangswarteschlange für Antworten einen neuen Warteschlangennamen und den Warteschlangenmanager-Namen enthalten und die Anwendungsspezifikation muss die Information enthalten, dass die Antworten in dieser anderen Warteschlange eingehen werden.

Die Anwendungen müssen die Nachrichten jetzt aus einer anderen Warteschlange abrufen, als sie beim Einreihen der ursprünglichen Nachricht als Empfangswarteschlange für Antworten angegeben haben.

## Clusterkomponenten

Cluster bestehen aus Warteschlangenmanagern, Clusterrepositorys, Clusterkanälen und Clusterwarteschlangen.

In den folgenden Abschnitten finden Sie Informationen zu den einzelnen Clusterkomponenten:

**Zugehörige Konzepte**

Vergleich von Clustering und verteilter Steuerung von Warteschlangen

**Zugehörige Tasks**

WS-Manager-Cluster konfigurieren

Neuen Cluster einrichten

### Cluster-Repository

Ein Repository ist eine Zusammenstellung von Informationen über die Warteschlangenmanager, die zu einem Cluster gehören.

Zu den Repositoryinformationen gehören die Namen der Warteschlangenmanager, ihre Standorte, ihre Kanäle, welche Warteschlangen von ihnen betrieben werden und weitere Informationen. Die Informationen werden in Form von Nachrichten in einer Warteschlange mit dem Namen `SYSTEM.CLUSTER.REPOSITORY.QUEUE` gespeichert. Diese Warteschlange ist eines der Standardobjekte. Multi Bei Multiplatforms wird sie definiert, wenn Sie einen IBM MQ-Warteschlangenmanager erstellen. z/OS Unter IBM MQ for z/OS wird sie als Teil der Warteschlangenmanageranpassung definiert.

### Vollständiges Repository und Teilrepository

Üblicherweise verfügen zwei Warteschlangenmanager in einem Cluster über ein vollständiges Repository. Die übrigen Warteschlangenmanager verfügen alle über ein Teilrepository.

Ein Warteschlangenmanager, der über einen vollständigen Satz von Informationen über jeden Warteschlangenmanager im Cluster verfügt, besitzt ein vollständiges Repository. Andere Warteschlangenmanager im Cluster besitzen Teilrepositorys mit einer Untermenge der Informationen in den vollständigen Repositorys.

Ein Teilrepository enthält nur Informationen über die Warteschlangenmanager, mit denen der Warteschlangenmanager Nachrichten austauschen muss. Die Warteschlangenmanager fordern Aktualisierun-

gen zu den von Ihnen benötigten Informationen an, d. h., wenn sich die Informationen ändern, sendet ihnen der Warteschlangenmanager mit dem vollständigen Repository die neuen Informationen. Die meiste Zeit enthält ein Teilrepository alle Informationen, die ein Warteschlangenmanager zur Ausführung innerhalb des Clusters benötigt. Falls zusätzliche Informationen benötigt werden, ruft er diese aus dem vollständigen Repository ab und nimmt eine entsprechende Aktualisierung des Teilrepositorys vor. Für die Anforderung und den Empfang von Aktualisierungen der Repositorys verwenden die Warteschlangenmanager die Warteschlange `SYSTEM.CLUSTER.COMMAND.QUEUE`.

Migrieren Sie bei der Migration von Warteschlangenmanagern, die zu einem Cluster gehören, die vollständigen Repositorys vor den Teilrepositorys. Der Grund dafür ist, dass ein älteres Repository neuere Attribute, die in einem neueren Release eingeführt wurden, nicht speichern kann. Es toleriert sie, speichert sie aber nicht.

## Clusterwarteschlangenmanager

Ein Clusterwarteschlangenmanager ist ein Warteschlangenmanager, der Mitglied eines Clusters ist.

Ein Warteschlangenmanager kann mehreren Clustern angehören. Jeder Clusterwarteschlangenmanager muss einen Namen haben, der für alle Cluster, denen er angehört, eindeutig ist.

Ein Clusterwarteschlangenmanager kann Warteschlangen betreiben, die er den anderen Warteschlangenmanagern im Cluster zugänglich macht. Dies muss er jedoch nicht tun. Stattdessen kann er Nachrichten in Warteschlangen stellen, die anderswo im Cluster gehostet werden, und nur Antworten empfangen, die explizit an ihn gerichtet sind.

z/OS In IBM MQ for z/OS kann ein Clusterwarteschlangenmanager Mitglied einer Gruppe mit gemeinsamer Warteschlange sein. In diesem Fall teilt er seine Warteschlangendefinitionen mit anderen Warteschlangenmanagern in derselben Gruppe mit gemeinsamer Warteschlange.

Clusterwarteschlangenmanager sind autonom. Sie besitzen die vollständige Kontrolle über die von ihnen definierten Warteschlangen und Kanäle. Ihre Definitionen können nicht von anderen Warteschlangenmanagern geändert werden (im Gegensatz zu Warteschlangenmanagern in derselben Gruppe mit gemeinsamer Warteschlange). Repository-Warteschlangenmanager kontrollieren nicht die Definitionen der anderen Warteschlangenmanager im Cluster. Sie verfügen über einen vollständigen Satz von Definitionen, der bei Bedarf verwendet werden kann. Ein Cluster ist eine Verknüpfung von Warteschlangenmanagern.

Nachdem Sie auf einem Clusterwarteschlangenmanager eine Definition erstellt oder geändert haben, werden die Informationen an den Warteschlangenmanager mit dem vollständigen Repository gesendet. Andere Repositorys im Cluster werden später aktualisiert.

## Warteschlangenmanager mit vollständigem Repository

Ein Warteschlangenmanager mit vollständigem Repository ist ein Clusterwarteschlangenmanager, der über eine vollständige Darstellung der Ressourcen des Clusters verfügt. Um Verfügbarkeit sicherzustellen, sollten in jedem Cluster mindestens zwei Warteschlangenmanager mit vollständigem Repository eingerichtet werden. Warteschlangenmanager mit vollständigem Repository empfangen Informationen, die von den anderen Warteschlangenmanagern im Cluster gesendet werden, und aktualisieren ihre Repositorys. Sie tauschen Nachrichten untereinander aus, um sicherzustellen, dass beide immer auf dem neuesten Stand der Informationen über den Cluster sind.

## Warteschlangenmanager und Repositorys

Jeder Cluster hat mindestens einen (besser zwei) Warteschlangenmanager mit vollständigen Repositorys, die alle Informationen über die Warteschlangenmanager, Warteschlangen und Kanäle in einem Cluster enthalten. Diese Repositorys enthalten auch Anforderungen von den anderen Warteschlangenmanagern im Cluster zur Aktualisierung der Informationen.

Jeder der anderen Warteschlangenmanager verfügt über ein Teilrepository, das Informationen über die Untergruppe von Warteschlangen und Warteschlangenmanagern enthält, mit denen er kommunizieren muss. Die Warteschlangenmanager erstellen ihre Teilrepositorys, indem sie Anfragen stellen, wenn sie zum ersten Mal auf eine andere Warteschlange oder einen anderen Warteschlangenmanager zugreifen

müssen. Dabei stellen sie die Anforderung, dass sie alle neuen Informationen erhalten, die die jeweilige Warteschlange oder den jeweiligen Warteschlangenmanager betreffen.

Jeder Warteschlangenmanager speichert seine Repositoryinformationen in Form von Nachrichten in der Warteschlange SYSTEM.CLUSTER.REPOSITORY.QUEUE. Die Warteschlangenmanager tauschen Repositoryinformationen in Form von Nachrichten in der Warteschlange SYSTEM.CLUSTER.COMMAND.QUEUE aus.

Jeder Warteschlangenmanager, der einem Cluster beitritt, definiert einen Clustersenderkanal (CLUSSDR) zu einem der Repositorys. Er erfährt sofort, welche anderen Warteschlangenmanager im Cluster über vollständige Repositorys verfügen. Von dem Moment an kann der Warteschlangenmanager Informationen aus den Repositorys anfordern. Wenn der Warteschlangenmanager Informationen an das ausgewählte Repository sendet, sendet er gleichzeitig Informationen an ein anderes Repository (falls vorhanden).

Ein vollständiges Repository wird aktualisiert, wenn der Warteschlangenmanager mit dem Repository Informationen von einem der Warteschlangenmanager empfängt, die mit ihm verbunden sind. Die neuen Informationen werden auch an ein anderes Repository gesendet, um das Risiko einer Verzögerung zu verringern, falls der Repository-Warteschlangenmanager außer Betrieb ist. Da alle Informationen doppelt gesendet werden, müssen Duplikate in den Repositorys gelöscht werden. Jedes Informationselement besitzt eine Folgenummer, die von den Repositorys zur Erkennung von Duplikaten genutzt wird. Alle Repositorys tauschen Nachrichten untereinander aus, um immer auf demselben Stand zu sein.

## Clusterwarteschlangen

Eine Clusterwarteschlange wird von einem Clusterwarteschlangenmanager anderen Warteschlangenmanagern im Cluster zur Verfügung gestellt.

Eine Clusterwarteschlangendefinition wird den anderen Warteschlangenmanagern im Cluster zugänglich gemacht. Die anderen Warteschlangenmanager im Cluster können ohne entsprechende Definition einer fernen Warteschlange Nachrichten in eine Clusterwarteschlange einreihen. Über eine Clusternamensliste kann eine Clusterwarteschlange in mehreren Clustern zugänglich gemacht werden.

Wenn eine Warteschlange zugänglich gemacht wird, können alle Warteschlangenmanager im Cluster Nachrichten in diese Warteschlange einreihen. Um eine Nachricht einzureihen, muss der Warteschlangenmanager anhand der vollständigen Repositorys ermitteln, wo sich die Warteschlange befindet. Anschließend fügt der Warteschlangenmanager der Nachricht einige Routing-Informationen hinzu und stellt sie dann in eine Clusterübertragungswarteschlange.

z/OS Bei einer Clusterwarteschlange kann es sich um eine Warteschlange handeln, die von Mitgliedern einer Gruppe mit gemeinsamer Warteschlange in IBM MQ for z/OS gemeinsam genutzt wird.

**Zugehörige Tasks**

Clusterwarteschlangen definieren

## Vergleich zwischen gemeinsam genutzten Warteschlangen und Clusterwarteschlangen

Diese Informationen sollen Ihnen helfen, gemeinsam genutzte Warteschlangen und Clusterwarteschlangen miteinander zu vergleichen und dann zu entscheiden, welche für Ihr System besser geeignet sind.

### Aufwand für Kanalinitiatoren

In Clusterwarteschlangen werden Nachrichten über Kanäle gesendet, sodass zum Anwendungsaufwand noch der Aufwand für Kanalinitiatoren hinzukommt. Es entsteht ein zusätzlicher Aufwand im Netz, weil Kanäle Nachrichten abrufen und einreihen. Dieser Aufwand entsteht bei gemeinsam genutzten Warteschlangen nicht, die deshalb bei der Verlagerung von Nachrichten zwischen Warteschlangenmanagern in einer Gruppe mit gemeinsamer Warteschlange weniger Verarbeitungskapazität in Anspruch nehmen.

### Verfügbarkeit von Nachrichten

Wird eine Nachricht in eine Warteschlange eingereiht, senden Clusterwarteschlangen die Nachricht an einen der Warteschlangenmanager mit aktiven Kanälen, die mit Ihrem Warteschlangenmanager verbunden sind. Wenn auf dem fernen Warteschlangenmanager Anwendungen, die zur Verarbeitung der Nachrichten verwendet werden, nicht aktiv sind, werden die Nachrichten nicht verarbeitet, sondern warten darauf, dass die Anwendung gestartet wird. Wurde ein Warteschlangenmanager heruntergefahren, gilt entsprechend, dass Nachrichten auf dem Warteschlangenmanager erst zur Verfügung gestellt werden, nachdem der Warteschlangenmanager erneut gestartet wurde. Diese Instanzen zeigen eine schlechtere Nachrichtenverfügbarkeit als gemeinsam genutzte Warteschlangen.

Bei der Verwendung gemeinsam genutzter Warteschlangen kann jede Anwendung in der Gruppe mit gemeinsamer Warteschlange Nachrichten, die gesendet werden, abrufen. Wird einer der Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange heruntergefahren, sind die Nachrichten für die anderen Warteschlangenmanager verfügbar, sodass eine höhere Nachrichtenverfügbarkeit als bei der Verwendung von Clusterwarteschlangen erreicht wird.

### Kapazität

Eine Coupling-Facility ist kostenintensiver als ein Datenträger. Deshalb sind die Kosten für die Speicherung von 1.000.000 Nachrichten in einer lokalen Warteschlange niedriger als für eine Coupling-Facility, die über ausreichend Kapazität zur Speicherung derselben Anzahl Nachrichten verfügt.

### Nachrichten an andere Warteschlangenmanager senden

Nachrichten in einer gemeinsam genutzten Warteschlange sind nur innerhalb einer Gruppe mit gemeinsamer Warteschlange verfügbar. Wenn Sie einen Warteschlangenmanager außerhalb der Gruppe mit gemeinsamer Warteschlange verwenden möchten, müssen Sie Kanäle nutzen. Für einen Lastausgleich zwischen mehreren fernen verteilten Warteschlangenmanagern können Sie Clustering einsetzen.

### Lastausgleich

Mithilfe des Clusterings können Sie eine Gewichtung festlegen, nach der gesendete Nachrichten auf Kanäle und Warteschlangenmanager verteilt werden sollen. Sie können beispielsweise festlegen, dass 60% der Nachrichten an einen bestimmten Warteschlangenmanager gesendet werden sollen und 40% der Nachrichten an einen anderen. Diese Instanz ist, was die Verarbeitung betrifft, nicht von der Verfügbarkeit des fernen Warteschlangenmanagers abhängig. Selbst wenn das System mit dem ersten Warteschlangenmanager überlastet und das System mit dem zweiten Warteschlangenmanager unausgelastet sein sollten, geht der größere Anteil der Nachrichten weiterhin an den ersten Warteschlangenmanager.

Bei gemeinsam genutzten Warteschlangen können zwei CICS-Systeme Nachrichten abrufen. Ist ein System überlastet, kann das andere System einen Großteil der Arbeitslast übernehmen.

## Clusterkanäle

In jedem vollständigen Repository definieren Sie einen Clusterempfängerkanal sowie eine Gruppe von Clustersenderkanälen für die Verbindung mit allen anderen vollständigen Repositorys im Cluster manuell. Wenn Sie ein Teilrepository hinzufügen, definieren Sie einen Clusterempfängerkanal und einen einzelnen Clustersenderkanal manuell, der sich mit einem der vollständigen Repositorys verbindet. Weitere Clustersenderkanäle werden bei Bedarf automatisch vom Cluster definiert. Automatisch definierte Clustersenderkanäle erhalten ihre Attribute von der jeweiligen Definition für den entsprechenden Clusterempfängerkanal im Empfangswarteschlangenmanager.

### Clusterempfängerkanal: CLUSRCVR

Eine CLUSRCVR-Kanaldefinition definiert das Ende eines Kanals, an dem ein Clusterwarteschlangenmanager Nachrichten von anderen Warteschlangenmanagern im Cluster empfangen kann.

Sie müssen für jeden Cluster-Warteschlangenmanager mindestens einen CLUSRCVR-Kanal definieren. Durch die Definition des CLUSRCVR-Kanals zeigt der Warteschlangenmanager den anderen Cluster-Warteschlangenmanagern an, dass er für Nachrichten empfangsbereit ist.

Eine CLUSRCVR-Kanaldefinition ermöglicht anderen Warteschlangenmanagern außerdem die automatische Erstellung entsprechender Clustersenderkanaldefinitionen. Weitere Informationen hierzu finden Sie im Abschnitt „Automatisch definierte Clustersenderkanäle” auf Seite 65 dieses Artikels.

## Clustersenderkanal: CLUSSDR

Sie definieren manuell einen CLUSSDR -Kanal von jedem vollständigen Repository-WS-Manager zu jedem anderen Repository-WS-Manager im Cluster, der vollständig in Repository enthalten ist. Alle Aktualisierungen, die von den vollständigen Repositorys ausgetauscht werden, fließen exklusiv über diese Kanäle. Durch die manuelle Definition dieser Kanäle steuern Sie explizit das Netz aus vollständigen Repositorys.

Wenn Sie einem Cluster ein Teilrepository hinzufügen, definieren Sie einen einzelnen CLUSSDR-Kanal manuell, der sich mit einem der vollständigen Repositorys verbindet. Welches vollständige Repository Sie wählen, macht kaum einen Unterschied, da nach der Herstellung des ersten Kontakts bei Bedarf weitere Cluster-Warteschlangenmanager-Objekte (einschließlich der CLUSSDR-Kanäle) automatisch für Ihren Warteschlangenmanager definiert werden. Dadurch kann Ihr Warteschlangenmanager Clusterinformationen an jedes beliebige vollständige Repository und Nachrichten an jeden beliebigen Warteschlangenmanager im Cluster senden.

Wie im Abschnitt dieses Artikels erläutert, basieren automatisch definierte Senderkanäle auf der Konfiguration des Clusterempfängerkanals. Daher sollten alle Kanaleigenschaften, die Sie für Clusterkanäle festlegen, auf den entsprechenden CLUSSDR- und Clusterempfängerkanälen identisch sein oder nur für Clusterempfängerkanäle festgelegt werden.

Sie sollten CLUSSDR-Kanäle nur aus den zuvor beschriebenen Gründen manuell definieren. Definieren Sie diese also nur, um eine Erstverbindung zwischen einem Teilrepository und einem vollständigen Repository herzustellen, oder um zwei vollständige Repositorys miteinander zu verbinden. Eine manuelle Konfiguration eines CLUSSDR-Kanals, der sich mit einem Teilrepository oder einem Warteschlangenmanager verbindet, der nicht zum Cluster gehört, führt zur Ausgabe von Fehlernachrichten, z. B. AMQ9427 und AMQ9428. In bestimmten Situationen ist dies jedoch als temporäre Lösung unvermeidlich (z. B. bei der Änderung des Standorts eines vollständigen Repositorys). Allerdings sollte die manuelle Definition so bald wie möglich gelöscht werden.

## Automatisch definierte Clustersenderkanäle

Wenn Sie einem Cluster ein Teilrepository hinzufügen, definieren Sie normalerweise nur zwei Clusterkanäle auf dem Warteschlangenmanager manuell:

- Einen Clustersenderkanal (CLUSSDR) zu einem vollständigen Warteschlangenmanager-Repository für den Cluster.
- Einen Clusterempfängerkanal (CLUSRCVR).

Der von Ihnen definierte CLUSSDR-Kanal ermöglicht es dem Warteschlangenmanager, den ersten Kontakt mit dem Cluster herzustellen. Nachdem der erste Kontakt hergestellt wurde, werden weitere CLUSSDR-Kanäle bei Bedarf automatisch vom Cluster definiert.

Ein automatisch definierter CLUSSDR-Kanal erhält seine Attribute von der entsprechenden CLUSRCVR-Kanaldefinition auf dem empfangenden Warteschlangenmanager. Selbst wenn ein manuell definierter CLUSSDR-Kanal vorhanden ist, werden die Attribute aus dem automatisch definierten CLUSSDR-Kanal verwendet. Angenommen, Sie definieren beispielsweise einen CLUSRCVR-Kanal ohne Angabe einer Portnummer im Parameter **CONNAME** und definieren manuell einen CLUSSDR-Kanal, der eine Portnummer angibt. Wenn der automatisch definierte CLUSSDR-Kanal den manuell definierten Kanal ersetzt, ist die Portnummer (die dem Kanal CLUSRCVR entnommen wird) nicht mehr belegt. Es wird die Standardportnummer verwendet und der Kanal fällt aus.

Falls Konfigurationsunterschiede zwischen einem manuell definierten CLUSSDR-Kanal und der entsprechenden CLUSRCVR-Kanaldefinition bestehen, werden manche Unterschiede sofort wirksam (zum Beispiel

die Parameter für den Lastausgleich), während andere erst beim Neustart des Kanals wirksam werden (zum Beispiel die TLS-Konfiguration).

Um Unklarheiten zu vermeiden, beachten Sie möglichst die folgenden Richtlinien:

- Definieren Sie nur CLUSSDR-Kanäle, die auf vollständige Repositorys verweisen, manuell.
- Wenn Sie über manuell definierte CLUSSDR-Kanäle verfügen, konfigurieren Sie diese so, dass sie genau mit der entsprechenden CLUSRCVR-Kanaldefinition auf dem empfangenden Warteschlangenmanager übereinstimmen.

Siehe auch Mit automatisch definierten Kanälen arbeiten.

**Zugehörige Konzepte**
Mit automatisch definierten Kanälen arbeiten
Arbeiten mit Clusterübertragungswarteschlangen und Clustersenderkanälen
**Zugehörige Tasks**
Neuen Cluster einrichten
WS-Manager zu einem Cluster hinzufügen

## Clusterthemen

Clusterthemen sind Verwaltungsthemen mit definiertem Attribut **`cluster`** . Informationen zu Clusterthemen werden an alle Clustermitglieder übertragen und mit lokalen Themen zu warteschlangenmanagerübergreifenden Thementeilbereichen verbunden. Damit können Nachrichten, die auf einem Warteschlangenmanager zu einem Thema veröffentlicht werden, an die Subskriptionen anderer Warteschlangenmanager im Cluster übermittelt werden.

Wenn Sie ein Clusterthema für einen Warteschlangenmanager definieren, wird diese Clusterthemendefinition an die Warteschlangenmanager mit vollständigem Repository gesendet. Anschließend leiten die vollständigen Repositorys die Clusterthemendefinition an alle Warteschlangenmanager im Cluster weiter, sodass das Clusterthema für alle Bereitsteller und Subskribenten verfügbar ist, die in einem Warteschlangenmanager im Cluster vorhanden sind. Der Warteschlangenmanager, in dem ein Clusterthema erstellt wird, wird als Clusterthemenhost bezeichnet. Das Clusterthema kann von allen Warteschlangenmanagern im Cluster verwendet werden; alle Änderungen an einem Clusterthema müssen jedoch in dem Warteschlangenmanager vorgenommen werden, in dem das Thema definiert ist (d. h. im Clusterthemenhost); anschließend wird die Änderung über die vollständigen Repositorys an alle Clustermitglieder weitergegeben.

Sie finden Informationen zur Konfiguration von Cluster-Topics für das *direkte Routing* oder *Topic-Host-Routing* sowie zur Cluster-Topic-Vererbung und Platzhaltersubskriptionen im Abschnitt Cluster-Topics definieren.

Informationen zu den Befehlen, mit denen Clusterthemen angezeigt werden, finden Sie in den zugehörigen Informationen.

**Zugehörige Konzepte**
Mit Verwaltungsthemen arbeiten
Mit Subskriptionen arbeiten
**Zugehörige Verweise**
ANZEIGETHEMA
ANZEIGETPSTATUS
ANZEIGEUNTERGEORDNET

## Standardclusterobjekte

Multi Auf Multiplatforms-Systemen gehören die Standardclusterobjekte zur Gruppe der Standardobjekte, die beim Definieren eines Warteschlangenmanagers automatisch erstellt werden. z/OS Unter z/OS sind die Definitionen zu Standardclusterobjekten in den Beispielen für die Anpassung enthalten.

**Anmerkung:** Sie können die Standardkanaldefinitionen auf dieselbe Weise wie jede andere Kanaldefinition durch die Ausführung von WebSphere MQ-Scriptbefehlen oder PCF-Befehlen ändern. Ändern Sie die Standardwarteschlangendefinitionen mit Ausnahme von SYSTEM.CLUSTER.HISTORY.QUEUE nicht.

**SYSTEM.CLUSTER.COMMAND.QUEUE**
Jeder Warteschlangenmanager in einem Cluster besitzt eine lokale Warteschlange namens SYSTEM.CLUSTER.COMMAND.QUEUE, über die Nachrichten an das vollständige Repository übertragen werden. Die Nachrichten enthalten neue oder geänderte Informationen über den Warteschlangenmanager selbst oder Anforderungen für Informationen über andere Warteschlangenmanager. SYSTEM.CLUSTER.COMMAND.QUEUE ist normalerweise leer.

**SYSTEM.CLUSTER.HISTORY.QUEUE**
Jeder Warteschlangenmanager in einem Cluster verfügt über eine lokale Warteschlange namens SYSTEM.CLUSTER.HISTORY.QUEUE. SYSTEM.CLUSTER.HISTORY.QUEUE wird verwendet, um das Protokoll der Clusterstatusinformationen zu Servicezwecken zu speichern.

In den Standardobjekteinstellungen ist SYSTEM.CLUSTER.HISTORY.QUEUE auf PUT ( ENABLED) gesetzt. Um die Erfassung von Protokolldaten zu unterdrücken, ändern Sie die Einstellung in PUT ( DISABLED).

**SYSTEM.CLUSTER.REPOSITORY.QUEUE**
Jeder Warteschlangenmanager in einem Cluster verfügt über eine lokale Warteschlange namens SYSTEM.CLUSTER.REPOSITORY.QUEUE. In dieser Warteschlange werden alle Informationen über das vollständige Repository gespeichert. Sie ist normalerweise nicht leer.

**SYSTEM.CLUSTER.TRANSMIT.QUEUE**
Jeder Warteschlangenmanager verfügt über eine Definition für eine lokale Warteschlange namens SYSTEM.CLUSTER.TRANSMIT.QUEUE. SYSTEM.CLUSTER.TRANSMIT.QUEUE ist die Standardübertragungswarteschlange für alle Nachrichten an alle Warteschlangen und Warteschlangenmanager innerhalb von Clustern. Sie können die Standardübertragungswarteschlange für jeden Clustersenderkanal in SYSTEM.CLUSTER.TRANSMIT. *ChannelName*ändern, indem Sie das Warteschlangenmanagerattribut DEFCLXQ ändern. Die Warteschlange SYSTEM.CLUSTER.TRANSMIT.QUEUE kann nicht gelöscht werden. Es wird auch verwendet, um Berechtigungsprüfungen festzulegen, ob die verwendete Standardübertragungswarteschlange SYSTEM.CLUSTER.TRANSMIT.QUEUE oder SYSTEM.CLUSTER.TRANSMIT. *ChannelName*ist.

**SYSTEM.DEF.CLUSRCVR**
In jedem Cluster gibt es eine Standarddefinition für den Clusterempfängerkanal (CLUSRCVR) mit dem Namen SYSTEM.DEF.CLUSRCVR. SYSTEM.DEF.CLUSRCVR liefert Standardwerte für alle Attribute, die Sie bei der Erstellung eines Clusterempfängerkanals für einen Warteschlangenmanager im Cluster nicht angeben.

**SYSTEM.DEF.CLUSSDR**
In jedem Cluster gibt es eine Standarddefinition für den Clustersenderkanal (CLUSSDR) mit dem Namen SYSTEM.DEF.CLUSSDR. SYSTEM.DEF.CLUSSDR liefert Standardwerte für alle Attribute, die Sie bei der Erstellung eines Clustersenderkanals für einen Warteschlangenmanager im Cluster nicht angeben.

**Zugehörige Konzepte**
Mit Standardclusterobjekten arbeiten

# Publish/Subscribe-Messaging

Das Publish/Subscribe-Messaging ermöglicht die Entkopplung des Informationsanbieters von den Informationsempfängern. Die Sender- und Empfängeranwendungen müssen untereinander keine Informationen über die gesendeten und empfangenen Nachrichten austauschen.

Bevor eine Punkt-zu-Punkt-Anwendung in IBM MQ eine Nachricht an eine andere Anwendung senden kann, werden Informationen zu dieser Anwendung benötigt. Beispielsweise ist der Name der Warteschlange erforderlich, an die die Information gesendet werden soll, und möglicherweise muss auch ein Warteschlangenmanagername angegeben werden.

Durch IBM MQ Publish/Subscribe entfällt für eine Anwendung die Notwendigkeit, Informationen zu der Zielanwendung zu haben. Die sendende Anwendung muss lediglich Folgendes ausführen:

- *Einreihen* einer IBM MQ -Nachricht, die die Informationen enthält, die die Anwendung wünscht.
- Die Nachricht einem Thema zuweisen, das den Betreff der Informationen angibt
- IBM MQ die Verteilung dieser Informationen überlassen

Ebenso muss auch die Zielanwendung keine Angaben zur Quelle der empfangenen Informationen haben.

In folgender Abbildung wird das einfachste Publish/Subscribe-System gezeigt. Es gibt einen Publisher, einen Warteschlangenmanager und einen Subskribenten. Eine Subskription wird von einem Subskribenten auf einem Warteschlangenmanager vorgenommen, eine Veröffentlichung wird vom Publisher an den Warteschlangenmanager gesendet und die Veröffentlichung wird anschließend vom Warteschlangenmanager an den Subskribenten weitergeleitet.

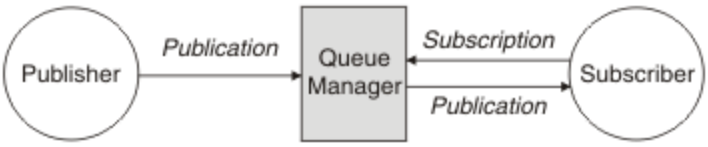



*Abbildung 17. Einfache Publish/Subscribe-Konfiguration*

Ein typisches Publish/Subscribe-System verfügt über mehrere Publisher und mehrere Subskribenten zu zahlreichen Themen und oft auch über mehr als einen Warteschlangenmanager. Eine Anwendung kann gleichzeitig Publisher und Subskribent sein.

Ein weiterer bedeutender Unterschied zwischen Publish/Subscribe- und Punkt-zu-Punkt-Messaging besteht darin, dass eine an eine Punkt-zu-Punkt-Warteschlange gesendete Nachricht nur von einer einzigen konsumierenden Anwendung verarbeitet wird. Eine für ein Publish/Subscribe-Thema veröffentlichte Nachricht, an der mehrere Subskribenten ihr Interesse gemeldet haben, wird von jedem interessierten Subskribenten verarbeitet.

## Publish/Subscribe-Komponenten

Publish/Subscribe ist der Mechanismus, über den Subskribenten von Publishern Informationen in Form von Nachrichten erhalten. Die Interaktionen zwischen Publishern und Subskribenten werden von Warteschlangenmanagern über Standardfunktionen von IBM MQ gesteuert.

Ein typisches Publish/Subscribe-System verfügt über mehrere Publisher und mehrere Subskribenten zu zahlreichen Themen und oft auch über mehr als einen Warteschlangenmanager. Eine Anwendung kann gleichzeitig Publisher und Subskribent sein.

Der Bereitsteller von Informationen wird als *Publisher* bezeichnet. Publisher stellen Informationen zu einem Thema bereit, ohne irgendetwas über die Anwendungen, für die diese Informationen bestimmt sind, wissen zu müssen. Publisher erstellen diese Informationen in Form von Nachrichten, so genannten *Veröffentlichungen*, deren Thema sie definieren und die sie veröffentlichen.

Der Konsument der Informationen wird als *Subskribent* bezeichnet. Subskribenten erstellen *Subskriptionen*, die das Thema beschreiben, an dem sie interessiert sind. Eine Subskription bestimmt also, welche Veröffentlichungen an den Subskribenten weitergeleitet werden. Subskribenten können mehrere Subskriptionen erstellen und Informationen von vielen verschiedenen Publishern erhalten.

Veröffentlichte Informationen werden in einer IBM MQ-Nachricht gesendet, wobei der Inhalt dieser Informationen durch das *Thema* angedeutet wird. Das Thema wird bei der Veröffentlichung der Informationen vom Publisher festgelegt. Der Subskribent wiederum gibt die Themen an, zu denen er Veröffentlichungen erhalten möchte. Der Subskribent erhält nur Informationen zu den von ihm subskribierten Themen.

Durch Themen können beim Publish/Subscribe-Messaging Informationsbereitsteller von den Konsumenten der Informationen getrennt werden, denn hier ist es im Gegensatz zum Punkt-zu-Punkt-Messaging nicht erforderlich, das Ziel einer Nachricht anzugeben.

Die Interaktionen zwischen Publishern und Subskribenten werden von einem Warteschlangenmanager gesteuert. Der Warteschlangenmanager empfängt die Nachrichten der Publisher und die Subskriptionen (zu bestimmten Themen) der Subskribenten. Die Aufgabe des Warteschlangenmanagers ist es, die veröffentlichten Nachrichten an die Subskribenten weiterzuleiten, die ihr Interesse an den Themen der Nachrichten registriert haben.

Die Nachrichten werden mithilfe der Standardfunktionen von IBM MQ verteilt. Ihre Anwendungen können daher alle Funktionen verwenden, die vorhandenen IBM MQ-Anwendungen zur Verfügung stehen. Für die zuverlässige Nachrichtenübermittlung können Sie also auch persistente Nachrichten verwenden. Ebenso können Ihre Nachrichten Teil einer transaktionsorientierten Arbeitseinheit sein, durch die sichergestellt wird, dass Nachrichten nur dann dem Subskribenten zugestellt werden, wenn sie vom Publisher festgeschrieben wurden.

## Veröffentlichungskomponenten und Veröffentlichungen

In der Publish/Subscribe-Funktion von IBM MQ bezeichnet eine Veröffentlichungskomponente eine Anwendung, die einem Warteschlangenmanager Informationen zu einem bestimmten Thema in Form einer IBM MQ-Standardnachricht zur Verfügung stellt, die "Veröffentlichung" genannt wird. Eine Veröffentlichungskomponente kann Informationen zu mehreren Themen veröffentlichen.

Veröffentlichungskomponenten verwenden das Verb MQPUT, um eine Nachricht in ein zuvor geöffnetes Thema einzureihen. Diese Nachricht ist eine Veröffentlichung. Der lokale Warteschlangenmanager leitet die Veröffentlichung dann an alle Subskribenten weiter, die Subskriptionen zum Thema der Veröffentlichung eingerichtet haben. Eine veröffentlichte Nachricht kann von mehreren Subskribenten gelesen werden.

Zusätzlich zum Verteilen von Veröffentlichungen an alle lokalen Subskribenten mit entsprechenden Subskriptionen kann ein Warteschlangenmanager eine Veröffentlichung auch an beliebige andere Warteschlangenmanager weiterleiten, die entweder direkt oder über ein Netz von Warteschlangenmanagern, die Subskribenten für dieses Thema verwalten, mit ihm verbunden sind.

In einem Publish/Subscribe-Netz von IBM MQ kann eine veröffentlichende Anwendung auch ein Subskribent sein.

### Veröffentlichungen unter Synchronisationspunkt

Veröffentlichungskomponenten können MQPUT- oder MQPUT1-Aufrufe im Rahmen einer Synchronisationspunktverarbeitung ausgeben, um alle Nachrichten einzuschließen, die in einer Arbeitseinheit an Subskribenten geliefert werden. Wenn die Option MQPMO_RETAIN oder die Themabereitstellungsoptionen NPMSGDLV und PMSGDLV mit dem Wert ALL oder ALLDUR angegeben werden, verwendet der Warteschlangenmanager interne MQPUT- oder MQPUT1-Aufrufe unter Synchronisationspunktverarbeitung innerhalb des Geltungsbereichs des MQPUT- oder MQPUT1-Aufrufs der Veröffentlichungskomponente.

#### *Status- und Ereignisinformationen*

Bei Veröffentlichungen kann es sich um Statusveröffentlichungen (z. B. der aktuelle Aktienkurs) oder um Ereignisveröffentlichungen (z. B. ein Aktienverkauf) handeln.

### Statusveröffentlichungen

*Statusveröffentlichungen* enthalten Informationen zum aktuellen Status oder Zustand von etwas, beispielsweise den Kurs einer Aktie oder den aktuellen Stand eines Fußballspiels. Sobald eine Änderung eintritt (sich also beispielsweise der Aktienpreis oder der Punktestand ändert), wird die vorherige Statusinformation nicht mehr benötigt, da sie durch die neue Information ersetzt wird.

Ein Subskribent möchte zu Beginn seiner Subskription die aktuelle Version der Statusinformationen erhalten. Diese möchte er aktualisiert haben, sobald sich der Status ändert.

Eine Veröffentlichung mit Statusinformationen wird meist als ständige Veröffentlichung veröffentlicht. Ein neuer Subskribent möchte die aktuellen Statusinformationen selbstverständlich sofort erhalten und nicht auf ein Ereignis warten müssen, das die erneute Veröffentlichung der Informationen auslöst. Subskriben-

ten erhalten die ständige Veröffentlichung eines Themas automatisch bei der Subskription, es sei denn, der Subskribent verwendet die Option MQSO_PUBLICATIONS_ON_REQUEST oder MQSO_NEW_PUBLICATIONS_ONLY.

## Ereignisveröffentlichungen

*Ereignisveröffentlichungen* enthalten Informationen zu einzelnen Ereignissen, beispielsweise zu einem Aktienhandel oder einem gefallenen Tor in einem Fußballspiel. Jedes Ereignis ist unabhängig von anderen Ereignissen.

Ein Subskribent möchte die Informationen zu einem Ereignis sofort erhalten, wenn es eintritt.

### *Ständige Veröffentlichungen*

Eine Veröffentlichung wird nach dem Senden an alle interessierten Subskribenten standardmäßig gelöscht. Für den Fall, dass später weitere Subskribenten ihr Interesse an der Veröffentlichung anmelden, kann der Publisher aber auch festlegen, dass eine Kopie der Veröffentlichung aufbewahrt wird.

An alle interessierte Subskribenten gesendete Veröffentlichungen können im Falle von Ereignisdaten meist gelöscht werden, im Falle von Statusinformationen häufig aber nicht. Wird eine solche Nachricht aufbewahrt, müssen neue Subskribenten nicht auf eine erneute Veröffentlichung der Informationen warten, um die anfänglichen Statusinformationen zu erhalten. Ein Subskribent eines Aktienkurses erhielte den aktuellen Kurs dann sofort, ohne dass er auf eine Änderung (und damit erneute Veröffentlichung) des Aktienkurses warten müsste.

Der Warteschlangenmanager kann pro Thema nur eine Veröffentlichung aufbewahren. Die vorhandene ständige Veröffentlichung eines Themas wird daher gelöscht, sobald beim Warteschlangenmanager für das gleiche Thema eine neue ständige Veröffentlichung eingeht. Jedoch kann es passieren, dass die vorhandene Veröffentlichung nicht exakt gleichzeitig mit der Ankunft der neuen Veröffentlichung gelöscht wird. Daher sollten Sie ständige Veröffentlichungen zum gleichen Thema möglichst nur von einem Publisher aus senden.

Mit der Subskriptionsoption MQSO_NEW_PUBLICATIONS_ONLY können Subskribenten angeben, dass sie keine ständigen Veröffentlichungen erhalten wollen. Genauso ist es aber auch möglich, Duplikatkopien ständiger Veröffentlichungen anzufordern.

Eventuell möchten Sie in bestimmten Fällen keine ständigen Veröffentlichungen erstellen, nicht einmal für Statusinformationen. Dies kann zum Beispiel in den folgenden Situationen der Fall sein:

- Wenn alle Subskriptionen zu einem Thema abgeschlossen wurden, bevor zu diesem Thema Veröffentlichungen vorlagen, und Sie nicht davon ausgehen, dass neue Subskriptionen eingehen (bzw. keine neuen Subskriptionen zulassen), besteht kein Grund zur Aufbewahrung ständiger Veröffentlichungen, da alle Veröffentlichungen bereits bei ihrer ersten Veröffentlichung allen Subskribenten zugestellt wurden.
- Wenn Veröffentlichungen in einer sehr schnellen Abfolge versendet werden, zum Beispiel jede Sekunde, erhält ein neuer Subskribent bzw. ein Subskribent nach einem Recovery den aktuellen Status nahezu sofort nach dem Abschluss der Subskription. Es besteht für solche Veröffentlichungen daher kein Grund zur Aufbewahrung.
- Bei sehr großen Veröffentlichungen und/oder sehr vielen Themen benötigen Sie erheblichen Speicherplatz zur Aufbewahrung der ständigen Veröffentlichungen zu allen Themen. In einer Umgebung mit mehreren Warteschlangenmanagern werden ständige Veröffentlichungen auf allen Warteschlangenmanagern im Netz gespeichert, auf denen eine entsprechende Subskription vorliegt.

In die Überlegung, ob ständige Veröffentlichungen erstellt werden sollen, sollte auch einfließen, wie die abonnierenden Anwendungen nach einem Fehler wiederhergestellt werden. Wenn der Publisher keine ständigen Veröffentlichungen verwendet, muss die abonnierende Anwendung ihren aktuellen Status eventuell lokal speichern.

Zur Aufbewahrung einer ständigen Veröffentlichung verwenden Sie die Put-Option MQPMO_RETAIN. Wenn diese Option angegeben, aber eine ständige Veröffentlichung nicht möglich ist, wird die betreffende Nachricht nicht veröffentlicht und der Aufruf schlägt mit MQRC_PUT_NOT_RETAINED fehl.

Eine ständige Veröffentlichung ist durch die Nachrichteneigenschaft MQIsRetained gekennzeichnet. Die Persistenz einer Nachricht wird bei ihrer ursprünglichen Veröffentlichung festgelegt und bleibt danach unverändert.

**Zugehörige Konzepte**

Designüberlegungen zu ständigen Veröffentlichungen in Publish/Subscribe-Clustern

### *Veröffentlichungen unter Synchronisationspunkt*

In der Publish/Subscribe-Funktion von IBM MQ kann der Synchronisationspunkt von Publishern oder intern vom Warteschlangenmanager verwendet werden.

Publisher verwenden den Synchronisationspunkt, wenn sie MQPUT/MQPUT1-Aufrufe mit der Option MQPMO_SYNCPOINT absetzen. Alle Nachrichten, die an Subskribenten zugestellt werden, werden bei der Berechnung der maximalen Anzahl von nicht festgeschriebenen Nachrichten in einer Arbeitseinheit berücksichtigt. Dieser Grenzwert wird über das Warteschlangenmanager-Attribut MAXUMSGS angegeben. Sobald der Grenzwert erreicht wird, empfängt der Publisher den Ursachencode 2024 (07E8) (RC2024): MQRC_SYNCPOINT_LIMIT_REACHED.

Wenn ein Publisher MQPUT/MQPUT1-Aufrufe unter Verwendung von MQPMO_NO_SYNCPOINT mit der Option MQPMO_RETAIN oder den Themenzustelloptionen NPMSGDLV/PMSGDLV mit den Werten ALL oder ALLDUR absetzt, verwendet der Warteschlangenmanager interne Synchronisationspunkte, um zu garantieren, dass die Nachrichten wie angefordert zugestellt werden. Der Publisher kann den Ursachencode 2024 (07E8) (RC2024): MQRC_SYNCPOINT_LIMIT_REACHED empfangen, wenn der Grenzwert im Bereich des Publisheraufrufs MQPUT/MQPUT1 erreicht wird.

## Subskribenten und Subskriptionen

Beim Publish/Subscribe von IBM MQ ist ein Subskribent eine Anwendung, die von einem Warteschlangenmanager in einem Publish/Subscribe-Netz Informationen zu einem bestimmten Thema anfordert. Ein Subskribent kann Nachrichten zu einem oder mehreren Themen von einem oder mehreren Publishern erhalten.

Subskriptionen können manuell mit einem WebSphere MQ-Scriptbefehl oder durch eine Anwendung erstellt werden. Diese Subskriptionen werden dem lokalen Warteschlangenmanager mit Informationen zu den Veröffentlichungen übergeben, die der Subskribent erhalten möchte:

- Das Thema, an dem der Subskribent interessiert ist. Wenn Platzhalter verwendet werden, können dies auch mehrere Themen sein.
- Eine optionale Auswahlzeichenfolge, die auf die veröffentlichten Nachrichten angewendet wird.
- Eine Kennung für eine Warteschlange (die *Subskribentenwarteschlange*), in die die ausgewählten Veröffentlichungen eingereiht werden sollen, sowie die optionale CorrelId.

Der lokale Warteschlangenmanager speichert die Subskriptionsinformationen und untersucht diese, sobald er eine Veröffentlichung erhält, um festzustellen, ob ihm eine Subskription mit dem Thema und der Auswahlzeichenfolge dieser Veröffentlichung vorliegt. Bei einer übereinstimmenden Subskription leitet der Warteschlangenmanager die Veröffentlichung an die entsprechende Subskribentenwarteschlange weiter. Die Informationen, die ein Warteschlangenmanager zu Subskriptionen speichert, können mit den Befehlen DIS SUB und DIS SBSTATUS angezeigt werden.

Eine Subskription wird nur im Falle der folgenden Ereignisse gelöscht:

- Der Subskribent hebt die Subskription mit dem Aufruf MQCLOSE auf (nur bei nicht persistenten Subskriptionen).
- Die Subskription läuft ab.
- Die Subskription wird vom Systemadministrator mit dem Befehl DELETE SUB gelöscht.
- Die Subskribentenanwendung wird beendet (nur bei nicht persistenten Subskriptionen).
- Der Warteschlangenmanager wird gestoppt oder neu gestartet (nur bei nicht persistenten Subskriptionen).

Achten Sie darauf, dass Sie beim Abrufen von Nachrichten im MQGET-Aufruf die richtigen Optionen angeben. Wenn Ihre Anwendung nur Nachrichten für eine Subskription verarbeitet, sollten Sie mindestens `get-by-correlid` verwenden, wie im C-Beispielprogramm `amqssbxa.c` und unter Nicht verwalteter MQ-Subskribent veranschaulicht. Die zu verwendende **`CorrelId`** wird von MQSUB im MQSD zurückgegeben.**`SubCorrelId`** .

**Zugehörige Konzepte**

Klone und gemeinsam genutzte Subskriptionen

**Zugehörige Verweise**

Beispiele für Definition der Eigenschaft sharedSubscription

### *Verwaltete Warteschlangen und Publish/Subscribe*

Wenn Sie eine Subskription erstellen, können Sie wählen, ob Sie die verwaltete Warteschlangensteuerung verwenden möchten. Falls Sie die verwaltete Warteschlangensteuerung verwenden, wird bei der Erstellung der Subskription automatisch eine Subskriptionswarteschlange erstellt. Verwaltete Warteschlangen werden entsprechend der Persistenz der Subskription automatisch bereinigt. Wenn Sie verwaltete Warteschlangen verwenden, brauchen Sie sich keine Gedanken mehr darüber zu machen, ob auch schon Warteschlangen für die Veröffentlichungen erstellt sind. Ebenso werden nicht konsumierte Veröffentlichungen automatisch aus den Subskribentenwarteschlangen gelöscht, sobald die Verbindung einer nicht persistenten Subskription geschlossen wird.

Muss für eine Anwendung keine bestimmte Warteschlange als Subskribentenwarteschlange verwendet werden (Ziel für die Veröffentlichungen, die die Anwendung subskribiert hat), können Sie für die Anwendung mit der Subskriptionsoption MQSO_MANAGED festlegen, dass sie *verwaltete Subskriptionen* verwendet. Bei der Erstellung einer verwalteten Subskription gibt der Warteschlangenmanager dem Subskribenten eine Objektkennung für eine Subskribentenwarteschlange zurück, die der Warteschlangenmanager erstellt und in die er die Veröffentlichungen für diese Subskription einreiht. Das ist darauf zurückzuführen, dass bei einer *verwalteten Subskription* IBM MQ die Subskription bearbeitet. Mithilfe dieser Objektkennung können Sie die Warteschlange suchen, ihren Inhalt abrufen und sie untersuchen. Deren Attribute können Sie allerdings nicht einstellen, es sei denn, Sie haben explizit Zugriff auf temporäre dynamische Warteschlangen.

Die Persistenz einer Subskription bestimmt, ob die verwaltete Warteschlange erhalten bleibt, wenn die Verbindung zwischen der abonnierenden Anwendung und dem Warteschlangenmanager getrennt wird.

Verwaltete Subskriptionen sind besonders praktisch in Verbindung mit nicht persistenten Subskriptionen, da andernfalls nach einer Beendigung der Verbindung der abonnierenden Anwendung nicht konsumierte Nachrichten in der Subskribentenwarteschlange verbleiben und auf dem Warteschlangenmanager für immer Speicherplatz belegen würden. Bei einer verwalteten Subskription ist die verwaltete Warteschlange hingegen eine temporäre dynamische Warteschlange und wird als solche zusammen mit nicht konsumierten Nachrichten gelöscht, wenn die Verbindung aus einer der folgenden Gründe getrennt wird:

- MQCLOSE wird mit MQCO_REMOVE_SUB ausgeführt und der verwaltete Hobj wird geschlossen.
- Eine Verbindung geht an eine Anwendung verloren, die eine nicht persistente Subskription (MQSO_NON_DURABLE) verwendet.
- Eine Subskription wird entfernt, da sie abgelaufen ist und der verwaltete Hobj geschlossen wurde.

Verwaltete Subskriptionen können auch bei persistenten Subskriptionen verwendet werden. In diesem Fall möchten Sie aber vermutlich nicht konsumierte Nachrichten in der Subskribentenwarteschlange belassen, sodass diese nach einer Wiederherstellung der Verbindung abgerufen werden können. Aus diesem Grund werden verwaltete Warteschlangen für persistente Subskriptionen als permanente dynamische Warteschlangen erstellt und bleiben erhalten, wenn die Verbindung der abonnierenden Anwendung zum Warteschlangenmanager getrennt wird.

Sie können für Ihre Subskription ein Ablaufdatum festlegen. Auch wenn sie eine Verbindungstrennung übersteht, bleibt sie in diesem Fall nicht für immer erhalten.

Wenn Sie eine verwaltete Warteschlange löschen, erhalten Sie eine Fehlernachricht.

An die Namen der verwalteten Warteschlangen wird am Ende eine Zeitmarke angefügt. Auf diese Weise erhält jede Warteschlange einen eindeutigen Namen.

### *Subskriptionspermanenz*

Subskriptionen können persistent oder nicht persistent sein. Die Persistenz einer Subskription bestimmt, was mit der Subskription geschieht, wenn die abonnierende Anwendung vom Warteschlangenmanager getrennt wird.

## Permanente Subskriptionen

Permanente Subskriptionen bleiben erhalten, wenn die Verbindung der subskribierenden Anwendung zum Warteschlangenmanager geschlossen wird. Die Subskription bleibt in diesem Fall auch bei einer Trennung der Verbindung erhalten und kann bei einer Wiederherstellung der Verbindung mit dem bei der Erstellung der Subskription zurückgegebenen **SubName** wieder von der abonnierenden Anwendung angefordert werden.

Bei der Erstellung einer persistenten Subskription ist ein Subskriptionsname (**SubName**) erforderlich. Subskriptionsnamen müssen innerhalb eines Warteschlangenmanagers eindeutig sein, sodass diese eine Subskription zweifelsfrei identifizieren. Diese Identifikation ist bei der Angabe einer Subskription erforderlich, die Sie wiederaufnehmen möchten, nachdem Sie deren Verbindung entweder absichtlich (mit der Option MQCO_KEEP_SUB) getrennt haben oder nachdem deren Verbindung vom Warteschlangenmanager getrennt wurde. Eine vorhandene Subskription können Sie mit dem Aufruf MQSUB mit der Option MQSO_RESUME wiederaufnehmen. Subskriptionsnamen werden auch angezeigt, wenn Sie den Befehl DISPLAY SBSTATUS mit SUBTYPE ALL oder ADMIN verwenden.

Wird eine persistente Subskription nicht mehr von der abonnierenden Anwendung benötigt, so kann diese mit dem Funktionsaufruf MQCLOSE und der Option MQCO_REMOVE_SUB bzw. manuell mit dem WebSphere MQ-Scriptbefehl DELETE SUB gelöscht werden.

Mit dem Themenattribut **DURSUB** können Sie angeben, ob für ein Thema permanente Subskriptionen eingerichtet werden können.

Während der Rückgabe eines MQSUB-Aufrufs mit der Option MQSO_RESUME wird der Subskriptionsablauf auf die ursprüngliche Ablaufzeit der Subskription gesetzt, nicht auf die verbleibende Ablaufzeit.

Ein Warteschlangenmanager sendet einer persistenten Subskription weiterhin Veröffentlichungen, selbst wenn zur Subskribentenanwendung keine Verbindung mehr besteht. Dies führt in der Subskribentenwarteschlange zu einem Rückstau. Die einfachste Methode, dies zu vermeiden, ist, wann immer möglich, die Verwendung von nicht persistenten Subskriptionen. Ist eine persistente Subskription absolut erforderlich, so lässt sich ein solcher Rückstau aber auch vermeiden, wenn die Subskription mit der Option Ständige Veröffentlichungen erstellt wird. Ein Subskribent kann dann mit dem Aufruf MQSUBRQ steuern, wann er Veröffentlichungen empfängt.

## Nicht persistente Subskriptionen

Nicht persistente Subskriptionen bleiben nur so lange bestehen, wie die Verbindung einer Subskribentenanwendung zu einem Warteschlangenmanager geöffnet bleibt. Die Subskription wird entfernt, wenn die abonnierende Anwendung absichtlich oder durch eine Verbindungsunterbrechung vom Warteschlangenmanager getrennt wird. Beim Schließen der Verbindung werden die Informationen zur Subskription vom Warteschlangenmanager gelöscht. Die Subskription tritt dann auch bei der Anzeige der Subskriptionen mit dem Befehl DISPLAY SBSTATUS nicht mehr in Erscheinung. Nach der Beendigung der Subskription werden keine weiteren Nachrichten mehr in die Subskribentenwarteschlange gestellt.

Was bei nicht persistenten Subskriptionen mit noch nicht konsumierten Veröffentlichungen in der Subskribentenwarteschlange geschieht, wird wie folgt bestimmt:

- Wenn eine abonnierende Anwendung ein verwaltetes Ziel verwendet, werden alle noch nicht konsumierten Veröffentlichungen automatisch entfernt.
- Hat die abonnierende Anwendung bei der Subskription eine Kennung ihrer eigenen Subskribentenwarteschlange bereitgestellt, so werden nicht konsumierte Veröffentlichungen nicht automatisch entfernt. In diesem Fall ist die Anwendung selbst dafür verantwortlich, die Warteschlange zu bereinigen. Falls die Warteschlange auch von anderen Subskribenten oder von Punkt-zu-Punkt-Anwendungen genutzt wird, darf die Warteschlange vermutlich nicht vollständig geleert werden.

Auch wenn dies für nicht persistente Subskriptionen nicht erforderlich ist, verwendet der Warteschlangenmanager den Subskriptionsnamen, sofern er bereitgestellt wird. Subskriptionsnamen müssen innerhalb eines Warteschlangenmanagers eindeutig sein, sodass diese eine Subskription zweifelsfrei identifizieren.

**Zugehörige Konzepte**

Klone und gemeinsam genutzte Subskriptionen

**Zugehörige Tasks**

Gemeinsam genutzte JMS 2.0-Subskriptionen verwenden

**Zugehörige Verweise**

Beispiele für Definition der Eigenschaft sharedSubscription

### *Auswahlzeichenfolgen*

Eine *Auswahlzeichenfolge* ist ein Ausdruck, der auf eine Veröffentlichung angewendet wird, um festzustellen, ob sie mit einer Subskription übereinstimmt. Auswahlzeichenfolgen können Platzhalterzeichen einschließen.

Beim Einrichten einer Subskription können Sie neben einem Thema auch eine Auswahlzeichenfolge angeben, um Veröffentlichungen anhand ihrer Nachrichteneigenschaften auszuwählen.

Die Auswahlzeichenfolge wird mit der Nachricht, so wie sie vom Publisher bereitgestellt wurde, abgeglichen, bevor diese für die Zustellung an die einzelnen Subskribenten modifiziert wird. Seien Sie vorsichtig, wenn Sie Felder in der Auswahlzeichenfolge verwenden, die möglicherweise im Rahmen des Veröffentlichungsvorgangs modifiziert werden. Dies gilt beispielsweise für die MQMD-Felder `UserIdentifier` (Benutzer-ID), `MsgId` (Nachrichten-ID) und `CorrelId` (Korrelations-ID).

Auswahlzeichenfolgen sollten auf keins der Nachrichteneigenschaftsfelder verweisen, die im Rahmen des Veröffentlichungsvorgangs vom Warteschlangenmanager hinzugefügt werden (siehe Publish/Subscribe-Nachrichteneigenschaften), außer auf das Nachrichteneigenschaftsfeld `MQTopicString`, die die Themenzeichenfolge für die Veröffentlichung enthält.

**Zugehörige Konzepte**

Regeln und Einschränkungen für Auswahlzeichenfolgen

## Themen

Ein Thema ist der Betreff der Informationen, die in einer Publish/Subscribe-Nachricht veröffentlicht werden.

Nachrichten in Punkt-zu-Punkt-Systemen werden an eine bestimmte Zieladresse gesendet. Nachrichten in betreffbasierten Publish/Subscribe-Systemen werden auf Basis des Betreffs, der den Inhalt der Nachricht beschreibt, an Subskribenten gesendet. Bei inhaltsbasierten Systemen werden Nachrichten auf Basis des Nachrichteninhalts an Subskribenten gesendet.

Das Publish/Subscribe-System von IBM MQ ist ein betreffbasiertes Publish/Subscribe-System. Ein Publisher erstellt eine Nachricht und veröffentlicht sie mit einer Themenzeichenfolge, die dem Betreff der Veröffentlichung am besten entspricht. Um Veröffentlichungen zu empfangen, erstellt eine Subskribent eine Subskription mit einer Zeichenfolge mit Mustererkennung, um Veröffentlichungsthemen auszuwählen. Der Warteschlangenmanager liefert Veröffentlichungen an Subskribenten, deren Subskriptionen dem Veröffentlichungsthema entsprechen und für den Empfang von Veröffentlichungen autorisiert sind. Der Artikel „Themenzeichenfolgen” auf Seite 75 beschreibt die Syntax von Themenzeichenfolgen, die den Betreff einer Veröffentlichung angeben. Subskribenten erstellen zudem auch Themazeichenfolgen, um die zu empfangenden Themen auszuwählen. Die Themazeichenfolgen, die Subskribenten erstellen, können eines von zwei Platzhalterschemas zur Musterübereinstimmung mit den Themazeichenfolgen in Veröffentlichungen enthalten. Informationen zur Musterübereinstimmung finden Sie in „Platzhalterschemas” auf Seite 76.

Beim betreffbasierten Publish/Subscribe sind Publisher oder Administratoren für die Klassifizierung von Betreffs in Artikeln verantwortlich. Normalerweise werden Betreffs hierarchisch in Themenstrukturen organisiert. Mit dem Zeichen `'/'` werden in der Themazeichenfolge Unterthemen erstellt. Beispiele zu Themenstrukturen finden Sie in „Themenstrukturen” auf Seite 82. Themen sind Knoten in einer

Themenstruktur. Themen können Blattknoten ohne weitere Unterthemen oder Zwischenknoten mit Unterthemen sein.

Parallel zur Verwaltung von Betreffs in hierarchischen Themenstrukturen können Sie administrativen Themenobjekten Themen zuordnen. Sie weisen einem Thema Attribute zu, z. B., ob das Thema in einem Cluster verteilt wird, indem Sie es einem administrativen Themenobjekt zuordnen. Die Zuordnung findet über die Benennung des Themas unter Verwendung des Attributs TOPICSTR des administrativen Themenobjekts statt. Wenn Sie ein administratives Themenobjekt nicht explizit einem Thema zuordnen, übernimmt das Thema die Attribute seines nächsten Vorgängers in der Themenstruktur, den Sie einem administrativen Themenobjekt *zugeordnet* haben. Wenn Sie keine übergeordneten Themen definiert haben, übernimmt das Thema die Attribute von SYSTEM.BASE.TOPIC. Eine Beschreibung der administrativen Themenobjekte finden Sie in „Verwaltungsthemenobjekte” auf Seite 83.

**Anmerkung:** Selbst wenn Sie alle Attribute eines Themas von SYSTEM.BASE.TOPIC übernehmen, müssen Sie ein Stammthema für Ihre Themen definieren, das die Attribute von SYSTEM.BASE.TOPIC direkt übernimmt. Beispielsweise ist im Themenbereich 'US-Bundesstaaten' (USA/Alabama, USA/Alaska usw.) USA das Stammthema. Hauptzweck eines Stammthemas ist es, getrennte, nicht überlappende Themenbereiche zu erstellen, um zu verhindern, dass Veröffentlichungen den falschen Subskriptionen entsprechen. Zudem können Sie auf diese Weise die Attribute Ihres Stammthemas ändern, um den gesamten Themenbereich entsprechend zu ändern. Sie können beispielsweise den Namen für das Attribut **CLUSTER** festlegen.

Wenn Sie als Publisher oder Subskribent auf ein Thema verweisen, können Sie eine Themazeichenfolge angeben oder auf ein Themenobjekt verweisen. Sie können auch beides tun, womit die von Ihnen angegebene Themazeichenfolge ein Unterthema des Themenobjekts definiert. Der Warteschlangenmanager identifiziert das Thema, indem er die Themazeichenfolge an das Präfix der im Themaobjekt genannten Themazeichenfolge anhängt. Dabei fügt er ein weiteres ' / ' -Zeichen zwischen die beiden Themazeichenfolgen ein, z. B. *Themazeichenfolge/Objektzeichenfolge*. Unter „Themenzeichenfolgen kombinieren” auf Seite 80 finden Sie weitere Informationen hierzu. Die daraus resultierende Themazeichenfolge dient zur Ermittlung des Themas und dessen Zuordnung zu einem administrativen Themenobjekt. Das administrative Themenobjekt ist nicht zwingend dasselbe Themenobjekt wie das Themenobjekt, das dem Masterthema entspricht.

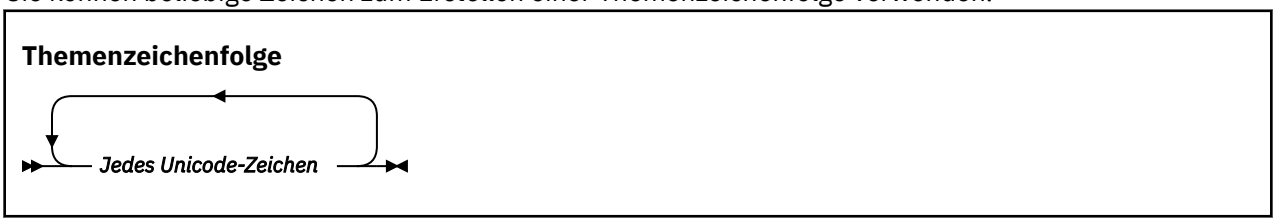
Beim inhaltsbasierten Publish/Subscribe definieren Sie, welche Nachrichten Sie empfangen möchten, indem Sie Auswahlzeichenfolgen angeben, die den Inhalt jeder Nachricht durchsuchen. IBM MQ stellt eine temporäre Form des inhaltsbasierten Publish/Subscribe bereit, bei dem Nachrichtenselektoren verwendet werden, die nicht den vollständigen Nachrichteninhalt, sondern die Nachrichteneigenschaften durchsuchen (siehe Selektoren). Normalerweise werden Nachrichtenselektoren so eingesetzt, dass zunächst ein Thema abonniert und die Auswahl dann mit einer numerischen Eigenschaft klassifiziert wird. Über den Selektor können Sie anzugeben, dass Sie nur an Werten in einem bestimmten Bereich interessiert sind. Dies ist mit Zeichen oder themenbasierten Platzhaltern nicht möglich. Wenn Sie auf Grundlage des gesamten Inhalts der Nachricht filtern möchten, müssen Sie IBM Integration Bus verwenden.

### *Themenzeichenfolgen*

Kennsatzinformationen, die Sie mithilfe einer Themenzeichenfolge als ein Thema veröffentlichen. Abonnieren Sie Themengruppen, indem Sie entweder zeichenbasierte oder themenbasierte Platzhalterthemenzeichenfolgen verwenden.

## Themen

Eine *Themenzeichenfolge* ist eine Zeichenfolge, die das Thema einer Publish/Subscribe-Nachricht angibt. Sie können beliebige Zeichen zum Erstellen einer Themenzeichenfolge verwenden.



Im Publish/Subscribe von IBM WebSphere MQ 7 haben drei Zeichen eine besondere Bedeutung. Sie sind zwar überall in einer Themenzeichenfolge zulässig, sollten aber mit Vorsicht verwendet werden. Die Verwendung der Sonderzeichen wird im Abschnitt „Themenbasiertes Platzhalterschema” auf Seite 77 erläutert.

**Schrägstrich (/)**

Dies ist das Trennzeichen für Themenebenen. Verwenden Sie das Zeichen '/', um eine Themenstruktur für das Thema zu erstellen.

Vermeiden Sie, wenn es möglich ist, leere Themenebenen ('//'). Denn diese entsprechen Knoten in der Themenhierarchie ohne Themenzeichenfolge. Ein führender oder abschließender '/' in einer Themenzeichenfolge entspricht einem führenden oder abschließenden leeren Knoten und sollte ebenfalls vermieden werden.

**Nummernzeichen (#)**

Wird in Kombination mit '/' verwendet, um in Subskriptionen ein Platzhalterzeichen für mehrere Ebenen zu erstellen. Geben Sie acht bei der Verwendung einer Kombination aus '#' und '/' in Themenzeichenfolgen, die zum Benennen von veröffentlichten Themen dienen. „Beispiele für Themenzeichenfolgen” auf Seite 76 zeigt eine überlegte Verwendung von '#'.

Die Zeichenfolgen '.../#/...', '#/...' und '.../#' haben in Subskriptionsthemenzeichenfolgen eine besondere Bedeutung. Die Zeichenfolgen stimmen mit allen Themen auf einer oder mehreren Ebenen in der Themenhierarchie überein. Wenn Sie ein Thema mit einer dieser Folgen erstellt haben, könnten Sie dieses Thema deshalb nicht abonnieren, ohne auch alle Themen auf mehreren Ebenen in der Themenhierarchie zu abonnieren.

**Pluszeichen (+)**

Wird in Kombination mit '/' verwendet, um in Subskriptionen ein Platzhalterzeichen für eine einzelne Ebene zu erstellen. Geben Sie acht bei der Verwendung einer Kombination aus '+' und '/' in Themenzeichenfolgen, die zum Benennen von veröffentlichten Themen dienen.

Die Zeichenfolgen '.../+/...', '+/...' und '.../+' haben in Subskriptionsthemenzeichenfolgen eine besondere Bedeutung. Die Zeichenfolgen stimmen mit allen Themen auf einer einzelnen Ebene in der Themenhierarchie überein. Wenn Sie ein Thema mit einer dieser Folgen erstellt haben, könnten Sie dieses Thema deshalb nicht abonnieren, ohne auch alle Themen auf einer einzelnen Ebene in der Themenhierarchie zu abonnieren.

### Beispiele für Themenzeichenfolgen

```
IBM/Business Area#/Results
IBM/Diversity/%African American
```

### Zugehörige Verweise

TOPIC

*Platzhalterschemas*
Es gibt zwei Platzhalterschemas, die zum Abonnieren mehrerer Themen verwendet werden. Die Auswahl des Schemas ist eine Subskriptionsoption.

**MQSO_WILDCARD_TOPIC**
Wählen Sie zu abonnierende Themen mithilfe des themenbasierten Platzhalterschemas aus.

Dies ist der Standard, wenn kein Platzhalterschema explizit ausgewählt ist.

**MQSO_WILDCARD_CHAR**
Wählen Sie zu abonnierende Themen mithilfe des zeichenbasierten Platzhalterschemas aus.

Legen Sie eins der Schemas fest, indem Sie den Parameter **wschema** im Befehl DEFINE SUB angeben. Weitere Informationen finden Sie im Abschnitt DEFINE SUB.

**Anmerkung:** In Subskriptionen, die vor IBM WebSphere MQ 7.0 erstellt wurden, wird immer das zeichenbasierte Platzhalterschema verwendet.

**Beispiele**

```
IBM/+/Results
#/Results
IBM/Software/Results
IBM/*ware/Results
```

*Themenbasiertes Platzhalterschema*
Themenbasierte Platzhalterzeichen ermöglichen es Subskribenten, mehrere Themen gleichzeitig zu abonnieren.

Themenbasierte Platzhalterzeichen sind eine leistungsfähige Funktion des Themensystems in IBM MQ Publish/Subscribe. Die Platzhalter für mehrere Ebenen bzw. für nur eine Ebene können für Subskriptionen, jedoch nicht innerhalb eines Themas durch den Publisher einer Nachricht verwendet werden.

Mithilfe des themenbasierten Platzhalterschemas können Sie Veröffentlichungen gruppiert nach Themenebene auswählen. Sie können für jede Ebene in der Themenhierarchie auswählen, ob die Zeichenfolge in der Subskription für die betreffende Themenebene mit der Zeichenfolge in der Veröffentlichung genau übereinstimmen muss oder nicht. Beispiel: Die Subskription `IBM/+/Results` wählt alle Themen aus.

```
IBM/Software/Results
IBM/Services/Results
IBM/Hardware/Results
```

Es gibt zwei Typen von Platzhalterzeichen:

**Platzhalterzeichen für mehrere Ebenen**

- Das Platzhalterzeichen für mehrere Ebenen wird in Subskriptionen verwendet. Bei Verwendung in einer Veröffentlichung wird es als Literal behandelt.
- Das Platzhalterzeichen `'#'` für mehrere Ebenen deckt alle Ebenen innerhalb eines Themas ab. Für das Themenstrukturbeispiel bedeutet dies, dass Sie bei einer Subskription von `'USA/Alaska/#'` Nachrichten zu den Themen `'USA/Alaska'` und `'USA/Alaska/Juneau'` empfangen.
- Das Platzhalterzeichen für mehrere Ebenen kann auf keine oder mehrere Ebenen zutreffen. Deshalb kann `'USA/#'` auch die einzelne Ebene `'USA'` abdecken, wobei `'#'` auf keine Ebene zutrifft. Das Trennzeichen für Themenebenen ist in diesem Kontext bedeutungslos, weil es keine zu trennenden Ebenen gibt.
- Das Platzhalterzeichen für mehrere Ebenen ist nur wirksam, wenn es allein steht oder neben dem Trennzeichen für Themenebenen. Deshalb sind `'#'` und `'USA/#'` gültige Themen, wobei das Zeichen `'#'` als Platzhalter behandelt wird. Aber obwohl `'USA#'` ebenfalls eine gültige Themenzeichenfolge ist, wird das Zeichen `'#'` nicht als Platzhalter angesehen und hat deshalb keine besondere Bedeutung. Weitere Informationen finden Sie im Abschnitt „Wenn themenbasierte Platzhalterzeichen unwirksam sind” auf Seite 79.

**Platzhalterzeichen für einzelne Ebenen**

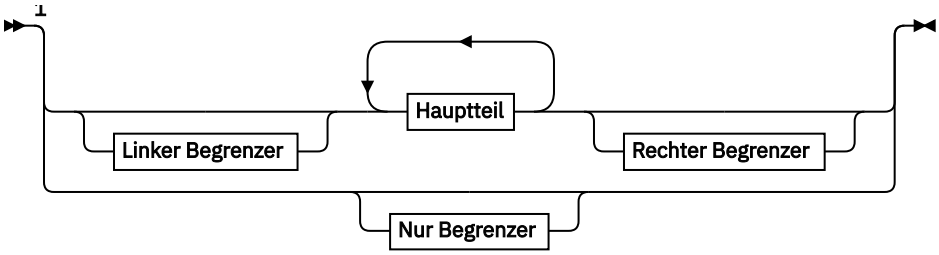
- Das Platzhalterzeichen für einzelne Ebenen wird in Subskriptionen verwendet. Bei Verwendung in einer Veröffentlichung wird es als Literal behandelt.
- Das Platzhalterzeichen `'+'` für einzelne Ebenen trifft nur auf eine einzige Themenebene zu. Beispielsweise deckt `'USA/+'` die Ebene `'USA/Alabama'` ab, aber nicht die Ebene `'USA/Alabama/Auburn'`. Da das Platzhalterzeichen für einzelne Ebenen nur für eine einzige Ebene zutrifft, deckt `'USA/+'` auch nicht die Ebene `'USA'` ab.
- Das Platzhalterzeichen für einzelne Ebenen kann auf jeder Ebene in der Themenstruktur und in Verbindung mit dem Platzhalterzeichen für mehrere Ebenen verwendet werden. Es muss neben dem Trennzeichen für Themenebenen angegeben werden, außer wenn es allein angegeben wird. Deshalb sind `'+'` und `'USA/+'` gültige Themen, wobei das Zeichen `'+'` als Platzhalter behandelt wird. Aber obwohl `'USA+'` ebenfalls eine gültige Themenzeichenfolge ist, wird das Zeichen `'+'` nicht als Platzhalter angesehen und hat deshalb keine besondere Bedeutung. Weitere Informatio-

nen finden Sie im Abschnitt „Wenn themenbasierte Platzhalterzeichen unwirksam sind” auf Seite 79.

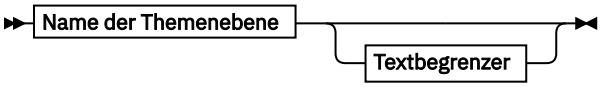
Die Syntax für das themenbasierte Platzhalterschema umfasst keine Escapezeichen. Ob '#' und '+' als Platzhalterzeichen behandelt werden oder nicht, hängt von ihrem Kontext ab. Weitere Informationen finden Sie unter „Wenn themenbasierte Platzhalterzeichen unwirksam sind” auf Seite 79.

**Anmerkung:** Anfang und Ende einer Themenzeichenfolge werden auf besondere Weise behandelt. Wenn Sie '$' verwenden, um das Ende der Zeichenfolge anzugeben, ist '$#/...' ein Platzhalter für mehrere Ebenen und '$/#/..'. ist ein leerer Knoten im Stammverzeichnis, gefolgt von einem Platzhalter für mehrere Ebenen.

**Themenbasierte Platzhalterzeichenfolge**

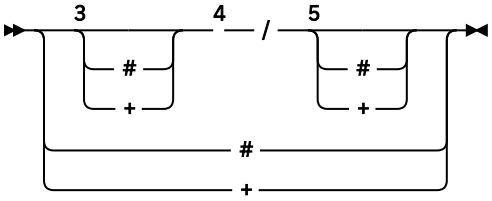



**Hauptteil**




**Name der Themenebene**

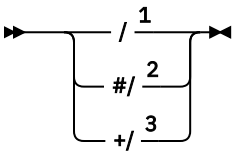

**Nur Begrenzer**



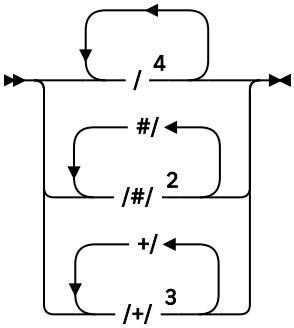

Anmerkungen:

[1] Eine Nullzeichenfolge oder leere Zeichenfolge als Themenzeichenfolge ist ungültig.
[2] Es wird empfohlen, keine *, ?, % in Ebenennamenzeichenfolgen zu verwenden, um die Kompatibilität zwischen zeichenbasierten und themenbasierten Platzhalterschemas zu gewährleisten.
[3] Diese Fälle entsprechen dem Muster von *Linker Begrenzer*.
[4] / ohne Platzhalterzeichen trifft auf ein einzelnes leeres Thema zu.
[5] Diese Fälle entsprechen dem Muster von *Rechter Begrenzer*.
[6] Trifft auf jedes Thema zu.
[7] Trifft auf jedes Thema zu, für das es nur eine Ebene gibt.

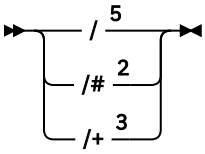
**Linker Begrenzer**




**Textbegrenzer**




**Rechter Begrenzer**




Anmerkungen:

[1] Die Themenzeichenfolge beginnt mit einem leeren Thema.
[2] Trifft auf keine oder mehrere Ebenen zu. Mehrere übereinstimmende Zeichenfolgen für mehrere Ebenen habe dieselbe Wirkung wie eine einzige übereinstimmende Zeichenfolge für mehrere Ebenen.
[3] Trifft auf genau eine Ebene zu.
[4] // ist ein leeres Thema - ein Themenobjekt ohne Themenzeichenfolge.
[5] Die Themenzeichenfolge endet mit einem leeren Thema.

## Wenn themenbasierte Platzhalterzeichen unwirksam sind

Die Platzhalterzeichen '+' und '#' haben keine besondere Bedeutung, wenn sie in einer Themenebene mit anderen Zeichen (einschließlich sich selbst) gemischt werden.

Dies bedeutet, dass Themen, die in einer Themenebene das Zeichen '+' oder '#' zusammen mit anderen Zeichen enthalten, veröffentlicht werden können.

Betrachten Sie beispielsweise die beiden folgenden Themen:

1. level0/level1/+/level4/#
2. level0/level1/#+/level4/level#

Im ersten Beispiel werden die Zeichen '+' und '#' als Platzhalterzeichen behandelt und sind deshalb in einer Themenzeichenfolge, für die Veröffentlichungen erfolgen sollen, nicht gültig, wohingegen sie in einer Subskription gültig sind.

Im zweiten Beispiel werden die Zeichen '+' und '#' nicht als Platzhalterzeichen behandelt, weshalb für die Themenzeichenfolge sowohl Veröffentlichungen als auch Subskriptionen möglich sind.

### Beispiele

```
IBM/+/Results
#/Results
IBM/Software/Results
```

*Zeichenbasiertes Platzhalterschema*
Mit dem zeichenbasierten Platzhalterschema können Sie Themen auf Grundlage des Abgleichs von konventionellen Zeichen auswählen.

Mit der Zeichenfolge '*' können Sie alle Themen auf mehreren Ebenen in einer Themenhierarchie auswählen. Die Verwendung von '*' im zeichenbasierten Platzhalterschema entspricht der Verwendung der themenbasierten Platzhalterzeichenfolge '#'

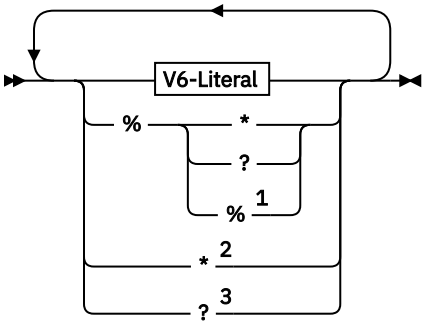
' *x*/*/*y* ' entspricht ' *x*/#/*y* ' im themenbasierten Schema und wählt alle Themen in der Themenhierarchie zwischen den Ebenen ' *x* und *y* 'aus, wobei ' *x* ' und ' *y* ' Themennamen sind, die nicht in der Gruppe der Ebenen enthalten sind, die vom Platzhalterzeichen zurückgegeben werden.

'/+/' im themenbasierten Schema hat keine exakte Entsprechung im zeichenbasierten Schema. 'IBM/*/Results' würde auch 'IBM/Patents/Software/Results'auswählen. Nur wenn die Gruppe der Themennamen in jeder Ebene der Hierarchie eindeutig ist, können Sie in jedem Fall Abfragen mit den beiden Schemas erstellen, die identische Übereinstimmungen ergeben.

Bei einer allgemeinen Verwendung haben '*' und '?' im zeichenbasierten Schema keine Entsprechungen im themenbasierten Schema. Das themenbasierte Schema führt keinen unvollständigen Abgleich mithilfe von Platzhaltern aus. Die zeichenbasierte Platzhaltersubskription 'IBM/*ware/Results' hat keine themenbasierte Entsprechung.

**Anmerkung:** Abgleiche mithilfe von Subskriptionen mit Platzhalterzeichen sind langsamer als Abgleiche mithilfe von themenbasierten Subskriptionen.

**Zeichenbasierte Platzhalterzeichen**




**V6-Literal**


Anmerkungen:

[1] Bedeutet "Folgendem Zeichen Escapezeichen voranstellen", damit es als Literal verwendet wird. Auf '%' muss entweder '*', '?' oder '%'folgen. Siehe „Beispiele für Themenzeichenfolgen” auf Seite 76.

[2] Bedeutet "Übereinstimmung mit null oder mehr Zeichen" in einer Subskription.

[3] Bedeutet "Übereinstimmung mit genau einem Zeichen" in einer Subskription.

**Beispiele**

```
IBM/*/Results
IBM/*ware/Results
```

## *Themenzeichenfolgen kombinieren*

Wenn Sie Subskriptionen erstellen oder Themen öffnen, um Nachrichten für sie veröffentlichen zu können, kann die Themenzeichenfolge gebildet werden, indem zwei separate Unterthemenzeichenfolgen (oder "Unterthemen") kombiniert werden. Das eine Unterthema wird von der Anwendung oder vom Verwaltungsbefehl als Themenzeichenfolge bereitgestellt und das andere ist die Themenzeichenfolge, die einem Themenobjekt zugeordnet ist. Sie können eines der Unterthemen als Themenzeichenfolge verwenden oder die Unterthemen kombinieren, um einen neuen Themennamen zu bilden.

Wenn Sie beispielsweise eine Subskription mit dem MQSC-Befehl **DEFINE SUB**definieren, kann der Befehl entweder **TOPICSTR** (Themenzeichenfolge) oder **TOPICOBJ** (Themenobjekt) als Attribut oder beides zusammen verwenden. Wird nur **TOPICOBJ** angegeben, wird die Themenzeichenfolge, die diesem Themenobjekt zugeordnet ist, als Themenzeichenfolge verwendet. Wird nur **TOPICSTR** angegeben, wird eben diese als Themenzeichenfolge verwendet. Werden beide angegeben, werden sie zu einer einzigen Themenzeichenfolge im Format **TOPICOBJ** / **TOPICSTR** verkettet, wobei **TOPICOBJ** immer den ersten Teil bildet und beide Teile immer durch das Zeichen "/" getrennt sind.

Entsprechend wird in einem MQI-Programm von MQOPEN der vollständige Themenname erstellt. Er besteht aus zwei Feldern, die in Publish/Subscribe-MQI-Aufrufen verwendet werden, in der aufgeführten Reihenfolge:

1. Das Attribut **TOPICSTR** des Themenobjekts, das im Feld **ObjectName** benannt ist.
2. Der Parameter **ObjectString**, der das von der Anwendung bereitgestellte Unterthema definiert.

Die daraus resultierende Themenzeichenfolge wird im Parameter **ResObjectString** zurückgegeben.

Diese Felder gelten als vorhanden, wenn das erste Zeichen jedes Felds kein Leerzeichen oder Nullzeichen ist und die Feldlänge größer als null ist. Wenn nur eines der Felder vorhanden ist, wird es unverändert als Themenname verwendet. Wenn keines der beiden Felder einen Wert enthält, schlägt der Aufruf mit dem Ursachencode MQRC_UNKNOWN_OBJECT_NAME oder MQRC_TOPIC_STRING_ERROR fehl, falls der vollständige Themenname ungültig ist.

Wenn beide Felder vorhanden sind, wird das Zeichen "/" zwischen die beiden Elemente des entstandenen kombinierten Themennamens eingefügt.

Die folgende Tabelle enthält Beispiele für die Verkettung von Themenzeichenfolgen:

*Tabelle 2. Beispiele für die Verkettung von Themenzeichenfolgen*

| **TOPICSTR des Themenobjekts** | **Themenzeichenfolge, die von der Anwendung oder dem Befehl DEFINE SUB bereitgestellt wird.** | **Vollständiger Themenname** | **Kommentar** |
|---|---|---|---|
| Fußball/Ergebnisse | '' | Fußball/Ergebnisse | Es wird nur TOPICSTR des Themenobjekts verwendet. |
| '' | Fußball/Ergebnisse | Fußball/Ergebnisse | Es wird nur ObjectString/TOPICSTR verwendet. |
| Fußball | Ergebnisse | Fußball/Ergebnisse | Am Verkettungspunkt wird das Zeichen "/" hinzugefügt. |
| Fußball | /Ergebnisse | Fußball//Ergebnisse | Zwischen den beiden Zeichenfolgen entsteht ein "leerer Knoten". Dies ist anders als bei "Fußball/Ergebnisse". |
| /Fußball | Ergebnisse | /Fußball/Ergebnisse | Das Thema beginnt mit einem "leeren Knoten". Dies ist anders als bei "Fußball/Ergebnisse". |

Das Zeichen "/" wird als Sonderzeichen betrachtet, das eine Struktur für den vollständigen Themennamen in „Themenstrukturen” auf Seite 82 angibt. Das Zeichen "/" darf zu keinem anderen Zweck verwendet werden, da die Themenstruktur betroffen ist. Das Thema "/Football" entspricht nicht dem Thema "Football".

**Anmerkung:** Wenn Sie beim Erstellen einer Subskription ein Themenobjekt verwenden, wird der Wert der Themenzeichenfolge des Themenobjekts zum Zeitpunkt der Definition in der Subskription festgelegt. Nachfolgende Änderungen des Themenobjekts haben keine Auswirkung auf die Themenzeichenfolge, für die die Subskription definiert ist.

## Platzhalterzeichen in Themenzeichenfolgen

Die folgenden Platzhalterzeichen sind Sonderzeichen:

- Pluszeichen (+)
- Nummernzeichen (#)
- Stern (*)
- Fragezeichen (?)

Platzhalterzeichen haben nur eine besondere Bedeutung, wenn sie von einer Subskription verwendet werden. Diese Zeichen werden nicht als ungültig betrachtet, wenn sie woanders verwendet werden, aber Sie müssen sehr genau wissen, wie sie verwendet werden, und verzichten eventuell lieber darauf, diese Zeichen bei der Veröffentlichung oder Definition von Themenobjekten in Ihren Themenzeichenfolgen zu verwenden.

Bei Veröffentlichungen für eine Themenzeichenfolge mit # oder + in Kombination mit anderen Zeichen (einschließlich sich selbst) innerhalb einer Themenebene kann die Themenzeichenfolge mit beiden Platzhalterschemas subskribiert werden.

Bei Veröffentlichungen für eine Themenzeichenfolge mit # oder + als dem einzigen Zeichen zwischen zwei /-Zeichen kann die Themenzeichenfolge nicht explizit von einer Anwendung subskribiert werden, die das Platzhalterschema MQSO_WILDCARD_TOPIC verwendet. Diese Situation führt dazu, dass die Anwendung mehr Veröffentlichungen abruft als erwartet.

Sie sollten kein Platzhalterzeichen in der Themenzeichenfolge eines definierten Themenobjekts verwenden. Wenn doch, wird das Zeichen bei Verwendung des Objekts durch eine Publisher als Literal und bei Verwendung durch eine Subskription als Platzhalterzeichen behandelt. Dies kann zu Unklarheiten führen.

#### Mustercodeausschnitt

Dieser Codeausschnitt, der dem Beispielprogramm Beispiel 2: Publisher eines variablen Themas entnommen wurde, kombiniert ein Themenobjekt mit einer variablen Themenzeichenfolge:

```
MQOD   td = {MQOD_DEFAULT}; /* Object Descriptor      */
td.ObjectType = MQOT_TOPIC;  /* Object is a topic      */
td.Version = MQOD_VERSION_4;  /* Descriptor needs to be V4  */
strncpy(td.ObjectName, topicName, MQ_TOPIC_NAME_LENGTH);
td.ObjectString.VSPtr = topicString;
td.ObjectString.VSLength = (MQLONG)strlen(topicString);
td.ResObjectString.VSPtr = resTopicStr;
td.ResObjectString.VSBufSize = sizeof(resTopicStr)-1;
MQOPEN(Hconn, &td, MQOO_OUTPUT | MQOO_FAIL_IF_QUIESCING, &Hobj, &CompCode, &Reason);
```

## *Themenstrukturen*

Jedes Thema, das Sie definieren, wird in der Themenstruktur durch ein Element oder einen Knoten dargestellt. Die Themenstruktur kann entweder leer sein, um ganz neu zu beginnen, oder Themen enthalten, die zuvor mithilfe von WebSphere MQ-Scriptbefehlen oder PCF-Befehlen definiert wurden. Sie können ein neues Thema definieren, indem Sie entweder die Befehle zum Erstellen von Themen verwenden oder das Thema zum ersten Mal in einer Veröffentlichung oder Subskription angeben.

Obwohl Sie zum Definieren der Themenzeichenfolge eines Themas eine beliebige Zeichenfolge verwenden können, sollten Sie eine Themenzeichenfolge auswählen, die in eine hierarchische Baumstruktur passt. Ein sorgfältiger Entwurf von Themenzeichenfolgen und Themenstrukturen erleichtert folgende Vorgänge:

- Abonnieren mehrerer Themen

- Erstellen von Sicherheitsrichtlinien

Obwohl Sie eine Themenstruktur auch als flache, lineare Struktur entwerfen können, ist es besser, eine Themenstruktur in einer hierarchische Struktur mit einem oder mehreren Stammthemen zu erstellen. Weitere Informationen zur Sicherheitsplanung und zu Themen finden Sie im Abschnitt Publish/Subscribe-Sicherheit.

In Abbildung 18 auf Seite 83 zeigt ein Beispiel für eine Themenstruktur mit einem einzigen Stammthema.

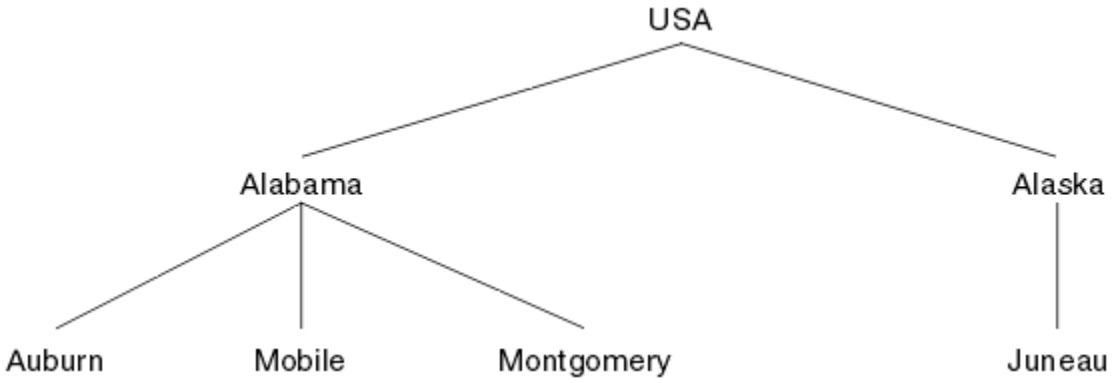



*Abbildung 18. Beispiel für eine Themenstruktur*

Jede Zeichenfolge in der Abbildung steht für einen Knoten in der Themenstruktur. Eine vollständige Themenzeichenfolge entsteht durch die Zusammenfassung von Knoten auf einer oder mehreren Ebenen in der Themenstruktur. Ebenen werden durch das Zeichen "/" getrennt. Eine vollständig angegebenen Themenzeichenfolge hat das Format "Stamm/Ebene2/Ebene3".

Die gültigen Themen in der Themenstruktur in Abbildung 18 auf Seite 83 lauten wie folgt:

"USA"
"USA/Alabama"
"USA/Alaska"
"USA/Alabama/Auburn"
"USA/Alabama/Mobile"
"USA/Alabama/Montgomery"
"USA/Alaska/Juneau"

Beachten Sie beim Entwerfen von Themenzeichenfolgen und Themenstrukturen, dass der Warteschlangenmanager nicht die Themenzeichenfolge selbst interpretiert oder auch nur versucht, eine Bedeutung daraus abzuleiten. Er verwendet die Themenzeichenfolge nur, um ausgewählte Nachrichten an Subskribenten des betreffenden Themas zu senden.

Für die Erstellung und den Inhalt einer Themenstruktur gelten folgende Prinzipien:

- Die Anzahl Ebenen in einer Themenstruktur ist nicht begrenzt.
- Der Länge des Namens einer Ebene in einer Themenstruktur ist nicht begrenzt.
- Es können beliebig viele Stammknoten vorhanden sein, d. h., es kann beliebig viele Themenstrukturen geben.

**Zugehörige Tasks**

Reduzieren der Anzahl unerwünschter Themen in der Themenstruktur

## *Verwaltungsthemenobjekte*

Mit einem Verwaltungsthemenobjekt können Sie Themen bestimmte, nicht standardmäßige Attribute zuweisen.

Abbildung 19 auf Seite 84 zeigt das übergeordnete Thema `Sport` an, unterteilt in mehrere Unterthemen für verschiedene Sportarten und in Form einer Themenstruktur:


*Abbildung 19. Visualisierung einer Themenstruktur*

Abbildung 20 auf Seite 84 zeigt, wie die Themenstruktur weiter unterteilt werden kann, um unterschiedliche Informationsarten für jede Sportart zu trennen:

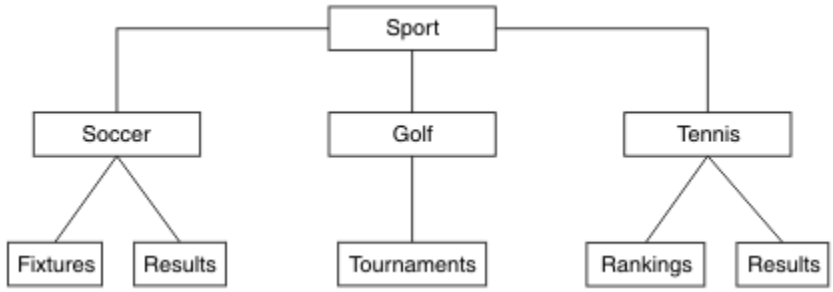



*Abbildung 20. Erweiterte Themenstruktur*

Zur Erstellung der gezeigten Themenstruktur müssen keine Objekte für Verwaltungsthemen definiert werden. Alle Knoten in dieser Baumstruktur sind durch eine Themenzeichenfolge definiert, die in einer Veröffentlichung oder Subskription erstellt wurde. Jedes Thema der Baumstruktur übernimmt seine Attribute von den ihm übergeordneten Elementen. Attribute werden von dem übergeordneten Themenobjekt übernommen, da standardmäßig alle Attribute auf ASPARENT eingestellt sind. In diesem Beispiel besitzt daher jedes Thema die gleichen Attribute wie das Thema `Sport`. Das Thema `Sport` hat kein Verwaltungsthemenobjekt und übernimmt seine Attribute aus SYSTEM.BASE.TOPIC.

Es wird dringend davon abgeraten, Berechtigungen für Benutzer, die nicht der Gruppe mqm angehören, auf Stammknotenebene der Themenstruktur (also auf der Ebene SYSTEM.BASE.TOPIC) zu erteilen, weil die Berechtigungen übernommen werden, aber nicht eingeschränkt werden können. Das heißt, wenn Sie Berechtigungen auf dieser Ebene erteilen, erteilen Sie damit Berechtigungen für die gesamte Struktur. Erteilen Sie die Berechtigung besser auf einer niedrigeren Themenebene in der Hierarchie.

Mit Verwaltungsthemenobjekten können bestimmte Attribute für bestimmte Knoten der Verzeichnisstruktur definiert werden. Im folgenden Beispiel wird das Verwaltungsthemenobjekt definiert, um die Eigenschaft der permanenten Subskription DURSUB des Themas 'Fußball' auf den Wert NO zu setzen:

```
DEFINE TOPIC(FOOTBALL.EUROPEAN)
TOPICSTR('Sport/Soccer')
DURSUB(NO)
DESCR('Administrative topic object to disallow durable subscriptions')
```

Die Verzeichnisstruktur kann nun visuell dargestellt werden als:

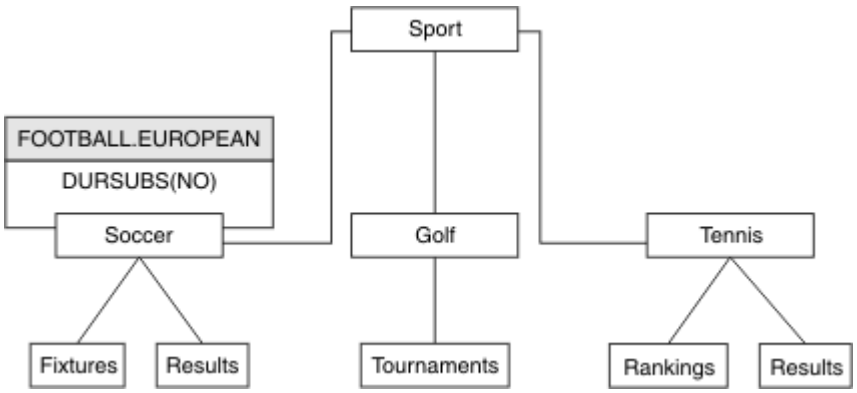



*Abbildung 21. Visualisierung eines dem Thema 'Sport/Fußball' zugeordneten Verwaltungsthemenobjekts*

Haben Anwendungen Themen abonniert, die in der Verzeichnisstruktur dem Thema 'Fußball' untergeordnet sind, können sie auch weiterhin die vor dem Hinzufügen des Verwaltungsthemenobjekts gültige Themenzeichenfolge verwenden. Eine Anwendung kann jetzt jedoch so geschrieben werden, dass sie den Objektnamen `FOOTBALL.EUROPEAN`anstelle der Zeichenfolge `/Sport/Soccer`subskribiert. Um beispielsweise `/Sport/Soccer/Results`zu subskribieren, kann eine Anwendung `MQSD.ObjectName` als `FOOTBALL.EUROPEAN` und `MQSD.ObjectString` als `Results`angeben.

Auf diese Weise können Sie einen Teil der Verzeichnisstruktur vor den Anwendungsentwicklern verbergen. Wenn Sie ein Verwaltungsthemenobjekt in einem bestimmten Knoten der Verzeichnisstruktur definieren, können Anwendungsentwickler somit ihre eigenen Themen diesem unterordnen. Die Entwickler müssen dabei das übergeordnete Thema, jedoch keine weiteren Knoten in der übergeordneten Baumstruktur kennen.

## Attribute übernehmen

Sind in einer Themenstruktur viele Verwaltungsthemenobjekte vorhanden, übernimmt jedes Verwaltungsthemenobjekt standardmäßig die Attribute des ihm am nächsten übergeordneten Verwaltungsthemas. Das vorhergehende Beispiel wurde in Abbildung 22 auf Seite 85 erweitert:

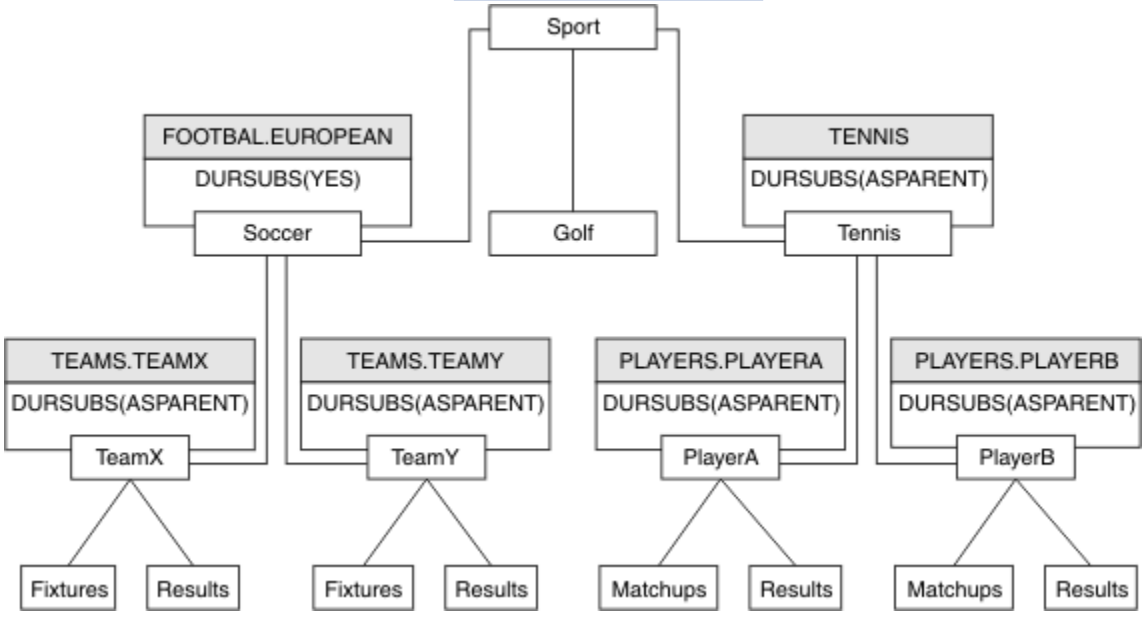



*Abbildung 22. Themenstruktur mit mehreren Verwaltungsthemenobjekten*

Verwenden Sie beispielsweise die Übernahme, um allen untergeordneten Themen von `/Sport/Soccer` die Eigenschaft zu geben, dass Subskriptionen nicht permanent sind. Ändern Sie das Attribut `DURSUB` von `FOOTBALL.EUROPEAN` in `NO`.

Dieses Attribut können Sie mit folgendem Befehl festlegen:

```
ALTER TOPIC(FOOTBALL.EUROPEAN) DURSUB(NO)
```

Für alle Verwaltungsthemenobjekte untergeordneter Themen von `Sport/Soccer` ist die Eigenschaft DURSUB auf den Standardwert ASPARENTgesetzt. Nachdem Sie den DURSUB -Eigenschaftswert von `FOOTBALL.EUROPEAN` in `NO`geändert haben, übernehmen die untergeordneten Themen von `Sport/Soccer` den DURSUB -Eigenschaftswert `NO`. Alle untergeordneten Themen von `Sport/Tennis` übernehmen den Wert von DURSUB aus dem Objekt `SYSTEM.BASE.TOPIC` . `SYSTEM.BASE.TOPIC` hat den Wert YES.

Der Versuch, eine permanente Subskription für das Thema `Sport/Soccer/TeamX/Results` zu erstellen, schlägt jetzt fehl. Der Versuch, eine permanente Subskription für `Sport/Tennis/PlayerB/Results` zu erstellen, wäre jedoch erfolgreich.

## Verwendung von Platzhaltern mit der Eigenschaft WILDCARD steuern

Mit der MQSC-Eigenschaft **`Topic`** WILDCARD oder der entsprechenden PCF-Eigenschaft Topic `WildcardOperation` können Sie die Zustellung von Veröffentlichungen an Subskribentenanwendungen steuern, die Namen von Platzhalterthemenzeichenfolgen verwenden. Die Eigenschaft WILDCARD kann einen von zwei möglichen Werten haben:

**WILDCARD**
Aktionen von Subskriptionen mit Platzhaltern bezüglich dieses Themas.

**PASSTHRU**
Subskriptionen für ein Thema mit Platzhalter, das weniger spezifisch ist als die Themenzeichenfolge für dieses Themenobjekt, empfangen Veröffentlichungen zu diesem Thema und zu spezifischeren Themenzeichenfolge.

**BLOCK**
Subskriptionen für ein Thema mit Platzhalter, das weniger spezifisch ist als die Themenzeichenfolge für dieses Themenobjekt, empfangen keine Veröffentlichungen zu diesem Thema und zu spezifischeren Themenzeichenfolge.

Der Wert für dieses Attribut wird bei der Definition von Subskriptionen verwendet. Wenn Sie dieses Attribut ändern, ist die Gruppe von Themen, die bereits durch vorhandene Subskriptionen abgedeckt sind, nicht durch die Änderung betroffen. Dieses Szenario gilt auch, wenn sich durch die Erstellung oder das Löschen von Themenobjekten die Topologie ändert; die Themen mit Subskriptionen, die nach der Änderung des Attributs WILDCARD erstellt wurden, werden mit der geänderten Topologie erstellt. Wenn die Themen mit den vorhandenen Subskriptionen übereinstimmen sollen, müssen Sie den Warteschlangenmanager neu starten.

Führen Sie die Schritte im Beispiel „Beispiel: Publish/Subscribe-Cluster Sport erstellen” auf Seite 90 aus, um die in Abbildung 23 auf Seite 87 gezeigte Themenstruktur zu erstellen.

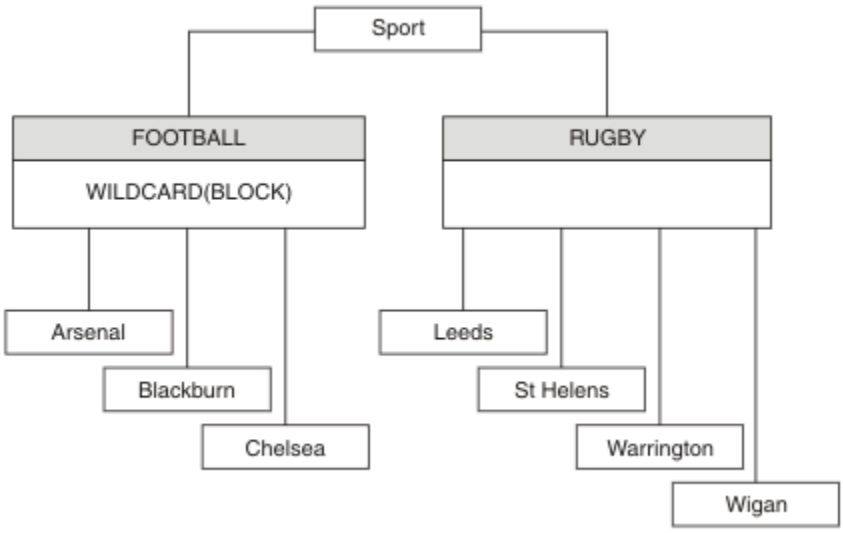



*Abbildung 23. Themenstruktur, die die WILDCARD-Eigenschaft BLOCK verwendet*

Ein Subskribent, der die Platzhalter-Themenzeichenfolge # verwendet, empfängt alle Veröffentlichungen für das Thema `Sport` und die Unterverzeichnisstruktur `Sport/Rugby` Der Subskribent empfängt keine Veröffentlichungen für die untergeordnete Baumstruktur `Sport/Football` , weil der Wert der Eigenschaft `WILDCARD` des Themas `Sport/Football` BLOCKist.

Die Standardeinstellung lautet PASSTHRU. Sie können den Wert PASSTHRU der Eigenschaft `WILDCARD` auf Knoten in der Baumstruktur `Sport` festlegen. Wenn die Knoten nicht über den Eigenschaftswert `WILDCARD` verfügen BLOCK, ändert die Einstellung PASSTHRU nicht das Verhalten, das von Subskribenten für Knoten in der `Sports` -Baumstruktur beobachtet wird.

Erstellen Sie im Beispiel Subskriptionen, um zu sehen, wie sich die Platzhaltereinstellung auf die bereitgestellten Veröffentlichungen auswirkt (siehe dazu Abbildung 27 auf Seite 92). Führen Sie den Publish-Befehl in Abbildung 30 auf Seite 93 aus, um einige Veröffentlichungen zu erstellen.

```
pub QMA
```

*Abbildung 24. Für QMA publizieren*

Die Ergebnisse werden in Tabelle 3 auf Seite 87 gezeigt. Beachten Sie, wie die Einstellung des `WILDCARD`-Eigenschaftswerts BLOCK verhindert, dass Subskriptionen mit Platzhaltern Veröffentlichungen für Themen empfangen, die innerhalb des Platzhalterumfangs liegen.

*Tabelle 3. Auf QMA empfangene Veröffentlichungen*

| **Abonnement** | **Themenzeichenfolge** | **Empfangene Veröffentlichungen** | **Anmerkungen** |
|---|---|---|---|
| SPORTS | `Sports/#` | `Sports`<br>`Sports/Rugby`<br>`Sports/Rugby/Leeds` | Alle Veröffentlichungen zur Unterverzeichnisstruktur 'Football', die von `WILDCARD(BLOCK)` auf `Sports/Football` blockiert werden |
| SARSENAL | `Sports/#/Arsenal` | - | `WILDCARD(BLOCK)` on `Sports/Football` verhindert Platzhaltersubskription unter `Arsenal` |
| SLEEDS | `Sports/#/Leeds` | `Sports/Rugby/Leeds` | Standardwert `WILDCARD` für 'Sport/Rugby' verhindert keine Platzhaltersubskription auf `Leeds`. |

**Anmerkung:**

Angenommen, eine Subskription hat einen Platzhalter, der einem Themenobjekt mit dem WILDCARD-Eigenschaftswert BLOCK entspricht. Wenn die Subskription auch eine Themenzeichenfolge rechts neben dem entsprechenden Platzhalter hat, empfängt die Subskription niemals Veröffentlichungen. Die nicht geblockten Veröffentlichungen sind Veröffentlichungen für Themen, die dem geblockten Platzhalter übergeordnet sind. Veröffentlichungen für Themen, die dem Thema mit dem Eigenschaftswert BLOCK untergeordnet sind, werden vom Platzhalter geblockt. Subskriptionsthemen-Zeichenfolgen, die ein Thema rechts neben dem Platzhalter einschließen, empfangen daher niemals passende Veröffentlichungen.

Wenn der Eigenschaftswert WILDCARD auf BLOCK gesetzt wird, ist keine Subskription mithilfe einer Themenzeichenfolge möglich, die Platzhalter beinhaltet. Eine solche Subskription ist normal. Die Subskription hat ein explizites Thema, das dem Thema mit einem Themenobjekt entspricht, das den WILDCARD-Eigenschaftswert BLOCK hat. Sie verwendet Platzhalter für Themen, die dem WILDCARD-Eigenschaftswert BLOCK über- oder untergeordnet sind. Im Beispiel in Abbildung 23 auf Seite 87kann eine Subskription wie Sports/Football/# Veröffentlichungen empfangen.

## Platzhalter und Cluster-Topics

Definitionen von Cluster-Topics werden an jeden Warteschlangenmanager in einem Cluster weitergegeben. Die Subskription eines Cluster-Topics auf einem Warteschlangenmanager in einem Cluster führt dazu, dass der Warteschlangenmanager Proxy-Subskriptionen erstellt. Eine Proxy-Subskription wird in jedem Warteschlangenmanager im Cluster erstellt. Subskriptionen, die Themenzeichenfolgen mit Platzhaltern verwenden, kombiniert mit Cluster-Topics, lassen das Verhalten schwer vorhersagen. Das Verhalten wird im folgenden Beispiel erläutert.

In dem für das Beispiel verwendeten Cluster „Beispiel: Publish/Subscribe-Cluster Sport erstellen” auf Seite 90 hat QMB dieselben Subskriptionen wie QMA. QMB hat jedoch keine Veröffentlichungen empfangen, nachdem der Publisher an QMA veröffentlicht hat (siehe Abbildung 24 auf Seite 87). Obwohl die Themen Sports/Footbal und Sports/Rugby Clusterthemen sind, verweisen die in fullsubs.tst definierten Subskriptionen nicht auf ein Clusterthema. Es werden keine Proxy-Subskriptionen von QMB an QMA weitergegeben. Ohne Proxy-Subskriptionen werden keine Veröffentlichungen für QMA an QMB weitergeleitet.

Einige Subskriptionen, wie z. B. Sports/#/Leeds, scheinen möglicherweise auf ein Clusterthema zu verweisen, in diesem Fall Sports/Rugby. Die Sports/#/Leeds -Subskription wird tatsächlich in das Themenobjekt SYSTEM.BASE.TOPICaufgelöst.

Die Regel zum Auflösen des Themenobjekts, das von einer Subskription wie Sports/#/Leeds referenziert wird, lautet wie folgt. Schneiden Sie die Themenzeichenfolge bis zum ersten Platzhalter ab. Suchen Sie links über die Themenzeichenfolge nach dem ersten Thema mit einem zugehörigen Verwaltungsthemenobjekt. Das Themenobjekt kann einen Clusternamen angeben oder ein lokales Themenobjekt definieren. Im Beispiel Sports/#/Leedsist die Themenzeichenfolge nach dem Abschneiden Sports, die kein Themenobjekt hat, und Sports/#/Leeds übernimmt daher von SYSTEM.BASE.TOPIC, einem lokalen Themenobjekt.

Um zu sehen, wie die Subskription von Cluster-Topics die Weitergabe von Platzhaltern ändern kann, führen Sie das Stapelscript upsubs.bat aus. Das Script löscht die Subskriptionswarteschlangen und fügt die Subskriptionen des Cluster-Topics zu fullsubs.tst hinzu. Führen Sie puba.bat erneut aus, um einen Stapel von Veröffentlichungen zu erstellen (siehe Abbildung 24 auf Seite 87).

Tabelle 4 auf Seite 89 zeigt das Ergebnis, nachdem zwei neue Subskriptionen dem Warteschlangenmanager hinzugefügt wurden, in dem die Veröffentlichungen veröffentlicht wurden. Wie erwartet, empfangen die neuen Subskriptionen je eine Veröffentlichung, und die Anzahl der von den anderen Subskriptionen empfangenen Veröffentlichungen ist unverändert. Die unerwarteten Resultate treten im anderen Cluster-Warteschlangenmanager auf, siehe Tabelle 5 auf Seite 89.

| *Tabelle 4. Auf QMA empfangene Veröffentlichungen* | | | |
|---|---|---|---|
| **Abonnement** | **Themenzeichenfolge** | **Empfangene Veröffentlichungen** | **Anmerkungen** |
| SPORTS | Sports/# | Sports<br>Sports/Rugby<br>Sports/Rugby/Leeds | Alle Veröffentlichungen zur Unterverzeichnisstruktur 'Football', die von WILDCARD(BLOCK) auf Sports/Football blockiert werden |
| SARSENAL | Sports/#/Arsenal | - | WILDCARD(BLOCK) on Sports/Football verhindert Platzhaltersubskription unter Arsenal |
| SLEEDS | Sports/#/Leeds | Sports/Rugby/Leeds | Standardwert WILDCARD für 'Sport/Rugby' verhindert keine Platzhaltersubskription auf Leeds. |
| FARSENAL | Sports/Football/Arsenal | Sports/Football/Arsenal | Arsenal empfängt eine Veröffentlichung, da die Subskription keinen Platzhalter hat. |
| FLEEDS | Sports/Rugby/Leeds | Sports/Rugby/Leeds | Leeds würde für jedes Ereignis eine Veröffentlichung empfangen. |

Tabelle 5 auf Seite 89 zeigt die Resultate, nachdem die beiden neuen Subskriptionen zu QMB hinzugefügt und auf QMA veröffentlicht wurden. QMB empfing keine Veröffentlichungen ohne diese beiden neuen Subskriptionen. Wie erwartet empfangen die beiden neuen Subskriptionen Veröffentlichungen, da Sports/FootBall und Sports/Rugby Clusterthemen sind. QMB weitergeleitete Proxy-Subskriptionen für Sports/Football/Arsenal und Sports/Rugby/Leeds an QMA, die dann die Veröffentlichungen an QMBgesendet haben

Das unerwartete Ergebnis ist, dass die beiden Subskriptionen Sports/# und Sports/#/Leeds , die zuvor keine Veröffentlichungen empfangen haben, jetzt Veröffentlichungen empfangen. Dies liegt daran, dass die Sports/Football/Arsenal -und Sports/Rugby/Leeds -Veröffentlichungen, die für die anderen Subskriptionen an QMB weitergeleitet werden, jetzt für jeden Subskribenten verfügbar sind, der QMBzugeordnet ist. Folglich empfangen die Subskriptionen für die lokalen Themen Sports/# und Sports/#/Leeds die Veröffentlichung Sports/Rugby/Leeds . Sports/#/Arsenal erhält weiterhin keine Veröffentlichung, weil für Sport/Fußball die Eigenschaft WILDCARD auf BLOCKgesetzt ist.

| *Tabelle 5. Auf QMB empfangene Veröffentlichungen* | | | |
|---|---|---|---|
| **Abonnement** | **Themenzeichenfolge** | **Empfangene Veröffentlichungen** | **Anmerkungen** |
| *SPORTS* | *Sports/#* | *Sports/Rugby/Leeds* | *All publications to Football subtree blocked by WILDCARD(BLOCK) on Sports/Football* |
| SARSENAL | Sports/#/Arsenal | - | WILDCARD(BLOCK) on Sports/Football verhindert Platzhaltersubskription unter Arsenal |
| *SLEEDS* | *Sports/#/Leeds* | *Sports/Rugby/Leeds* | *Der Standardwert WILDCARD für Sport/Rugby verhindert nicht das Platzhalterabonnement für Leeds.* |
| FARSENAL | Sports/Football/Arsenal | Sports/Football/Arsenal | Arsenal empfängt eine Veröffentlichung, da die Subskription keinen Platzhalter hat. |

| *Tabelle 5. Auf QMB empfangene Veröffentlichungen (Forts.)* | | | |
|---|---|---|---|
| **Abonnement** | **Themenzeichenfolge** | **Empfangene Veröffentlichungen** | **Anmerkungen** |
| FLEEDS | Sports/Rugby/Leeds | Sports/Rugby/Leeds | Leeds würde für jedes Ereignis eine Veröffentlichung empfangen. |

Bei den meisten Anwendungen ist es nicht erwünscht, dass eine Subskription das Verhalten einer anderen Subskription beeinflusst. Eine wichtige Verwendung der Eigenschaft WILDCARD mit dem Wert BLOCK ist es, dass sich die Subskriptionen für dieselbe Themenzeichenfolge mit Platzhalterzeichen einheitlich verhalten. Unabhängig davon, ob sich die Subskription in demselben Warteschlangenmanager wie der Publisher oder in einem anderen Warteschlangenmanager befindet, sind die Resultate der Subskription dieselben.

## Platzhalter und Datenströme

Wenn eine neue Anwendung für die Publish/Subscribe-API geschrieben wird, empfängt eine Subskription von * keine Veröffentlichungen. Um alle Sportveröffentlichungen zu erhalten, müssen Sie Sports/*oder Sports/#und ähnlich für Business -Veröffentlichungen subskribieren.

Das Verhalten einer vorhandenen Publish/Subscribe-Anwendung in der Warteschlange ändert sich nicht, wenn der Publish/Subscribe-Broker auf eine höhere Version von IBM MQmigriert wird. Die Eigenschaft **StreamName** in den Befehlen **Publish**, **Register Publisher** oder **Subscriber** wird dem Namen des Themas zugeordnet, auf das der Datenstrom migriert wurde.

## Platzhalter und Subskriptionspunkte

Wenn eine neue Anwendung für die Publish/Subscribe-API geschrieben wird, empfängt eine Subskription von * keine Veröffentlichungen. Um alle Sportveröffentlichungen zu erhalten, müssen Sie Sports/*oder Sports/#und ähnlich für Business -Veröffentlichungen subskribieren.

Das Verhalten einer vorhandenen Publish/Subscribe-Anwendung in der Warteschlange ändert sich nicht, wenn der Publish/Subscribe-Broker auf eine höhere Version von IBM MQmigriert wird. Die Eigenschaft **SubPoint** in den Befehlen **Publish**, **Register Publisher** oder **Subscriber** wird dem Namen des Themas zugeordnet, auf das die Subskription migriert wurde.

### Beispiel: Publish/Subscribe-Cluster Sport erstellen

In den nachfolgenden Schritten wird ein Cluster CL1 mit vier Warteschlangenmanagern erstellt: zwei vollständige Repositorys, CL1A und CL1B, und zwei Teilrepositorys, QMA und QMB. Die vollständigen Repositorys enthalten nur Clusterdefinitionen. QMA wird als Cluster-Topic-Host festgelegt. Permanente Subskriptionen sind in QMA und QMB definiert.

**Anmerkung:** Das Beispiel ist für Windows codiert. Wenn Sie das Beispiel auf anderen Plattformen konfigurieren und testen möchten, müssen Sie den Programmcode für Create qmgrs.bat und create pub.bat ändern.

1. Erstellen Sie die Scriptdateien.
   a. Create topics.tst
   b. Create wildsubs.tst
   c. Create fullsubs.tst
   d. Create qmgrs.bat
   e. create pub.bat
2. Führen Sie Create qmgrs.bat aus, um die Konfiguration zu erstellen.

```
qmgrs
```

Erstellen Sie die Themen in Abbildung 23 auf Seite 87. Das Script in Abbildung 5 erstellt die Clusterthemen `Sports/Football` und `Sports/Rugby`.

**Anmerkung:** Die Option REPLACE ersetzt nicht die TOPICSTR-Eigenschaften eines Themas. TOPICSTR ist eine Eigenschaft, die im Beispiel gezielt variiert wird, um verschiedene Themenstrukturen zu testen. Wenn Sie ein Thema ändern möchten, müssen Sie das Thema zuerst löschen.

```
DELETE TOPIC ('Sports')
DELETE TOPIC ('Football')
DELETE TOPIC ('Arsenal')
DELETE TOPIC ('Blackburn')
DELETE TOPIC ('Chelsea')
DELETE TOPIC ('Rugby')
DELETE TOPIC ('Leeds')
DELETE TOPIC ('Wigan')
DELETE TOPIC ('Warrington')
DELETE TOPIC ('St. Helens')

DEFINE TOPIC ('Sports')   TOPICSTR('Sports')
DEFINE TOPIC ('Football')  TOPICSTR('Sports/Football') CLUSTER(CL1) WILDCARD(BLOCK)
DEFINE TOPIC ('Arsenal')  TOPICSTR('Sports/Football/Arsenal')
DEFINE TOPIC ('Blackburn') TOPICSTR('Sports/Football/Blackburn')
DEFINE TOPIC ('Chelsea')  TOPICSTR('Sports/Football/Chelsea')
DEFINE TOPIC ('Rugby')   TOPICSTR('Sports/Rugby')  CLUSTER(CL1)
DEFINE TOPIC ('Leeds')   TOPICSTR('Sports/Rugby/Leeds')
DEFINE TOPIC ('Wigan')   TOPICSTR('Sports/Rugby/Wigan')
DEFINE TOPIC ('Warrington') TOPICSTR('Sports/Rugby/Warrington')
DEFINE TOPIC ('St. Helens') TOPICSTR('Sports/Rugby/St. Helens')
```

*Abbildung 25. Themen löschen und erstellen: topics.tst*

**Anmerkung:** Löschen Sie die Themen, da REPLACE keine Themenzeichenfolgen ersetzt.

Erstellen Sie Subskriptionen mit Platzhaltern. Die Platzhalter entsprechen den Themen mit Themenobjekten in Abbildung 23 auf Seite 87. Erstellen Sie für jede Subskription eine Warteschlange. Beim Ausführen bzw. erneuten Ausführen des Scripts werden die Warteschlangen geleert, und die Subskriptionen werden gelöscht.

**Anmerkung:** Die Option REPLACE ersetzt nicht die die Eigenschaften TOPICOBJ oder TOPICSTR einer Subskription. TOPICOBJ oder TOPICSTR sind die Eigenschaften, die im Beispiel sinnvoll variiert werden, um verschiedene Subskriptionen zu testen. Um sie zu ändern, löschen Sie zunächst die Subskription.

```
DEFINE QLOCAL(QSPORTS)  REPLACE
DEFINE QLOCAL(QSARSENAL) REPLACE
DEFINE QLOCAL(QSLEEDS)  REPLACE
CLEAR QLOCAL(QSPORTS)
CLEAR QLOCAL(QSARSENAL)
CLEAR QLOCAL(QSLEEDS)

DELETE SUB  (SPORTS)
DELETE SUB  (SARSENAL)
DELETE SUB  (SLEEDS)
DEFINE SUB  (SPORTS)  TOPICSTR('Sports/#')     DEST(QSPORTS)
DEFINE SUB  (SARSENAL) TOPICSTR('Sports/+/Arsenal') DEST(QSARSENAL)
DEFINE SUB  (SLEEDS)  TOPICSTR('Sports/+/Leeds')  DEST(QSLEEDS)
```

*Abbildung 26. Platzhaltersubskriptionen erstellen: wildsubs.tst*

Erstellen Sie Subskriptionen, die die Cluster-Topic-Objekte referenzieren.

**Anmerkung:**

Das Begrenzungszeichen / wird automatisch zwischen die von TOPICOBJ referenzierte Themenzeichenfolge und die von TOPICSTR definierte Themenzeichenfolge eingefügt.

Die Definition `DEFINE SUB(FARSENAL) TOPICSTR('Sports/Football/Arsenal') DEST(QFARSENAL)` erstellt dieselbe Subskription. Die Verwendung von TOPICOBJ ist eine zeiteffiziente Methode, bereits definierte Themenzeichenfolgen zu referenzieren. Die erstellte Subskription referenziert das Themenobjekt nicht länger.

```
DEFINE QLOCAL(QFARSENAL) REPLACE
DEFINE QLOCAL(QRLEEDS)  REPLACE
CLEAR QLOCAL(QFARSENAL)
CLEAR QLOCAL(QRLEEDS)

DELETE SUB  (FARSENAL)
DELETE SUB  (RLEEDS)
DEFINE SUB  (FARSENAL) TOPICOBJ('Football') TOPICSTR('Arsenal') DEST(QFARSENAL)
DEFINE SUB  (RLEEDS)  TOPICOBJ('Rugby')  TOPICSTR('Leeds')  DEST(QRLEEDS)
```

*Abbildung 27. Subskriptionen löschen und erstellen: fullsubs.tst*

Erstellen Sie einen Cluster mit zwei Repositorys. Erstellen Sie zwei Teilrepositorys zum Veröffentlichen und Abonnieren. Führen Sie das Script erneut aus, um alles zu löschen und neu zu starten. Das Script erstellt zudem die Themenhierarchie und die ursprünglichen Platzhaltersubskriptionen.

**Anmerkung:**

Schreiben Sie auf anderen Plattformen ein ähnliches Script, oder geben Sie alle Befehle ein. Mit einem Script lässt sich alles schneller löschen und mit einer identischen Konfiguration erneut starten.

```
@echo off
set port.CL1B=1421
set port.CLIA=1420
for %%A in (CL1A CL1B QMA QMB) do call :createQM %%A
call :configureQM CL1A CL1B %port.CL1B% full
call :configureQM CL1B CL1A %port.CL1A% full
for %%A in (QMA QMB) do call :configureQM %%A CL1A %port.CL1A% partial
for %%A in (topics.tst wildsubs.tst) do runmqsc QMA < %%A
for %%A in (wildsubs.tst) do runmqsc QMB < %%A
goto:eof

:createQM
echo Configure Queue manager %1
endmqm -p %1
for %%B in (dlt crt str) do %%Bmqm %1
goto:eof

:configureQM
if %1==CL1A set p=1420
if %1==CL1B set p=1421
if %1==QMA set p=1422
if %1==QMB set p=1423
echo configure %1 on port %p% connected to repository %2 on port %3 as %4 repository
echo DEFINE LISTENER(LST%1) TRPTYPE(TCP) PORT(%p%) CONTROL(QMGR) REPLACE | runmqsc %1
echo START LISTENER(LST%1) | runmqsc %1
if full==%4 echo ALTER QMGR REPOS(CL1) DEADQ(SYSTEM.DEAD.LETTER.QUEUE) | runmqsc %1
echo DEFINE CHANNEL(TO.%2) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('LOCALHOST(%3)') CLUSTER(CL1)
REPLACE | runmqsc %1
echo DEFINE CHANNEL(TO.%1) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('LOCALHOST(%p%)') CLUS
TER(CL1) REPLACE | runmqsc %1
goto:eof
```

*Abbildung 28. Warteschlangenmanager erstellen: qmgrs.bat*

Aktualisieren Sie die Konfiguration durch Hinzufügen der Subskriptionen zu den Cluster-Topics.

```
@echo off
for %%A in (QMA QMB) do runmqsc %%A < wildsubs.tst
for %%A in (QMA QMB) do runmqsc %%A < upsubs.tst
```

*Abbildung 29. Subskriptionen aktualisieren: upsubs.bat*

Führen Sie `pub.bat` mit einem Warteschlangenmanager als Parameter aus, um Nachrichten zu veröffentlichen, die die Veröffentlichungsthema-Zeichenfolge enthalten. `Pub.bat` verwendet das Beispielprogramm **amqspub**.

```
@echo off
@rem Provide queue manager name as a parameter
set S=Sports
set S=6 Sports/Football Sports/Football/Arsenal
set S=6 Sports/Rugby Sports/Rugby/Leeds
for %%B in (6) do echo %%B | amqspub %%B %1
```

*Abbildung 30. Veröffentlichen: pub.bat*

### *Datenströme und Themen*

Das eingereihte Publish/Subscribe basiert auf dem Konzept eines Veröffentlichungsdatenstroms, das es im Modell für integriertes Publish/Subscribe nicht gibt. Im eingereihten Publish/Subscribe bieten Datenströme die Möglichkeit, den Informationsfluss für unterschiedliche Themen zu trennen. Ein Datenstrom wird als Thema der höchsten Ebene implementiert, das administrativ einer anderen Themenkennung zugeordnet werden kann.

Der Standarddatenstrom SYSTEM.BROKER.DEFAULT.STREAM wird für alle Broker und Warteschlangenmanager in einem Netz automatisch eingerichtet und es ist keine zusätzliche Konfiguration erforderlich, um den Standarddatenstrom verwenden zu können. Sie können sich den Standarddatenstrom als nicht benannten Themenspeicherbereich vorstellen. Themen, die im Standarddatenstrom veröffentlicht werden, sind sofort für alle verbundenen Warteschlangenmanager mit aktiviertem eingereihtem Publish/Subscribe verfügbar. Benannte Datenströme gleichen separaten, benannten Themenspeicherbereichen. Der benannte Datenstrom muss auf jedem Broker definiert werden, auf dem er verwendet wird.

Wenn sich die Publisher und Subskribenten auf verschiedenen Warteschlangenmanagern befinden, ist für den Austausch der Veröffentlichungen und Subskriptionen zwischen den Brokern keine weitere Konfiguration erforderlich, sobald die Broker in derselben Brokerhierarchie miteinander verbunden sind. Diese Interoperabilität funktioniert auch umgekehrt.

## Benannte Datenströme

Ein Lösungsentwickler, der mit dem Programmiermodell für eingereihtes Publish/Subscribe arbeitet, entscheidet sich möglicherweise dafür, alle Sportveröffentlichungen in einen benannten Datenstrom mit dem Namen `Sport` zu stellen. Damit der Datenstrom für einen Warteschlangenmanager verfügbar ist, der unter IBM MQ mit aktiviertem eingereihtem Publish/Subscribe ausgeführt wird, muss der Datenstrom manuell hinzugefügt werden.

Eingereihte Publish/Subscribe-Anwendungen, die `Soccer/Results` im Datenstrom `Sport` subskribieren, funktionieren unverändert. Integrierte Publish/Subscribe-Anwendungen, die das Thema `Sport` mit `MQSUB`subskribieren und die Themenzeichenfolge `Soccer/Results` angeben, empfangen dieselben Veröffentlichungen ebenfalls.

Die Aufgabe zum Hinzufügen eines Datenstroms wird im Abschnitt Datenstrom hinzufügen beschrieben. Sie müssen Datenströme eventuell aus zwei Gründen manuell hinzufügen.

1. Sie entwickeln weiterhin Ihre Anwendungen mit eingereihtem Publish/Subscribe, die auf Warteschlangenmanagern einer höheren Version ausgeführt werden, statt die Anwendungen auf die MQI-Schnittstelle für integriertes Publish/Subscribe zu migrieren.
2. Die Standardzuordnung von Datenströmen zu Themen führt zu einer "Kollision" im Themenspeicherbereich und Veröffentlichungen in einem Datenstrom verfügen über dieselbe Themenzeichenfolge wie Veröffentlichungen aus anderen Quellen.

## Berechtigungen

Standardmäßig befinden sich im Stammelement der Themenstruktur mehrere Themenobjekte: SYSTEM.BASE.TOPIC, SYSTEM.BROKER.DEFAULT.STREAM und SYSTEM.BROKER.DEFAULT.SUBPOINT. Berechtigungen (z. B. für die Veröffentlichung oder Subskription) werden von den Behörden auf dem SYSTEM.BASE.TOPIC festgelegt; Alle Berechtigungen unter SYSTEM.BROKER.DEFAULT.STREAM oder SYSTEM.BROKER.DEFAULT.SUBPOINT werden ignoriert. Wenn SYSTEM.BROKER.DEFAULT.STREAM oder SYSTEM.BROKER.DEFAULT.SUBPOINT gelöscht und mit einer nicht leeren Themenzeichenfolge neu

erstellt werden, werden die für diese Objekte definierten Berechtigungen wie ein normales Themenobjekt verwendet.

## Zuordnung von Datenströmen zu Themen

Ein Datenstrom für eingereihtes Publish/Subscribe wird in IBM MQ nachgeahmt, indem eine Warteschlange erstellt und ihr der Name des Datenstroms zugeordnet wird. Manchmal wird die Warteschlange auch Datenstromwarteschlange genannt, weil sie von Anwendungen mit eingereihtem Publish/Subscribe so wahrgenommen wird. Die Warteschlange wird für die Publish/Subscribe-Steuerkomponente identifiziert, indem sie zu der speziellen Namensliste namens `SYSTEM.QPUBSUB.QUEUE.NAMELIST` hinzugefügt wird. Sie können so viele Datenströme wie nötig hinzufügen, indem Sie der Namensliste weitere spezielle Warteschlangen hinzufügen. Schließlich müssen Sie Themen hinzufügen, die dieselben Namen wie die Datenströme und dieselben Themenzeichenfolgen wie der Datenstromname aufweisen, damit Sie Inhalte in den Themen veröffentlichen und abonnieren können.

Unter außergewöhnlichen Umständen können Sie für die Themen, die den Datenströmen entsprechen, jedoch beliebige Themenzeichenfolgen angeben, die Sie bei der Auswahl der Themen definieren. Zweck der Themenzeichenfolge ist es, dem Thema einen eindeutigen Namen im Themenspeicherbereich zu geben. Normalerweise erfüllt der Datenstromname diesen Zweck perfekt. Manchmal kollidieren ein Datenstromname und ein vorhandener Themenname. Wählen Sie zur Lösung des Problems eine andere Themenzeichenfolge für das dem Datenstrom zugeordnete Thema aus. Stellen Sie bei der Auswahl einer Themenzeichenfolge sicher, dass sie eindeutig ist.

Die Themenzeichenfolge, die in der Themendefinition definiert ist, ist wie üblich im Vergleich zu der Themenzeichenfolge, die von Publishern und Subskribenten über die MQI-Aufrufe `MQOPEN` oder `MQSUB` bereitgestellt werden, mit einem Präfix versehen. Anwendungen, die auf Themen verweisen, die Themenobjekte verwenden, sind von der Auswahl der Präfixthemenzeichenfolge nicht betroffen. Deshalb können Sie eine beliebige Themenzeichenfolge auswählen, sofern die Veröffentlichungen im Themenspeicherbereich eindeutig bleiben.

Für die Neuzuordnung von unterschiedlichen Datenströmen zu anderen Themen ist es wichtig, dass die Präfixe, die für die Themenzeichenfolgen verwendet werden, eindeutig sind, damit eine Themengruppe komplett von einer anderen Themengruppe getrennt werden kann. Sie müssen eine allgemeine Benennungskonvention für Themen definieren, die strikt befolgt werden muss, damit die Zuordnung funktioniert.

In IBM MQverwenden Sie den Vorfixierungsmechanismus, um eine Topiczeichenfolge einem anderen Bereich im Topicbereich zuzuordnen.

**Anmerkung:** Wenn Sie einen Datenstrom löschen, löschen Sie zuerst alle Subskriptionen im Datenstrom. Diese Aktion ist dann besonders wichtig, wenn Subskriptionen von anderen Brokern in der Brokerhierarchie stammen.

### *Subskriptionspunkte und Themen*

Benannte Subskriptionspunkte werden durch Themen und Themenobjekte emuliert.

Informationen zum manuellen Hinzufügen von Subskriptionspunkten finden Sie im Abschnitt Subskriptionspunkt hinzufügen.

## Subskriptionspunkte in IBM MQ

IBM MQ ordnet Subskriptionspunkte verschiedenen Themenspeicherbereichen in der IBM MQ-Themenstruktur zu. Themen in Befehlsnachrichten ohne Subskriptionspunkt werden unverändert dem Stammelement der IBM MQ-Themenstruktur zugeordnet und übernehmen Eigenschaften von `SYSTEM.BASE.TOPIC`.

Befehlsnachrichten mit einem Subskriptionspunkt werden mithilfe der Liste der Themenobjekte in `SYSTEM.QPUBSUB.SUBPOINT.NAMELIST` verarbeitet. Der Name des Subskriptionspunktes in der Befehlsnachricht wird für jedes Themenobjekt in der Liste mit der Themenzeichenfolge abgeglichen. Wenn eine Übereinstimmung gefunden wird, wird der Name des Subskriptionspunktes als Themenknoten der

Themenzeichenfolge vorangestellt. Das Thema übernimmt die Eigenschaften von dem in SYSTEM.QPUBSUB.SUBPOINT.NAMELIST gefundenen zugehörigen Themenobjekt.

Die Verwendung von Subskriptionspunkten führt dazu, dass für jeden Subskriptionspunkt ein separater Themenspeicherbereich erstellt wird. Das Stammelement des Themenspeicherbereichs ist ein Thema, das denselben Namen wie der Subskriptionspunkt hat. Themen in den einzelnen Themenspeicherbereichen übernehmen die Eigenschaften von dem Themenobjekt, das denselben Namen wie der Subskriptionspunkt hat.

Alle Eigenschaften, die in dem übereinstimmenden Themenobjekt nicht festgelegt sind, werden wie üblich von SYSTEM.BASE.TOPIC übernommen.

Vorhandene Publish/Subscribe-Anwendungen in der Warteschlange, die MQRFH2 -Nachrichtenheader verwenden, funktionieren weiterhin, indem sie die Eigenschaft **SubPoint** in den Befehlsnachrichten Publish oder Register subscriber festlegen. Der Subskriptionspunkt wird in der Befehlsnachricht mit der Themenzeichenfolge kombiniert und das sich ergebende Thema wird wie alle anderen Themen verarbeitet.

IBM MQ -Anwendungen sind von Subskriptionspunkten nicht betroffen. Wenn eine Anwendung ein Thema verwendet, das Informationen von einem der passenden Themenobjekte übernimmt, interagiert sie mit einer eingereihten Anwendung, die den entsprechenden Subskriptionspunkt verwendet.

**Beispiel**

Von einer vorhandenen Publish/Subscribe-Anwendung von WebSphere Message Broker (jetzt "IBM Integration Bus"), die nach IBM MQ migriert wurde, wurden die beiden Themenobjekte GBP und USD mit den entsprechenden Themenzeichenfolgen 'GBP' und 'USD' erstellt.

Vorhandene Publisher zum Thema NYSE/IBM/SPOT, die migriert wurden, um in IBM MQ ausgeführt werden zu können und die den Subskriptionspunkt USD verwenden, erstellen Veröffentlichungen zum Thema USD/NYSE/IBM/SPOT. Ebenso erstellen vorhandene Subskribenten von NYSE/IBM/SPOT, die den Subskriptionspunkt USD verwenden, Subskriptionen in USD/NYSE/IBM/SPOT.

Abonnieren Sie den Spotpreis in Dollar in einem IBM MQ -Publish/Subscribe-Programm, indem Sie MQSUBaufrufen. Erstellen Sie eine Subskription mithilfe des Themenobjekts USD und der Themenzeichenfolge 'NYSE/IBM/SPOT', wie im 'C'-Codefragment dargestellt.

```
strncpy(sd.ObjectName, "USD", MQ_TOPIC_NAME_LENGTH);
sd.ObjectString.VSPtr = "NYSE/IBM/SPOT";
sd.ObjectString.VSLength = MQVS_NULL_TERMINATED;
MQSUB(Hconn, &sd, &Hobj, &Hsub, &CompCode, &Reason);
```

1. Legen Sie das Attribut CLUSTER der Themenobjekte USD und GBP auf dem Clusterthemenhost fest.
2. Löschen Sie alle Kopien der Themenobjekte USD und GBP in anderen Warteschlangenmanagern im Cluster.
3. Vergewissern Sie sich, dass USD und GBP in SYSTEM.QPUBSUB.SUBPOINT.NAMELIST in jedem Warteschlangenmanager im Cluster definiert sind.

## Beispiel für die Publish/Subscribe-Konfiguration für einen einzelnen Warteschlangenmanager

Abbildung 31 auf Seite 96 zeigt eine grundlegende Publish/Subscribe-Konfiguration für einen einzelnen Warteschlangenmanager. Das Beispiel zeigt die Konfiguration für einen Nachrichtenservice, der Informationen von verschiedenen Publishern zu mehreren Themen bereitstellt:

- Publisher 1 veröffentlicht Sportergebnisse unter dem Thema "Sport"
- Publisher 2 veröffentlicht Aktienkurse unter dem Thema "Stock" (Aktien)
- Publisher 3 veröffentlicht Filmkritiken unter dem Thema "Films" (Filme) und das Fernsehprogramm unter dem Thema "TV"

Drei Subskribenten haben ihr Interesse an verschiedenen Themen angemeldet, sodass ihnen der Warteschlangenmanager die entsprechenden Informationen zusendet:

- Subskribent 1 erhält die Sportergebnisse und Aktienkurse
- Subskribent 2 erhält die Filmkritiken
- Subskribent 3 erhält die Sportergebnisse

Keiner der Subskribenten ist am Thema "TV" interessiert, das Fernsehprogramm wird daher nicht verteilt.

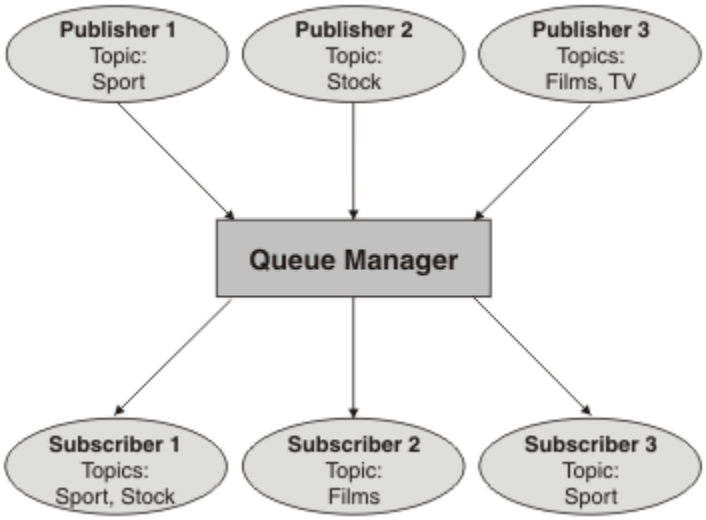



*Abbildung 31. Beispiel für die Publish/Subscribe-Konfiguration für einen einzelnen Warteschlangenmanager*

## Verteilte Publish/Subscribe-Netzwerke

Jeder Warteschlangenmanager gleicht die für ein Thema veröffentlichten Nachrichten mit den lokal erstellten Subskriptionen für das Thema ab. Sie können ein Netz aus Warteschlangenmanagern konfigurieren, sodass Nachrichten, die von einer mit einem Warteschlangenmanager verbundenen Anwendung veröffentlicht werden, an alle übereinstimmenden Subskriptionen auf den anderen Warteschlangenmanagern des Netzes übertragen werden. Dazu müssen die Warteschlangenmanager zusätzlich über einfache Kanäle miteinander verbunden sein.

Bei einer verteilten Publish/Subscribe-Konfiguration handelt es sich um ein Netz aus miteinander verbundenen Warteschlangenmanagern. Die Warteschlangenmanager können sich alle auf dem gleichen physischen System befinden oder über mehrere physische Systeme verteilt sein. Wenn Sie Warteschlangenmanager miteinander verbinden, können Subskribenten eine Subskription für einen Warteschlangenmanager einrichten und über diesen Nachrichten empfangen, die auf einem anderen Warteschlangenmanager veröffentlicht wurden. Zur Veranschaulichung dieses Konzepts wurde der im Abschnitt „Beispiel für die Publish/Subscribe-Konfiguration für einen einzelnen Warteschlangenmanager” auf Seite 95 beschriebenen Konfiguration in der folgenden Abbildung ein zweiter Warteschlangenmanager hinzugefügt.

- Die veröffentlichende Stelle 4 veröffentlicht auf Warteschlangenmanager 2 Wettervorhersagen unter dem Thema 'Wetter' sowie Verkehrsmeldungen für wichtige Straßenverbindungen unter dem Thema 'Verkehr'.
- Subskribent 4 verwendet ebenfalls diesen Warteschlangenmanager und hat das Thema 'Verkehr' subskribiert.
- Subskribent 3 hat zusätzlich das Thema 'Wetter' subskribiert, obwohl er den Warteschlangenmanager einer anderen veröffentlichenden Stelle verwendet. Dies ist möglich, da die Warteschlangenmanager miteinander verbunden sind.

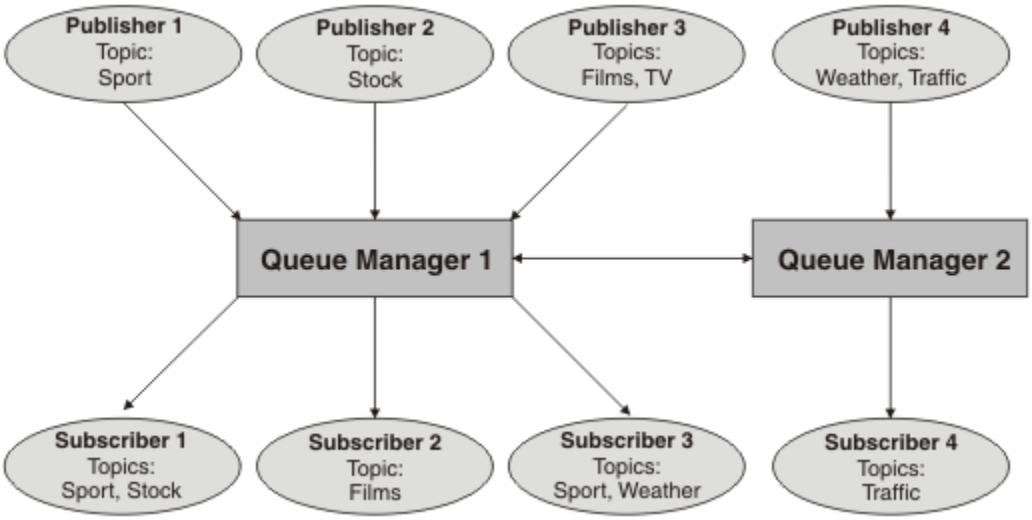



*Abbildung 32. Beispiel für Publish/Subscribe mit zwei Warteschlangenmanagern*

Sie können Warteschlangenmanager manuell in einer Parent-/Child-Hierarchie verbinden oder einen Publish/Subscribe-Cluster erstellen und IBM MQ den größten Teil der Verbindungskonfiguration überlassen. Sie können auch beide Topologien in Kombination verwenden, zum Beispiel indem sie mehrere Cluster zu einer Hierarchie zusammenfassen.

## Übersicht über Publish/Subscribe-Cluster

Ein Publish/Subscribe-Cluster ist ein Standardcluster mit einem oder mehreren Themenobjekten. Wenn Sie auf einem beliebigen Warteschlangenmanager in einem Cluster ein administratives Themenobjekt definieren und aus diesem Themenobjekt durch Angabe eines Clusternamens ein Clusterobjekt machen, können die Publisher und Subskribenten dieses Themas eine Verbindung mit einem beliebigen Warteschlangenmanager des Clusters herstellen, um die veröffentlichten Nachrichten über die Clusterkanäle zwischen den Warteschlangenmanagern an die Subskribenten weiterzuleiten.

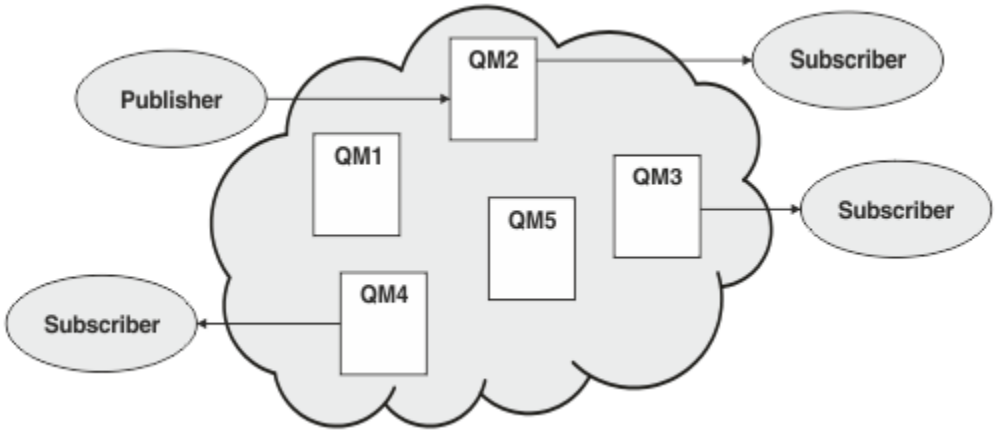



*Abbildung 33. Publish/Subscribe-Cluster*

Die Weiterleitung der Publish/Subscribe-Nachrichten in einem Cluster kann auf zwei Weisen erfolgen:

- Direktes Routing (DIRECT-Routing)
- Topic-Host-Routing (TOPICHOST-Routing)

Wenn Sie ein Cluster-Topic mit DIRECT-Routing konfigurieren, werden die auf einem Warteschlangenmanager veröffentlichten Nachrichten direkt von diesem Warteschlangenmanager an alle Subskriptionen auf den anderen Warteschlangenmanagern des Clusters weitergeleitet. Dadurch wird bei der Weiterleitung von Veröffentlichungen zwar der direkteste Weg eingeschlagen, allerdings führt diese Konfiguration auch

dazu, dass alle Warteschlangenmanager des Clusters alle anderen Warteschlangenmanager kennen und potenziell zwischen allen Warteschlangenmanager des Clusters Clusterkanäle eingerichtet werden.

Beim TOPICHOST-Routing hingegen werden die auf einem Warteschlangenmanager veröffentlichten Nachrichten von diesem Warteschlangenmanager an einen Warteschlangenmanager übertragen, der über eine Definition des verwalteten Themenobjekts verfügt. Der *Topic-Host-Warteschlangenmanager* leitet die Nachricht an jede Subskription in jedem anderen Warteschlangenmanager im Cluster weiter. Falls sich die Publisher und Subskribenten nicht auf den Topic-Host-Warteschlangenmanagern befinden, wird der Übertragungsweg für die Veröffentlichungen länger. Der Vorteil jedoch ist, dass nur die Topic-Host-Warteschlangenmanager die anderen Warteschlangenmanager im Cluster kennen müssen und nur zwischen diesen Clusterkanäle eingerichtet werden.

Weitere Informationen finden Sie unter „Publish/Subscribe-Cluster” auf Seite 99.

## Übersicht über die Publish/Subscribe-Hierarchie

Bei einer Publish/Subscribe-Hierarchie handelt es sich um eine hierarchische Struktur aus Warteschlangenmanagern, die über Kanäle miteinander verbunden sind. Jeder Warteschlangenmanager erkennt seinen *übergeordneten* Warteschlangenmanager, wie im Abschnitt Warteschlangenmanager zu einer Publish/Subscribe-Hierarchie verbinden beschrieben.

Publisher und Subskribenten eines Themas können eine Verbindung mit jedem Warteschlangenmanager der Hierarchie herstellen und die Nachrichten werden in der hierarchischen Struktur der verbundenen Warteschlangenmanager übertragen.

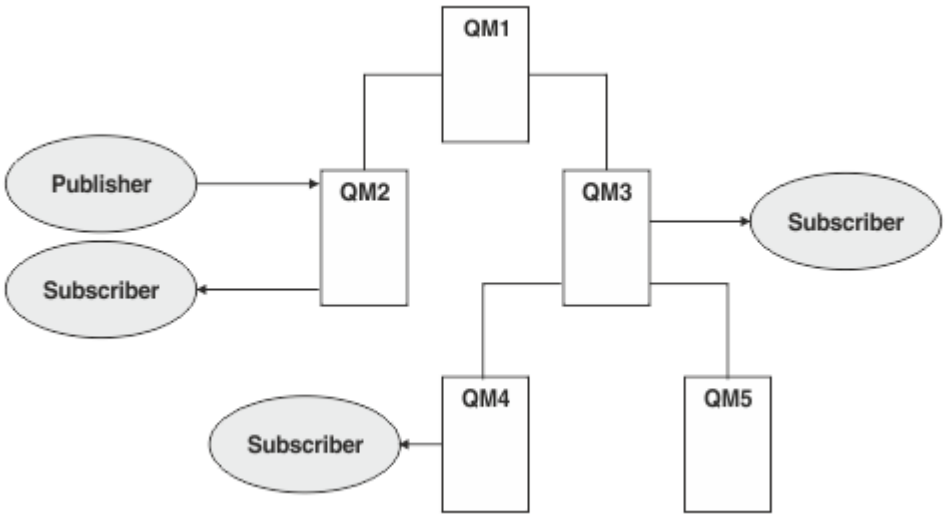



*Abbildung 34. Publish/Subscribe-Hierarchie*

In dieser Abbildung werden Veröffentlichungen, die für die Subskribenten auf QM3 und QM4 bestimmt sind, von QM2 an QM1 und von dort an QM3 und schließlich an QM4 übertragen.

Hierarchien geben Ihnen unmittelbare Kontrolle über die Beziehungen zwischen den einzelnen Warteschlangenmanagern der Hierarchie. Dies ermöglicht eine differenzierte Steuerung der Nachrichtenweiterleitung von Publishern zu Subskribenten, was besonders für das Routing zwischen Warteschlangenmanagernetzen mit eingeschränkter Konnektivität wichtig ist. Sie sollten hierbei besonders auf die Verfügbarkeit und die Möglichkeiten eines jeden Warteschlangenmanagers achten, über den eine Nachricht auf dem Weg vom Publisher zum Subskribenten weitergeleitet wird.

Weitere Informationen finden Sie unter „Publish/Subscribe-Hierarchien” auf Seite 102.

### Verteilung der Veröffentlichungen zwischen den Warteschlangenmanagern

Neben den verschiedenen Routing-Möglichkeiten gibt es auch zwei Methoden der Verteilung der Veröffentlichungen im Netz der Warteschlangenmanager:

- Senden von Veröffentlichungen von einem Warteschlangenmanager nur an diejenigen Warteschlangenmanager, denen zur Zeit eine Subskription für diese Veröffentlichung vorliegt
- Senden aller Veröffentlichungen an alle Warteschlangenmanager - erst auf diesen erfolgt der Abgleich mit den eigenen Subskriptionen

Bei der ersten Methode werden Veröffentlichungen nur bei Bedarf gesendet. Hierzu müssen allerdings alle Warteschlangenmanager über ein gewisses Grad an Subskriptionskenntnissen verfügen. Bei der zweiten Methode sind keine Kenntnisse der Subskriptionen auf den anderen Warteschlangenmanagern erforderlich, allerdings kommt es bei dieser Methode, was die Veröffentlichungen anbelangt, vermutlich zu einem erhöhten und unnötigen Übertragungsaufwand.

IBM MQ verwendet standardmäßig die erste Methode, bei der Veröffentlichungen nur an Warteschlangenmanager gesendet werden, die auch die entsprechende Subskription aufweisen. Die Subskriptionsdaten werden zwischen den Warteschlangenmanagern in Form von *Proxy-Subskriptionen* übertragen. Welche Methode in einer Publish/Subscribe-Topologie die effizientere ist, hängt von der Verteilung und der Lebensdauer der Subskriptionen und dem Veröffentlichungsintervall ab. Weitere Informationen finden Sie im Abschnitt Subskriptionsleistung in Publish/Subscribe-Netzen.

**Zugehörige Konzepte**

„Themenstrukturen” auf Seite 82
Jedes Thema, das Sie definieren, wird in der Themenstruktur durch ein Element oder einen Knoten dargestellt. Die Themenstruktur kann entweder leer sein, um ganz neu zu beginnen, oder Themen enthalten, die zuvor mithilfe von WebSphere MQ-Scriptbefehlen oder PCF-Befehlen definiert wurden. Sie können ein neues Thema definieren, indem Sie entweder die Befehle zum Erstellen von Themen verwenden oder das Thema zum ersten Mal in einer Veröffentlichung oder Subskription angeben.

Publish/Subscribe-Hierarchieszenarios

**Zugehörige Tasks**

Publish/Subscribe-Cluster entwerfen

## Publish/Subscribe-Cluster

Ein Publish/Subscribe-Cluster ist ein Standardcluster aus verbundenen Warteschlangenmanagern, in dem Veröffentlichungen automatisch aus den veröffentlichenden Anwendungen in Subskriptionen verschoben werden, die auf einem beliebigen der zu einem Cluster zusammengefassten Warteschlangenmanager vorliegen. Es gibt zwei Optionen zum Weiterleiten von Publikationen über einen Publish/Subscribe-Cluster: *direktes Routing* und *Topic-Host-Routing*. Für welches Verfahren Sie sich entscheiden, hängt von der Größe und dem erwarteten Aktivitätsmuster Ihres Clusters ab.

Ein Cluster für Publish/Subscribe-Messaging unterscheidet sich nicht von einem IBM MQ-Standardcluster. Das bedeutet, dass sich die Warteschlangenmanager innerhalb eines Publish/Subscribe-Clusters auf physisch getrennten Computern befinden können und bei Bedarf automatisch über Clusterkanäle miteinander verbunden werden. Weitere Informationen hierzu finden Sie im Abschnitt Clusters.

Zur Konfiguration eines Standard-Warteschlangenmanagerclusters für Publish/Subscribe-Messaging definieren Sie auf einem Warteschlangenmanager des Clusters ein oder mehrere verwaltete Themenobjekte. Um aus diesem Thema ein Cluster-Topic zu machen, geben Sie unter der Eigenschaft **CLUSTER** den Namen des Clusters an. Danach wird jedes Thema, das von einem Publisher oder Subskribenten an dieser Stelle in der Themenstruktur oder darunter verwendet wird, für alle Warteschlangenmanager des Clusters freigegeben und Nachrichten, die in einem geclusterten Zweig der Themenstruktur veröffentlicht werden, werden automatisch an die Subskriptionen auf anderen Warteschlangenmanagern des Clusters weitergeleitet.

Unabhängig von der Anzahl der Subskribenten der Nachricht auf dem Ziel-Warteschlangenmanager wird nur eine Kopie jeder Nachricht von dem Warteschlangenmanager des Publishers an jeden der anderen

Warteschlangenmanager übertragen. Erst beim Eingang auf einem Warteschlangenmanager mit mehreren Subskriptionen wird die Nachricht entsprechend der Anzahl der Subskriptionen dupliziert.

Jeder Warteschlangenmanager, der neu in den Cluster aufgenommen wird, wird automatisch über die Clusterthemen informiert, und die Publisher und Subskribenten auf diesem Warteschlangenmanager nehmen automatisch am Cluster teil.

In einem Publish/Subscribe-Cluster können auch nicht geclusterte Publish/Subscribe-Aktivitäten stattfinden, wenn Themenzeichenfolgen verwendet werden, die keinem geclusterten Themenobjekt angehören.

Es gibt zwei Optionen zum Weiterleiten von Publikationen über einen Publish/Subscribe-Cluster: *direktes Routing* und *Topic-Host-Routing*. Um das Nachrichtenrouting auszuwählen, das im Cluster verwendet werden soll, müssen Sie für das verwaltete Themenobjekt für die Eigenschaft **CLROUTE** einen der folgenden Werte festlegen:

- **DIRECT**
- **TOPICHOST**

Topic-Routing ist standardmäßig **DIRECT**. Vor IBM MQ 8.0 war dies die einzige Option. Wenn Sie ein direkt geroutetes Cluster-Topic in einem Warteschlangenmanager konfigurieren, werden sämtliche Warteschlangenmanager im Cluster aller anderen Warteschlangenmanager im Cluster gewahr. Bei der Ausführung von Publish- und Subscribe-Operationen kann jeder Warteschlangenmanager direkt eine Verbindung zu anderen Warteschlangenmanagern im Cluster herstellen.

Ab IBM MQ 8.0können Sie Topic-Routing stattdessen als **TOPICHOST**konfigurieren. Bei Verwendung der Routing-Methode TOPICHOST können alle Warteschlangenmanager im Cluster die Clusterwarteschlangenmanager erkennen, die die Definition des weitergeleiteten Themas enthalten (d. h. die Warteschlangenmanager, in denen Sie das Themenobjekt definiert haben). Beim Ausführen von Publish/Subscribe-Operationen werden Warteschlangenmanager im Cluster nur mit diesen Topic-Host-Warteschlangenmanagern und nicht direkt miteinander verbunden. Die Topic-Host-Warteschlangenmanager sind für das Routing von Publikationen aus Warteschlangenmanagern verantwortlich, in denen Publikationen für Warteschlangenmanager mit übereinstimmenden Subskriptionen veröffentlicht werden.

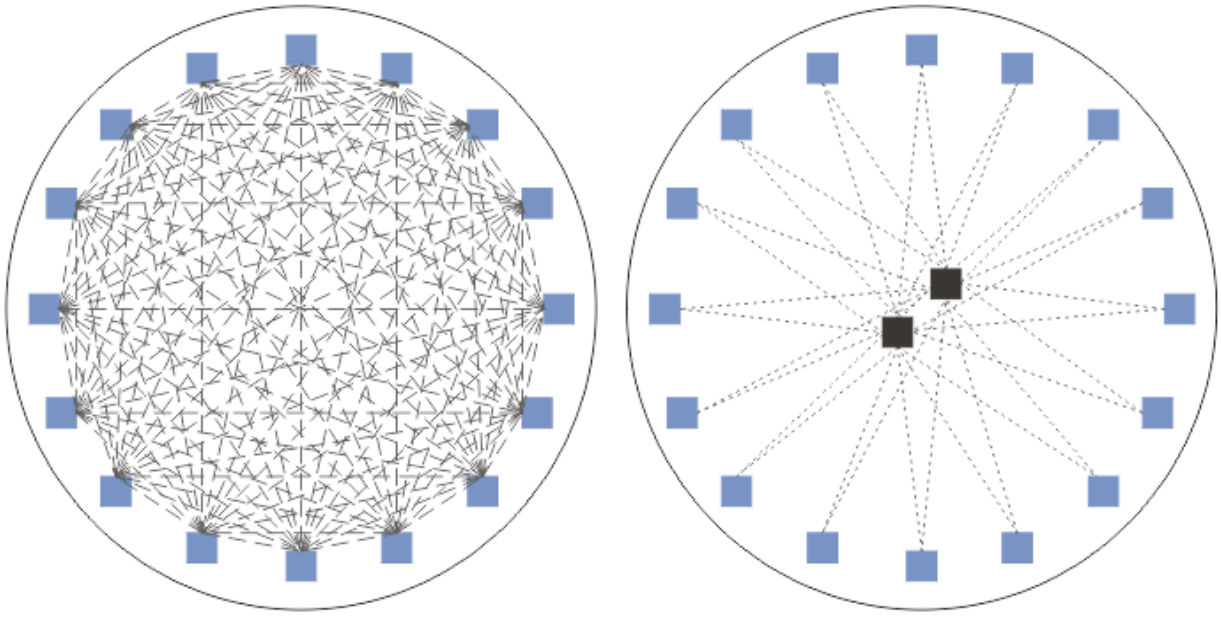


*Abbildung 35. DIRECT- und TOPICHOST-Routing*

## Übersicht über DIRECT-Routing

Wenn ein verwaltetes Topic-Objekt für DIRECT-Routing konfiguriert wird, muss das Topic-Objekt nur auf einem der Warteschlangenmanager im Cluster definiert werden, damit alle Warteschlangenmanager

darüber erfahren. Auf welchem Warteschlangenmanager das Topic-Objekt definiert wird, wirkt sich nicht auf das Verhalten des Publish/Subscribe-Messaging für das Thema aus.

Jede Nachricht wird auf direktem Weg vom veröffentlichenden Warteschlangenmanager ohne Zwischenstation über andere Warteschlangenmanager an jede Subskription auf den anderen Warteschlangenmanagern des Clusters übertragen.

Standardmäßig werden Nachrichten nur an die Warteschlangenmanager im Cluster gesendet, auf denen eine oder mehrere Subskriptionen vorliegen.

- Es ist Aufgabe jedes Warteschlangenmanagers, alle anderen Warteschlangenmanager im Cluster direkt über alle Themen zu informieren, die deren Subskriptionen betreffen. Dies führt dazu, dass allen Warteschlangenmanagern im Cluster alle subskribierten Themen bekannt sind, und dass alle Warteschlangenmanager, die eine Subskription bereitstellen, einen Verbindungskanal mit allen anderen Warteschlangenmanagern aufbauen. Dieser Mechanismus erfolgt unabhängig davon, ob die einzelnen Warteschlangenmanager über Publisher verfügen.
- Die Informationen zu den subskribierten Themen der einzelnen Warteschlangenmanager können durch einen Wechsel zu einem Modell entfernt werden, bei dem alle Veröffentlichungen unabhängig von den auf den einzelnen Warteschlangenmanagern vorliegenden Subskriptionen an alle Warteschlangenmanager des Clusters gesendet werden. Dadurch reduziert sich der Austausch der Subskriptionsdaten, der durch Veröffentlichungen bewirkte Datenverkehr sowie möglicherweise die Anzahl der von den Warteschlangenmanagern eingerichteten Kanäle erhöht sich hingegen. Weitere Informationen finden Sie im Abschnitt Subskriptionsleistung in Publish/Subscribe-Netzen.

Publish/Subscribe-Nachrichtenflüsse mit direkt weitergeleiteten Cluster-Topics können über mehrere Publish/Subscribe-Cluster verlaufen, wenn aus jedem Cluster ein Warteschlangenmanager zu einer Publish/Subscribe-Hierarchie verbunden wird. Weitere Informationen hierzu finden Sie im Abschnitt Themenbereiche mehrerer Cluster zusammenfassen.

Nähere Ausführungen zum direkten Routing finden Sie im Abschnitt Direktes Routing in Publish/Subscribe-Clustern.

## Übersicht über TOPICHOST-Routing

Wenn ein verwaltetes Topic-Objekt für TOPICHOST-Routing konfiguriert ist, werden die Veröffentlichungen eines Warteschlangenmanagers des Clusters über einen Warteschlangenmanager weitergeleitet, auf dem das Topic-Objekt konfiguriert ist (dies ist der "Topic-Host"). Von diesem Topic-Host werden die Veröffentlichungen dann an die Warteschlangenmanager weitergeleitet, auf denen die entsprechenden Subskriptionen vorliegen.

- Es ist Aufgabe jedes Warteschlangenmanagers, alle Topic-Hosts über alle Themen zu informieren, für die eine oder mehrere Subskriptionen vorliegen. Jeder Warteschlangenmanager, auf dem eine Subskription vorliegt, baut für die jeweilige Subskription einen Kanal mit dem zugehörigen Topic-Host auf.
- Warteschlangenmanager, die keine Themen bereitstellen, werden im Rahmen des Publish/Subscribe nicht über andere Warteschlangenmanager des Clusters informiert, die ebenfalls keine Topic-Hosts sind, und zwischen diesen Nicht-Topic-Hosts werden auch keine Kanäle eingerichtet.
- Wenn die veröffentlichende Anwendung mit einem Warteschlangenmanager verbunden ist, auf dem das Thema bereitgestellt wird, werden die veröffentlichten Nachrichten direkt - ohne zwischengeschaltete "Hops" - an die Warteschlangenmanager weitergeleitet, auf dem entsprechende Subskriptionen erstellt wurden. Ebenso werden für ein Thema veröffentlichte Nachrichten, wenn die passenden Subskriptionen auf dem einzigen Warteschlangenmanager erstellt wurden, auf dem das Thema bereitgestellt wird, direkt - ohne zwischengeschaltete "Hops" - an diesen Warteschlangenmanager weitergeleitet.
- Subskriptionen, die sich auf demselben Warteschlangenmanager wie der Publisher befinden, werden unmittelbar bedient, ohne dass die Veröffentlichungen zuerst an die Hosts des Topic-Objekts weitergeleitet werden.

Wie Clusterwarteschlangen können auch mehrere Warteschlangenmanager das gleiche Topic-Verwaltungsobjekt konfigurieren. Dadurch wird eine höhere Verfügbarkeit des Nachrichtenroutings sowie eine horizontale Skalierung durch Lastausgleich sichergestellt. Wenn auf mehreren Warteschlangenmanagern das gleiche benannte Thema für den gleichen Zweig der Themenstruktur konfiguriert wird, wird bei

Topic-Objekten, die über Topic-Hosts weitergeleitet werden, jeder Topic-Host über die von den einzelnen Warteschlangenmanagern subskribierten Themen informiert.

- Eine veröffentlichte Nachricht wird an einen der Topic-Host-Warteschlangenmanager weitergeleitet, von dem aus die Nachricht an die Warteschlangenmanager mit der entsprechenden Subskription weitergeleitet wird. Bei der Auswahl des Topic-Host-Warteschlangenmanagers wird nach denselben Standardlastausgleichsregeln vorgegangen wie bei Punkt-zu-Punkt-Clusterwarteschlangen.
- Können ein oder auch mehrere Topic-Host-Warteschlangenmanager vom veröffentlichenden Warteschlangenmanager nicht kontaktiert werden, so werden die Nachrichten an die übrigen verfügbaren Topic-Host-Warteschlangenmanager weitergeleitet.

Jede Veröffentlichung zu einem Thema in einem weitergeleiteten Zweig der Themenstruktur wird an einen der Topic-Hosts weitergeleitet, selbst wenn im gesamten Cluster keine Subskription zu diesem Thema vorliegt. Standardmäßig werden Nachrichten von dort aus nur an die Warteschlangenmanager im Cluster gesendet, auf denen eine oder mehrere Subskriptionen vorliegen.

- Erforderlich hierzu ist, dass jeder Topic-Host-Warteschlangenmanager über alle subskribierten Themenzeichenfolgen auf allen Warteschlangenmanagern im Cluster informiert wird.
- Die Informationen zu den subskribierten Themen können durch einen Wechsel zu einem Modell entfernt werden, bei dem alle Veröffentlichungen unabhängig von den auf den einzelnen Warteschlangenmanagern vorliegenden Subskriptionen an alle Warteschlangenmanager des Clusters gesendet werden. Dadurch reduziert sich der Austausch der Subskriptionsdaten, der durch Veröffentlichungen bewirkte Datenverkehr sowie möglicherweise die Anzahl der mit den Topic-Host-Warteschlangenmanagern eingerichteten Kanäle erhöht sich hingegen. Weitere Informationen finden Sie im Abschnitt Subskriptionsleistung in Publish/Subscribe-Netzen.

Publish/Subscribe-Nachrichtenflüsse mit via Topic-Host-Routing weitergeleiteten Cluster-Topics können **nicht** mittels einer Publish/Subscribe-Hierarchie über mehrere Publish/Subscribe-Cluster verlaufen.

Nähere Ausführungen zum Topic-Host-Routing finden Sie im Abschnitt Topic-Host-Routing in Publish/Subscribe-Clustern.

## Publish/Subscribe-Hierarchien

Sie haben eine Publish/Subscribe-Hierarchie erstellt, indem Sie Warteschlangenmanager über Kanäle verbunden und dann jeweils zwischen Paaren dieser Warteschlangenmanager eine Child/Parent-Beziehung definiert haben. Eine Nachricht wird in dieser Hierarchie durch die direkten Beziehungen zwischen den Warteschlangenmanagern von einem Publisher zu den Subskriptionen übertragen. Allerdings können mehrere "Hops" erforderlich sein, bis eine Nachricht an ihr Ziel gelangt.

Unabhängig von der Anzahl der Subskribenten der Nachricht auf dem Ziel-Warteschlangenmanager wird jeweils zwischen den Warteschlangenmanagerpaaren nur eine Kopie der Nachricht übertragen. Erst beim Eingang auf einem Warteschlangenmanager mit mehreren Subskriptionen wird die Nachricht entsprechend der Anzahl der Subskriptionen dupliziert.

Standardmäßig werden Nachrichten nur an diejenigen Warteschlangenmanager in der Hierarchie gesendet, die direkt auf dem Weg zu einer Subskription auf einem anderen Warteschlangenmanager liegen:

- Es ist Aufgabe jedes Warteschlangenmanagers, jede direkte Beziehung über alle Themen zu informieren, für die zur Zeit auf dem gleichen oder dem anderen Warteschlangenmanager der jeweiligen Beziehung eine oder mehrere Subskriptionen vorliegen. Dies führt dazu, dass allen Warteschlangenmanagern in der Hierarchie alle subskribierten Themen bekannt sind.
- Dieses Verhalten kann jedoch auch geändert werden, sodass Veröffentlichungen generell an alle Warteschlangenmanager der Hierarchie gesendet werden, unabhängig davon, ob auf dem jeweiligen Warteschlangenmanager Subskriptionen vorliegen. In diesem Fall brauchen innerhalb der Hierarchie keine Subskriptionsdaten übertragen zu werden, dafür erhöht sich der Datenverkehr für die Veröffentlichungen.

Wenn Sie einen Cluster erstellen, müssen Sie darauf achten, dass keine Schleife entsteht, über die die Nachrichten endlos innerhalb des Netzes kreisen. In einer Hierarchie ist eine solche Schleife erst gar nicht möglich.

Jeder Warteschlangenmanager muss einen eindeutigen Namen haben.

Publish/Subscribe-Nachrichtenflüsse können über mehrere Publish/Subscribe-Cluster verlaufen. Dazu brauchen Sie lediglich einen Warteschlangenmanager aus jedem Cluster zu einer Hierarchie zusammenzufügen.

Nähere Ausführungen zum Routing innerhalb einer Hierarchie finden Sie im Abschnitt Routing in Publish/Subscribe-Hierarchien.

## Proxy-Subskriptionen in einem Publish/Subscribe-Netz

Eine Proxy-Subskription ist eine Subskription, die von einem Warteschlangenmanager für Themen eingerichtet wird, die auf einem anderen Warteschlangenmanager veröffentlicht werden. Eine Proxy-Subskription fließt zwischen Warteschlangenmanagern für jede einzelne Themenzeichenfolge, für die eine Subskription eingerichtet wurde. Sie müssen Proxy-Subskriptionen nicht explizit erstellen; das macht der Warteschlangenmanager automatisch für Sie.

Sie können Warteschlangenmanager in einem Publish/Subscribe-Cluster oder in einer Publish/Subscribe-Hierarchie miteinander verbinden. Proxy-Subskriptionen fließen zwischen den verbundenen Warteschlangenmanagern. Proxy-Subskriptionen sorgen dafür, dass Veröffentlichungen für ein Thema, die von einem Publisher erstellt werden, der mit einem der Warteschlangenmanager verbunden ist, auch von Subskribenten des Themas empfangen werden, die mit anderen Warteschlangenmanagern verbunden sind. Siehe „Verteilte Publish/Subscribe-Netzwerke” auf Seite 96.

In Publish/Subscribe-Topologien, in denen es Tausende von Subskriptionen für einzelne Themenzeichenfolgen gibt oder in denen die Anzahl dieser Subskriptionen möglicherweise stark schwankt, muss der Aufwand für die Weitergabe der Proxy-Subskriptionen berücksichtigt werden. Neben der im verbleibenden Teil dieses Abschnitts beschriebenen Aggregation können Sie manuelle Konfigurationsänderungen vornehmen, die den Fluss der Proxy-Subskriptionen und Veröffentlichungen zwischen verbundenen Warteschlangenmanagern weiterhin einschränken und die Latenzzeit reduzieren, bis eine Proxy-Subskription an alle verbundenen Warteschlangenmanager weitergeleitet ist. Weitere Informationen finden Sie im Abschnitt Subskriptionsleistung in Publish/Subscribe-Netzen.

Proxy-Subskriptionen enthalten keine von lokalen Subskriptionen verwendeten Selektoren, und Subskriptions-Themenzeichenfolgen, die Platzhalterzeichen enthalten, könnten vereinfacht werden. Dies kann zu mit Proxy-Subskriptionen übereinstimmenden Veröffentlichungen führen, obwohl die eigentliche Subskription nicht übereinstimmt, was wiederum dazu führt, dass zwischen den Warteschlangenmanagern zusätzliche Veröffentlichungen übertragen werden. Der die Subskriptionen bereitstellende Warteschlangenmanager filtert solche Abweichungen aus, sodass zusätzliche Veröffentlichungen nicht an die Subskriptionen zurückgegeben werden.

### Zusammenfassung von Proxy-Subskriptionen

Proxy-Subskriptionen werden mithilfe eines Duplikateliminierungssystems zusammengefasst. Für eine einzelne aufgelöste Themenzeichenfolge wird bei der ersten lokalen Subskription oder empfangenen Proxy-Subskription eine Proxy-Subskription gesendet. Nachfolgende Subskriptionen für dieselbe Themenzeichenfolge nutzen diese bestehende Proxy-Subskription.

Die Proxy-Subskription wird aufgehoben, nachdem die letzte lokale Subskription oder empfangene Proxy-Subskription aufgehoben wurde.

### Zusammenfassung von Veröffentlichungen

Wenn es auf einem Warteschlangenmanager mehrere Subskriptionen für dieselbe Themenzeichenfolge gibt, senden andere Warteschlangenmanager in der Publish/Subscribe-Topologie nur eine einzige Kopie von jeder Veröffentlichung für die betreffende Themenzeichenfolge. Bei Eingang der Nachricht liefert der lokale Warteschlangenmanager eine Kopie der Nachricht an alle übereinstimmenden Subskriptionen.

Wenn die Proxy-Subskriptionen Platzhalterzeichen enthalten, ist es möglich, dass mehrere Proxy-Subskriptionen mit der Themenzeichenfolge einer einzelnen Veröffentlichung übereinstimmen. Wird auf einem Warteschlangenmanager eine Nachricht veröffentlicht, die mit mehreren, von einem einzelnen ver-

bundenen Warteschlangenmanager erstellte Proxy-Subskriptionen übereinstimmt, wird nur eine einzige Kopie der Veröffentlichung an den fernen Warteschlangenmanager weitergeleitet, um alle vorhandenen Proxy-Subskriptionen zu bedienen.

**Zugehörige Konzepte**

Schleifenermittlung in einem Netz mit verteiltem Publish/Subscribe

### *Platzhalter in Proxy-Subskriptionen*

Zum Abgleich mehrerer Themenzeichenfolgen in Veröffentlichungen können Subskriptionen Platzhalterzeichen in Themenzeichenfolgen verwenden.

Für Subskriptionen sind zwei Schemas für Platzhalterzeichen möglich: *topic-based* und *character-based*. Siehe„Platzhalterschemas” auf Seite 76.

In IBM MQ werden alle Proxy-Subskriptionen für Platzhaltersubskriptionen für die Verwendung von themenbasierten Platzhalterzeichen konvertiert. Zeichenbasierte (character-based) Platzhalterzeichen werden bis zurück zum nächsten / durch das Zeichen # ersetzt. Zum Beispiel wird `/aaa/bbb/c*d` umgewandelt in `/aaa/bbb/#`. Die Umwandlung führt dazu, dass ferne Warteschlangenmanager etwas mehr Veröffentlichungen senden, als explizit subskribiert wurden. Die zusätzlichen Veröffentlichungen werden vom lokalen Warteschlangenmanager ausgefiltert, wenn er die Veröffentlichungen an seine lokalen Subskribenten weitergibt.

## Verwendung von Platzhaltern mit der Eigenschaft WILDCARD steuern

Mit der MQSC-Eigenschaft **`Topic`** WILDCARD oder der entsprechenden PCF-Eigenschaft Topic `WildcardOperation` können Sie die Zustellung von Veröffentlichungen an Subskribentenanwendungen steuern, die Namen von Platzhalterthemenzeichenfolgen verwenden. Die Eigenschaft WILDCARD kann einen von zwei möglichen Werten haben:

**WILDCARD**
Aktionen von Subskriptionen mit Platzhaltern bezüglich dieses Themas.

**PASSTHRU**
Subskriptionen für ein Thema mit Platzhalter, das weniger spezifisch ist als die Themenzeichenfolge für dieses Themenobjekt, empfangen Veröffentlichungen zu diesem Thema und zu spezifischeren Themenzeichenfolge.

**BLOCK**
Subskriptionen für ein Thema mit Platzhalter, das weniger spezifisch ist als die Themenzeichenfolge für dieses Themenobjekt, empfangen keine Veröffentlichungen zu diesem Thema und zu spezifischeren Themenzeichenfolge.

Der Wert für dieses Attribut wird bei der Definition von Subskriptionen verwendet. Wenn Sie dieses Attribut ändern, ist die Gruppe von Themen, die bereits durch vorhandene Subskriptionen abgedeckt sind, nicht durch die Änderung betroffen. Dieses Szenario gilt auch, wenn sich durch die Erstellung oder das Löschen von Themenobjekten die Topologie ändert; die Themen mit Subskriptionen, die nach der Änderung des Attributs WILDCARD erstellt wurden, werden mit der geänderten Topologie erstellt. Wenn die Themen mit den vorhandenen Subskriptionen übereinstimmen sollen, müssen Sie den Warteschlangenmanager neu starten.

Führen Sie die Schritte im Beispiel „Beispiel: Publish/Subscribe-Cluster Sport erstellen” auf Seite 90 aus, um die in Abbildung 23 auf Seite 87 gezeigte Themenstruktur zu erstellen.

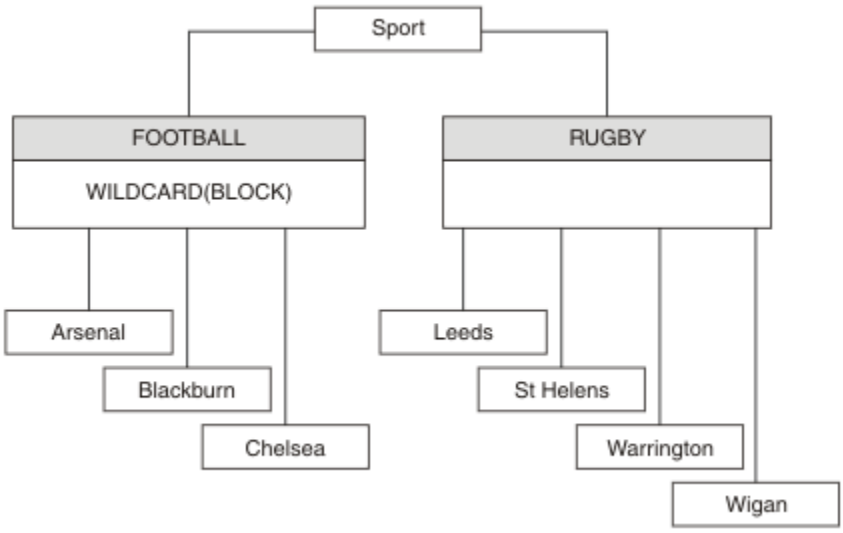



*Abbildung 36. Themenstruktur, die die WILDCARD-Eigenschaft BLOCK verwendet*

Ein Subskribent, der die Platzhalter-Themenzeichenfolge # verwendet, empfängt alle Veröffentlichungen für das Thema `Sport` und die Unterverzeichnisstruktur `Sport/Rugby` Der Subskribent empfängt keine Veröffentlichungen für die untergeordnete Baumstruktur `Sport/Football` , weil der Wert der Eigenschaft `WILDCARD` des Themas `Sport/Football` BLOCKist.

Die Standardeinstellung lautet PASSTHRU. Sie können den Wert PASSTHRU der Eigenschaft `WILDCARD` auf Knoten in der Baumstruktur `Sport` festlegen. Wenn die Knoten nicht über den Eigenschaftswert `WILDCARD` verfügen BLOCK, ändert die Einstellung PASSTHRU nicht das Verhalten, das von Subskribenten für Knoten in der `Sports` -Baumstruktur beobachtet wird.

Erstellen Sie im Beispiel Subskriptionen, um zu sehen, wie sich die Platzhaltereinstellung auf die bereitgestellten Veröffentlichungen auswirkt (siehe dazu Abbildung 27 auf Seite 92). Führen Sie den Publish-Befehl in Abbildung 30 auf Seite 93 aus, um einige Veröffentlichungen zu erstellen.

```
pub QMA
```

*Abbildung 37. Für QMA publizieren*

Die Ergebnisse werden in Tabelle 3 auf Seite 87 gezeigt. Beachten Sie, wie die Einstellung des `WILDCARD`-Eigenschaftswerts BLOCK verhindert, dass Subskriptionen mit Platzhaltern Veröffentlichungen für Themen empfangen, die innerhalb des Platzhalterumfangs liegen.

*Tabelle 6. Auf QMA empfangene Veröffentlichungen*

| **Abonnement** | **Themenzeichenfolge** | **Empfangene Veröffentlichungen** | **Anmerkungen** |
|---|---|---|---|
| SPORTS | `Sports/#` | `Sports`<br>`Sports/Rugby`<br>`Sports/Rugby/Leeds` | Alle Veröffentlichungen zur Unterverzeichnisstruktur 'Football', die von `WILDCARD(BLOCK)` auf `Sports/Football` blockiert werden |
| SARSENAL | `Sports/#/Arsenal` | - | `WILDCARD(BLOCK)` on `Sports/Football` verhindert Platzhaltersubskription unter `Arsenal` |
| SLEEDS | `Sports/#/Leeds` | `Sports/Rugby/Leeds` | Standardwert `WILDCARD` für 'Sport/Rugby' verhindert keine Platzhaltersubskription auf `Leeds`. |

**Anmerkung:**

Angenommen, eine Subskription hat einen Platzhalter, der einem Themenobjekt mit dem WILDCARD-Eigenschaftswert BLOCK entspricht. Wenn die Subskription auch eine Themenzeichenfolge rechts neben dem entsprechenden Platzhalter hat, empfängt die Subskription niemals Veröffentlichungen. Die nicht geblockten Veröffentlichungen sind Veröffentlichungen für Themen, die dem geblockten Platzhalter übergeordnet sind. Veröffentlichungen für Themen, die dem Thema mit dem Eigenschaftswert BLOCK untergeordnet sind, werden vom Platzhalter geblockt. Subskriptionsthemen-Zeichenfolgen, die ein Thema rechts neben dem Platzhalter einschließen, empfangen daher niemals passende Veröffentlichungen.

Wenn der Eigenschaftswert WILDCARD auf BLOCK gesetzt wird, ist keine Subskription mithilfe einer Themenzeichenfolge möglich, die Platzhalter beinhaltet. Eine solche Subskription ist normal. Die Subskription hat ein explizites Thema, das dem Thema mit einem Themenobjekt entspricht, das den WILDCARD-Eigenschaftswert BLOCK hat. Sie verwendet Platzhalter für Themen, die dem WILDCARD-Eigenschaftswert BLOCK über- oder untergeordnet sind. Im Beispiel in Abbildung 23 auf Seite 87kann eine Subskription wie Sports/Football/# Veröffentlichungen empfangen.

## Platzhalter und Cluster-Topics

Definitionen von Cluster-Topics werden an jeden Warteschlangenmanager in einem Cluster weitergegeben. Die Subskription eines Cluster-Topics auf einem Warteschlangenmanager in einem Cluster führt dazu, dass der Warteschlangenmanager Proxy-Subskriptionen erstellt. Eine Proxy-Subskription wird in jedem Warteschlangenmanager im Cluster erstellt. Subskriptionen, die Themenzeichenfolgen mit Platzhaltern verwenden, kombiniert mit Cluster-Topics, lassen das Verhalten schwer vorhersagen. Das Verhalten wird im folgenden Beispiel erläutert.

In dem für das Beispiel verwendeten Cluster „Beispiel: Publish/Subscribe-Cluster Sport erstellen” auf Seite 90 hat QMB dieselben Subskriptionen wie QMA. QMB hat jedoch keine Veröffentlichungen empfangen, nachdem der Publisher an QMA veröffentlicht hat (siehe Abbildung 24 auf Seite 87). Obwohl die Themen Sports/Football und Sports/Rugby Clusterthemen sind, verweisen die in fullsubs.tst definierten Subskriptionen nicht auf ein Clusterthema. Es werden keine Proxy-Subskriptionen von QMB an QMA weitergegeben. Ohne Proxy-Subskriptionen werden keine Veröffentlichungen für QMA an QMB weitergeleitet.

Einige Subskriptionen, wie z. B. Sports/#/Leeds, scheinen möglicherweise auf ein Clusterthema zu verweisen, in diesem Fall Sports/Rugby. Die Sports/#/Leeds -Subskription wird tatsächlich in das Themenobjekt SYSTEM.BASE.TOPICaufgelöst.

Die Regel zum Auflösen des Themenobjekts, das von einer Subskription wie Sports/#/Leeds referenziert wird, lautet wie folgt. Schneiden Sie die Themenzeichenfolge bis zum ersten Platzhalter ab. Suchen Sie links über die Themenzeichenfolge nach dem ersten Thema mit einem zugehörigen Verwaltungsthemenobjekt. Das Themenobjekt kann einen Clusternamen angeben oder ein lokales Themenobjekt definieren. Im Beispiel Sports/#/Leedsist die Themenzeichenfolge nach dem Abschneiden Sports, die kein Themenobjekt hat, und Sports/#/Leeds übernimmt daher von SYSTEM.BASE.TOPIC, einem lokalen Themenobjekt.

Um zu sehen, wie die Subskription von Cluster-Topics die Weitergabe von Platzhaltern ändern kann, führen Sie das Stapelscript upsubs.bat aus. Das Script löscht die Subskriptionswarteschlangen und fügt die Subskriptionen des Cluster-Topics zu fullsubs.tst hinzu. Führen Sie puba.bat erneut aus, um einen Stapel von Veröffentlichungen zu erstellen (siehe Abbildung 24 auf Seite 87).

Tabelle 4 auf Seite 89 zeigt das Ergebnis, nachdem zwei neue Subskriptionen dem Warteschlangenmanager hinzugefügt wurden, in dem die Veröffentlichungen veröffentlicht wurden. Wie erwartet, empfangen die neuen Subskriptionen je eine Veröffentlichung, und die Anzahl der von den anderen Subskriptionen empfangenen Veröffentlichungen ist unverändert. Die unerwarteten Resultate treten im anderen Cluster-Warteschlangenmanager auf, siehe Tabelle 5 auf Seite 89.

| *Tabelle 7. Auf QMA empfangene Veröffentlichungen* | | | |
|---|---|---|---|
| **Abonnement** | **Themenzeichenfolge** | **Empfangene Veröffentlichungen** | **Anmerkungen** |
| SPORTS | Sports/# | Sports<br>Sports/Rugby<br>Sports/Rugby/Leeds | Alle Veröffentlichungen zur Unterverzeichnisstruktur 'Football', die von WILDCARD(BLOCK) auf Sports/Football blockiert werden |
| SARSENAL | Sports/#/Arsenal | - | WILDCARD(BLOCK) on Sports/Football verhindert Platzhaltersubskription unter Arsenal |
| SLEEDS | Sports/#/Leeds | Sports/Rugby/Leeds | Standardwert WILDCARD für 'Sport/Rugby' verhindert keine Platzhaltersubskription auf Leeds. |
| FARSENAL | Sports/Football/Arsenal | Sports/Football/Arsenal | Arsenal empfängt eine Veröffentlichung, da die Subskription keinen Platzhalter hat. |
| FLEEDS | Sports/Rugby/Leeds | Sports/Rugby/Leeds | Leeds würde für jedes Ereignis eine Veröffentlichung empfangen. |

Tabelle 5 auf Seite 89 zeigt die Resultate, nachdem die beiden neuen Subskriptionen zu QMB hinzugefügt und auf QMA veröffentlicht wurden. QMB empfing keine Veröffentlichungen ohne diese beiden neuen Subskriptionen. Wie erwartet empfangen die beiden neuen Subskriptionen Veröffentlichungen, da Sports/FootBall und Sports/Rugby Clusterthemen sind. QMB weitergeleitete Proxy-Subskriptionen für Sports/Football/Arsenal und Sports/Rugby/Leeds an QMA, die dann die Veröffentlichungen an QMBgesendet haben

Das unerwartete Ergebnis ist, dass die beiden Subskriptionen Sports/# und Sports/#/Leeds , die zuvor keine Veröffentlichungen empfangen haben, jetzt Veröffentlichungen empfangen. Dies liegt daran, dass die Sports/Football/Arsenal -und Sports/Rugby/Leeds -Veröffentlichungen, die für die anderen Subskriptionen an QMB weitergeleitet werden, jetzt für jeden Subskribenten verfügbar sind, der QMBzugeordnet ist. Folglich empfangen die Subskriptionen für die lokalen Themen Sports/# und Sports/#/Leeds die Veröffentlichung Sports/Rugby/Leeds . Sports/#/Arsenal erhält weiterhin keine Veröffentlichung, weil für Sport/Fußball die Eigenschaft WILDCARD auf BLOCKgesetzt ist.

| *Tabelle 8. Auf QMB empfangene Veröffentlichungen* | | | |
|---|---|---|---|
| **Abonnement** | **Themenzeichenfolge** | **Empfangene Veröffentlichungen** | **Anmerkungen** |
| *SPORTS* | *Sports/#* | *Sports/Rugby/Leeds* | *All publications to Football subtree blocked by WILDCARD(BLOCK) on Sports/Football* |
| SARSENAL | Sports/#/Arsenal | - | WILDCARD(BLOCK) on Sports/Football verhindert Platzhaltersubskription unter Arsenal |
| *SLEEDS* | *Sports/#/Leeds* | *Sports/Rugby/Leeds* | *Der Standardwert WILDCARD für Sport/Rugby verhindert nicht das Platzhalterabonnement für Leeds.* |
| FARSENAL | Sports/Football/Arsenal | Sports/Football/Arsenal | Arsenal empfängt eine Veröffentlichung, da die Subskription keinen Platzhalter hat. |

| *Tabelle 8. Auf QMB empfangene Veröffentlichungen (Forts.)* | | | |
|---|---|---|---|
| **Abonnement** | **Themenzeichenfolge** | **Empfangene Veröffentlichungen** | **Anmerkungen** |
| FLEEDS | Sports/Rugby/Leeds | Sports/Rugby/Leeds | Leeds würde für jedes Ereignis eine Veröffentlichung empfangen. |

Bei den meisten Anwendungen ist es nicht erwünscht, dass eine Subskription das Verhalten einer anderen Subskription beeinflusst. Eine wichtige Verwendung der Eigenschaft WILDCARD mit dem Wert BLOCK ist es, dass sich die Subskriptionen für dieselbe Themenzeichenfolge mit Platzhalterzeichen einheitlich verhalten. Unabhängig davon, ob sich die Subskription in demselben Warteschlangenmanager wie der Publisher oder in einem anderen Warteschlangenmanager befindet, sind die Resultate der Subskription dieselben.

## Platzhalter und Datenströme

Wenn eine neue Anwendung für die Publish/Subscribe-API geschrieben wird, empfängt eine Subskription von * keine Veröffentlichungen. Um alle Sportveröffentlichungen zu erhalten, müssen Sie Sports/*oder Sports/#und ähnlich für Business -Veröffentlichungen subskribieren.

Das Verhalten einer vorhandenen Publish/Subscribe-Anwendung in der Warteschlange ändert sich nicht, wenn der Publish/Subscribe-Broker auf eine höhere Version von IBM MQmigriert wird. Die Eigenschaft **StreamName** in den Befehlen **Publish**, **Register Publisher** oder **Subscriber** wird dem Namen des Themas zugeordnet, auf das der Datenstrom migriert wurde.

## Platzhalter und Subskriptionspunkte

Wenn eine neue Anwendung für die Publish/Subscribe-API geschrieben wird, empfängt eine Subskription von * keine Veröffentlichungen. Um alle Sportveröffentlichungen zu erhalten, müssen Sie Sports/*oder Sports/#und ähnlich für Business -Veröffentlichungen subskribieren.

Das Verhalten einer vorhandenen Publish/Subscribe-Anwendung in der Warteschlange ändert sich nicht, wenn der Publish/Subscribe-Broker auf eine höhere Version von IBM MQmigriert wird. Die Eigenschaft **SubPoint** in den Befehlen **Publish**, **Register Publisher** oder **Subscriber** wird dem Namen des Themas zugeordnet, auf das die Subskription migriert wurde.

### Beispiel: Publish/Subscribe-Cluster Sport erstellen

In den nachfolgenden Schritten wird ein Cluster CL1 mit vier Warteschlangenmanagern erstellt: zwei vollständige Repositorys, CL1A und CL1B, und zwei Teilrepositorys, QMA und QMB. Die vollständigen Repositorys enthalten nur Clusterdefinitionen. QMA wird als Cluster-Topic-Host festgelegt. Permanente Subskriptionen sind in QMA und QMB definiert.

**Anmerkung:** Das Beispiel ist für Windows codiert. Wenn Sie das Beispiel auf anderen Plattformen konfigurieren und testen möchten, müssen Sie den Programmcode für Create qmgrs.bat und create pub.bat ändern.

1. Erstellen Sie die Scriptdateien.
   a. Create topics.tst
   b. Create wildsubs.tst
   c. Create fullsubs.tst
   d. Create qmgrs.bat
   e. create pub.bat
2. Führen Sie Create qmgrs.bat aus, um die Konfiguration zu erstellen.

```
qmgrs
```

Erstellen Sie die Themen in Abbildung 23 auf Seite 87. Das Script in Abbildung 5 erstellt die Clusterthemen `Sports/Football` und `Sports/Rugby`.

**Anmerkung:** Die Option REPLACE ersetzt nicht die TOPICSTR-Eigenschaften eines Themas. TOPICSTR ist eine Eigenschaft, die im Beispiel gezielt variiert wird, um verschiedene Themenstrukturen zu testen. Wenn Sie ein Thema ändern möchten, müssen Sie das Thema zuerst löschen.

```
DELETE TOPIC ('Sports')
DELETE TOPIC ('Football')
DELETE TOPIC ('Arsenal')
DELETE TOPIC ('Blackburn')
DELETE TOPIC ('Chelsea')
DELETE TOPIC ('Rugby')
DELETE TOPIC ('Leeds')
DELETE TOPIC ('Wigan')
DELETE TOPIC ('Warrington')
DELETE TOPIC ('St. Helens')

DEFINE TOPIC ('Sports')   TOPICSTR('Sports')
DEFINE TOPIC ('Football')  TOPICSTR('Sports/Football') CLUSTER(CL1) WILDCARD(BLOCK)
DEFINE TOPIC ('Arsenal')  TOPICSTR('Sports/Football/Arsenal')
DEFINE TOPIC ('Blackburn') TOPICSTR('Sports/Football/Blackburn')
DEFINE TOPIC ('Chelsea')  TOPICSTR('Sports/Football/Chelsea')
DEFINE TOPIC ('Rugby')   TOPICSTR('Sports/Rugby')  CLUSTER(CL1)
DEFINE TOPIC ('Leeds')   TOPICSTR('Sports/Rugby/Leeds')
DEFINE TOPIC ('Wigan')   TOPICSTR('Sports/Rugby/Wigan')
DEFINE TOPIC ('Warrington') TOPICSTR('Sports/Rugby/Warrington')
DEFINE TOPIC ('St. Helens') TOPICSTR('Sports/Rugby/St. Helens')
```

*Abbildung 38. Themen löschen und erstellen: topics.tst*

**Anmerkung:** Löschen Sie die Themen, da REPLACE keine Themenzeichenfolgen ersetzt.

Erstellen Sie Subskriptionen mit Platzhaltern. Die Platzhalter entsprechen den Themen mit Themenobjekten in Abbildung 23 auf Seite 87. Erstellen Sie für jede Subskription eine Warteschlange. Beim Ausführen bzw. erneuten Ausführen des Scripts werden die Warteschlangen geleert, und die Subskriptionen werden gelöscht.

**Anmerkung:** Die Option REPLACE ersetzt nicht die die Eigenschaften TOPICOBJ oder TOPICSTR einer Subskription. TOPICOBJ oder TOPICSTR sind die Eigenschaften, die im Beispiel sinnvoll variiert werden, um verschiedene Subskriptionen zu testen. Um sie zu ändern, löschen Sie zunächst die Subskription.

```
DEFINE QLOCAL(QSPORTS)  REPLACE
DEFINE QLOCAL(QSARSENAL) REPLACE
DEFINE QLOCAL(QSLEEDS)  REPLACE
CLEAR QLOCAL(QSPORTS)
CLEAR QLOCAL(QSARSENAL)
CLEAR QLOCAL(QSLEEDS)

DELETE SUB  (SPORTS)
DELETE SUB  (SARSENAL)
DELETE SUB  (SLEEDS)
DEFINE SUB  (SPORTS)  TOPICSTR('Sports/#')     DEST(QSPORTS)
DEFINE SUB  (SARSENAL) TOPICSTR('Sports/+/Arsenal') DEST(QSARSENAL)
DEFINE SUB  (SLEEDS)  TOPICSTR('Sports/+/Leeds')  DEST(QSLEEDS)
```

*Abbildung 39. Platzhaltersubskriptionen erstellen: wildsubs.tst*

Erstellen Sie Subskriptionen, die die Cluster-Topic-Objekte referenzieren.

**Anmerkung:**

Das Begrenzungszeichen / wird automatisch zwischen die von TOPICOBJ referenzierte Themenzeichenfolge und die von TOPICSTR definierte Themenzeichenfolge eingefügt.

Die Definition `DEFINE SUB(FARSENAL) TOPICSTR('Sports/Football/Arsenal') DEST(QFARSENAL)` erstellt dieselbe Subskription. Die Verwendung von TOPICOBJ ist eine zeiteffiziente Methode, bereits definierte Themenzeichenfolgen zu referenzieren. Die erstellte Subskription referenziert das Themenobjekt nicht länger.

```
DEFINE QLOCAL(QFARSENAL) REPLACE
DEFINE QLOCAL(QRLEEDS)  REPLACE
CLEAR QLOCAL(QFARSENAL)
CLEAR QLOCAL(QRLEEDS)

DELETE SUB  (FARSENAL)
DELETE SUB  (RLEEDS)
DEFINE SUB  (FARSENAL) TOPICOBJ('Football') TOPICSTR('Arsenal') DEST(QFARSENAL)
DEFINE SUB  (RLEEDS)  TOPICOBJ('Rugby')  TOPICSTR('Leeds')  DEST(QRLEEDS)
```

*Abbildung 40. Subskriptionen löschen und erstellen: fullsubs.tst*

Erstellen Sie einen Cluster mit zwei Repositorys. Erstellen Sie zwei Teilrepositorys zum Veröffentlichen und Abonnieren. Führen Sie das Script erneut aus, um alles zu löschen und neu zu starten. Das Script erstellt zudem die Themenhierarchie und die ursprünglichen Platzhaltersubskriptionen.

**Anmerkung:**

Schreiben Sie auf anderen Plattformen ein ähnliches Script, oder geben Sie alle Befehle ein. Mit einem Script lässt sich alles schneller löschen und mit einer identischen Konfiguration erneut starten.

```
@echo off
set port.CL1B=1421
set port.CLIA=1420
for %%A in (CL1A CL1B QMA QMB) do call :createQM %%A
call :configureQM CL1A CL1B %port.CL1B% full
call :configureQM CL1B CL1A %port.CL1A% full
for %%A in (QMA QMB) do call :configureQM %%A CL1A %port.CL1A% partial
for %%A in (topics.tst wildsubs.tst) do runmqsc QMA < %%A
for %%A in (wildsubs.tst) do runmqsc QMB < %%A
goto:eof

:createQM
echo Configure Queue manager %1
endmqm -p %1
for %%B in (dlt crt str) do %%Bmqm %1
goto:eof

:configureQM
if %1==CL1A set p=1420
if %1==CL1B set p=1421
if %1==QMA set p=1422
if %1==QMB set p=1423
echo configure %1 on port %p% connected to repository %2 on port %3 as %4 repository
echo DEFINE LISTENER(LST%1) TRPTYPE(TCP) PORT(%p%) CONTROL(QMGR) REPLACE | runmqsc %1
echo START LISTENER(LST%1) | runmqsc %1
if full==%4 echo ALTER QMGR REPOS(CL1) DEADQ(SYSTEM.DEAD.LETTER.QUEUE) | runmqsc %1
echo DEFINE CHANNEL(TO.%2) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('LOCALHOST(%3)') CLUSTER(CL1)
REPLACE | runmqsc %1
echo DEFINE CHANNEL(TO.%1) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('LOCALHOST(%p%)') CLUS
TER(CL1) REPLACE | runmqsc %1
goto:eof
```

*Abbildung 41. Warteschlangenmanager erstellen: qmgrs.bat*

Aktualisieren Sie die Konfiguration durch Hinzufügen der Subskriptionen zu den Cluster-Topics.

```
@echo off
for %%A in (QMA QMB) do runmqsc %%A < wildsubs.tst
for %%A in (QMA QMB) do runmqsc %%A < upsubs.tst
```

*Abbildung 42. Subskriptionen aktualisieren: upsubs.bat*

Führen Sie `pub.bat` mit einem Warteschlangenmanager als Parameter aus, um Nachrichten zu veröffentlichen, die die Veröffentlichungsthema-Zeichenfolge enthalten. `Pub.bat` verwendet das Beispielprogramm **amqspub**.

```
@echo off
@rem Provide queue manager name as a parameter
set S=Sports
set S=6 Sports/Football Sports/Football/Arsenal
set S=6 Sports/Rugby Sports/Rugby/Leeds
for %%B in (6) do echo %%B | amqspub %%B %1
```

*Abbildung 43. Veröffentlichen: pub.bat*

**Zugehörige Konzepte**

Subskriptionen mit Platzhalterzeichen und ständige Veröffentlichungen

## Veröffentlichungsumfang

In einem Publish/Subscribe-Cluster bzw. einer Publish/Subscribe-Hierarchie bestimmt auch der Bereich einer Veröffentlichung, ob Warteschlangenmanager die Veröffentlichung an ferne Warteschlangenmanager weiterleiten. Verwalten Sie den Bereich von Veröffentlichungen mit dem Themenattribut **PUBSCOPE**.

Wenn eine Veröffentlichung nicht an ferne Warteschlangenmanager weitergeleitet wird, empfangen nur lokale Subskribenten die Veröffentlichung.

In einem Publish/Subscribe-Cluster wird der Veröffentlichungsumfang weitgehend durch die an bestimmten Stellen der Themenstruktur definierten Cluster-Themenobjekten gesteuert. Der Veröffentlichungsumfang muss festgelegt werden, damit Veröffentlichungen an andere Warteschlangenmanager im Cluster übertragen werden können. Sie sollten den Veröffentlichungsumfang eines Cluster-Topics nur dann einschränken, wenn Sie auf bestimmten Warteschlangenmanagern feinste Kontrolle über bestimmte Themen benötigen.

In einer Publish/Subscribe-Hierarchie wird der Veröffentlichungsumfang weitgehend durch dieses Attribut in Verbindung mit dem Attribut subscription scope (Veröffentlichungsumfang) gesteuert.

Das Attribut **PUBSCOPE** bestimmt den Veröffentlichungsumfang eines bestimmten Themas. Sie können das Attribut auf einen der folgenden Werte setzen:

**QMGR**
: Die Veröffentlichung wird nur an lokale Subskribenten übermittelt. Diese Veröffentlichungen werden als *lokale Veröffentlichungen* bezeichnet. Lokale Veröffentlichungen werden nicht an ferne Warteschlangenmanager weitergeleitet und deshalb nicht von Subskribenten empfangen, die mit fernen Warteschlangenmanagern verbunden sind.

**ALLE**
: Die Veröffentlichung wird an lokale Subskribenten und an Subskribenten, die in einem Publish/Subscribe-Cluster mit fernen Warteschlangenmanagern verbunden sind, übermittelt. Diese Veröffentlichungen werden als *globale Veröffentlichungen* bezeichnet.

**ASPARENT**
: Verwenden Sie die Einstellung **PUBSCOPE** des übergeordneten Themas in der Themenstruktur.

Publisher können auch mit der Nachrichteneinreihungsoption `MQPMO_SCOPE_QMGR` angeben, ob es sich um eine lokale oder globale Veröffentlichung handelt. Wird diese Option verwendet, setzt sie das Verhalten außer Kraft, das mit dem Themenattribut **PUBSCOPE** festgelegt wurde.

**Zugehörige Konzepte**

„Verwaltungsthemenobjekte” auf Seite 83
Mit einem Verwaltungsthemenobjekt können Sie Themen bestimmte, nicht standardmäßige Attribute zuweisen.

**Zugehörige Tasks**

Verteilte Publish/Subscribe-Netze konfigurieren

## Subskriptionsumfang

Der Bereich einer Subskription bestimmt, ob eine Subskription auf einem Warteschlangenmanager nur Veröffentlichungen von lokalen Publishern empfängt oder auch Veröffentlichungen, die auf einem ande-

ren Warteschlangenmanager in einem Publish/Subscribe-Cluster oder einer Publish/Subscribe-Hierarchie erfolgen.

Eine Begrenzung des Subskriptionsbereich auf einen Warteschlangenmanager verhindert, dass Proxy-Subskriptionen an andere Warteschlangenmanager in der Publish/Subscribe-Topologie weitergeleitet werden. Dadurch wird der Publish/Subscribe-Nachrichtenübertragungsverkehr zwischen den Warteschlangenmanagern reduziert.

In einem Publish/Subscribe-Cluster wird der Subskriptionsumfang weitgehend durch die an bestimmten Stellen der Themenstruktur definierten Cluster-Themenobjekten gesteuert. Der Subskriptionsumfang muss festgelegt werden, damit Proxy-Subskriptionen an andere Warteschlangenmanager im Cluster übertragen werden können. Sie sollten den Subskriptionsumfang eines Cluster-Topics nur dann einschränken, wenn Sie auf bestimmten Warteschlangenmanagern feinste Kontrolle über bestimmte Themen benötigen.

In einer Publish/Subscribe-Hierarchie wird der Subskriptionsumfang weitgehend durch dieses Attribut in Verbindung mit dem Attribut publication scope (Veröffentlichungsumfang) gesteuert.

Mithilfe des Topic-Attributs **SUBSCOPE** wird der Umfang der Subskriptionen für ein bestimmtes Thema ermittelt. Sie können das Attribut auf einen der folgenden Werte setzen:

**QMGR**
Eine Subskription empfängt nur lokale Veröffentlichungen und Proxy-Subskriptionen werden nicht an ferne Warteschlangenmanager weitergegeben.

**ALLE**
Eine Proxy-Subskription wird an ferne Warteschlangenmanager in einem Publish/Subscribe-Cluster oder einer Publish/Subscribe-Hierarchie weitergegeben und der Subskribent empfängt lokale und ferne Veröffentlichungen.

**ASPARENT**
Verwenden Sie die Einstellung **SUBSCOPE** des übergeordneten Themas in der Themenstruktur.

Wenn der Subskriptionsbereich für ein Thema auf ALLE gesetzt ist (entweder direkt oder über ASPARENT aufgelöst), können einzelne Subskriptionen für dieses Thema ihren Geltungsbereich auf QMGR beschränken, indem sie beim Erstellen der Subskription MQSO_SCOPE_QMGR angeben. Für eine Themensubskription, deren Umfang auf QMGR gesetzt ist, kann der Umfang jedoch nicht umgekehrt auf ALL erweitert werden.

**Zugehörige Konzepte**
„Verwaltungsthemenobjekte” auf Seite 83
Mit einem Verwaltungsthemenobjekt können Sie Themen bestimmte, nicht standardmäßige Attribute zuweisen.

**Zugehörige Tasks**
Verteilte Publish/Subscribe-Netze konfigurieren

## Themenbereiche

Ein Themenbereich ist eine Gruppe von Themen, die Sie abonnieren oder veröffentlichen können. Ein Warteschlangenmanager in einer Topologie mit verteiltem Publish/Subscribe verfügt über einen Themenbereich, der potenziell Themen einschließt, die auf verbundenen Warteschlangenmanagern in dieser Topologie subskribiert oder veröffentlicht wurden.

**Anmerkung:** Eine Übersicht über die Themen innerhalb eines Warteschlangenmanagers wie administrative Themenobjekte, Themenzeichenfolgen und Themenstrukturen finden Sie im Abschnitt „Themen” auf Seite 74. Weitere Verweise auf Themen (*topics*) im aktuellen Artikel beziehen sich, wenn nicht anders angegeben, auf Themenzeichenfolgen (*topic strings*).

Themen werden zunächst auf eine der folgenden Arten erstellt:

- administrativ, wenn Sie ein Themenobjekt oder eine permanente Subskription definieren
- dynamisch, wenn eine Anwendung eine Veröffentlichung oder Subskription für ein neues Thema erstellt

Themen werden sowohl über Proxy-Subskriptionen als auch durch die Erstellung administrativer Cluster-Topic-Objekte an andere Warteschlangenmanager weitergegeben. Proxy-Subskriptionen bewirken, dass

Veröffentlichungen von dem Warteschlangenmanager, mit dem ein Publisher verbunden ist, an die Warteschlangenmanager von Subskribenten weitergeleitet werden.

Proxy-Subskriptionen werden zwischen allen Warteschlangenmanagern weitergegeben, die durch Beziehungen zwischen übergeordneten und untergeordneten Elementen innerhalb einer Warteschlangenmanagerhierarchie miteinander verbunden sind. Daraus folgt, dass Sie auf jedem der Warteschlangenmanager ein Thema abonnieren können, das auf einem der anderen Warteschlangenmanager in der Hierarchie definiert ist. Solange es einen verbundenen Pfad zwischen den Warteschlangenmanagern gibt, spielt es keine Rolle, wie die Warteschlangenmanager verbunden sind.

Proxy-Subskriptionen werden auch für Subskriptionen für Cluster-Topics in einem Publish/Subscribe-Cluster weitergegeben. Ein Cluster-Topic ist ein Thema, das an ein Themenobjekt angehängt ist, das entweder selbst das Attribut **CLUSTER** besitzt oder das Attribut von seinem übergeordneten Element übernimmt. Themen, die keine Cluster-Topics sind, sind als lokale Themen bekannt und werden nicht im Cluster repliziert. Subskriptionen für lokale Themen geben keine Proxy-Subskriptionen an den Cluster weiter.

Zusammenfassend gesagt, werden also in zwei Situationen Proxy-Subskriptionen für Subskribenten erstellt:

1. Ein Warteschlangenmanager ist Mitglied einer Hierarchie und eine Proxy-Subskription wird an das übergeordnete Element und an die untergeordneten Elemente des Warteschlangenmanagers weitergeleitet.
2. Ein Warteschlangenmanager ist Mitglied eines Clusters und die Subskriptionsthemenzeichenfolge wird in ein Thema aufgelöst, das einem Cluster-Topic-Objekt zugeordnet ist. Bei einem *direkt weitergeleiteten* Cluster-Topic werden Proxy-Subskriptionen an alle Mitglieder des Clusters weitergeleitet. Bei einem über einen *Topic-Host weitergeleiteten* Cluster-Topic werden Proxy-Subskriptionen nur an die Warteschlangenmanager im Cluster weitergeleitet, auf denen das Cluster-Topic-Objekt definiert ist. Weitere Informationen hierzu finden Sie unter „Publish/Subscribe-Cluster” auf Seite 99.

Wenn ein Warteschlangenmanager Mitglied eines Clusters und einer Hierarchie ist, werden Proxy-Subskriptionen durch beide Mechanismen weitergeleitet, ohne das doppelte Veröffentlichungen an den Subskribenten übergeben werden.

In der folgenden Liste werden die Themenbereiche von drei Publish/Subscribe-Topologien beschrieben:

- „Fall 1. Publish/Subscribe-Cluster” auf Seite 113.
- „Fall 2. Publish/Subscribe-Hierarchien” auf Seite 114.

In separaten Abschnitten wird anhand der folgenden Konfigurationsaufgaben beschrieben, wie Themenbereiche zusammengefasst werden:

- Einzelnen Themenbereich in einem Publish/Subscribe-Cluster erstellen
- Themenbereiche mehrerer Cluster zusammenfassen
- Themenbereiche in mehreren Clustern zusammenfassen und isolieren
- Veröffentlichungen und Subskriptionen für Themenbereiche in mehreren Clustern durchführen

### Fall 1. Publish/Subscribe-Cluster

Im Beispiel wird vorausgesetzt, dass der Warteschlangenmanager nicht mit einer Publish/Subscribe-Hierarchie verbunden ist.

Wenn ein Warteschlangenmanager Mitglied eines Publish/Subscribe-Clusters ist, besteht sein Themenbereich aus lokalen Themen und Cluster-Topics. Lokale Themen sind Themenobjekten ohne das Attribut **CLUSTER** zugeordnet. Wenn ein Warteschlangenmanager über lokale Themenobjektdefinitionen verfügt, unterscheidet sich sein Themenbereich von dem eines anderen Warteschlangenmanagers im Cluster, der ebenfalls über eigene, lokal definierte Themenobjekte verfügt.

In einem Publish/Subscribe-Cluster können Sie kein Thema abonnieren, das auf einem anderen Warteschlangenmanager definiert ist, außer wenn das Thema, das Sie abonnieren, in ein Cluster-Topic-Objekt aufgelöst wird.

Wenn die gleichen benannten Definitionen eines Cluster-Topic-Objekts auf mehreren Warteschlangenmanagern erforderlich sind (z. B. bei Verwendung des *Topic-Host-Routing*, müssen alle Definitionen an den erforderlichen Stellen übereinstimmen. Weitere Informationen hierzu finden Sie im Abschnitt Einzelnen Themenbereich in einem Publish/Subscribe-Cluster erstellen.

Eine lokale Definition eines Themenobjekts, sei sie nun für ein Cluster-Topic oder ein lokales Thema, hat Vorrang vor demselben Themenobjekt, das anderswo im Cluster definiert ist. Es wird das lokal definierte Thema verwendet, auch wenn das anderswo definierte Thema jünger ist.

Es ist wichtig, dass ein Cluster-Topic-Objekt überall im Cluster derselben Themenzeichenfolge zugeordnet ist. Sie können die Themenzeichenfolge, der ein Themenobjekt zugeordnet ist, nicht ändern. Um dasselbe Themenobjekt einer anderen Themenzeichenfolge zuzuordnen, müssen Sie das Themenobjekt löschen und es mit der neuen Themenzeichenfolge erneut erstellen. Wenn das Thema zu einem Cluster gehört, bedeutet dies, dass die Kopien des Themenobjekts, die auf den anderen Mitgliedern des Clusters gespeichert sind, gelöscht und dann überall im Cluster Kopien des neuen Themenobjekts erstellt werden müssen. Die Kopien des Themenobjekts verweisen alle auf dieselbe Themenzeichenfolge.

Es kann vorkommen, dass versehentlich zwei Definitionen eines gleichnamigen Themenobjekts auf unterschiedlichen Warteschlangenmanagern im Cluster mit zwei unterschiedlichen Themenzeichenfolgen erstellt werden. Dies kann zu einem verwirrenden Verhalten führen, da mehrere Definitionen desselben Themenobjekts mit unterschiedlichen Themenzeichenfolgen je nachdem, wie und wo das Thema referenziert wird, zu unterschiedlichen Ergebnissen führen können. Weitere Informationen zu diesem wichtigen Punkt finden Sie im Abschnitt Mehrere Cluster-Topic-Definitionen mit gleichen Namen.

**Fall 2. Publish/Subscribe-Hierarchien**

Im Beispiel wird vorausgesetzt, dass der Warteschlangenmanager nicht Mitglied eines Publish/Subscribe-Clusters ist.

Wenn in IBM MQein Warteschlangenmanager Mitglied einer Publish/Subscribe-Hierarchie ist, besteht sein Topicbereich aus allen Themen, die lokal und auf verbundenen Warteschlangenmanagern definiert sind. Der Themenbereich ist bei allen Warteschlangenmanagern in einer Hierarchie derselbe. Es gibt keine Unterscheidung der Themen in lokale Themen und globale Themen.

Setzen Sie entweder die Option **PUBSCOPE** oder die Option **SUBSCOPE** auf `QMGR`, um zu verhindern, dass eine Veröffentlichung für ein Thema von einem Publisher an einen Subskribenten, der mit anderen Warteschlangenmanagern in der Hierarchie verbunden ist, übergeben wird.

Angenommen, Sie definieren ein Themenobjekt namens `Alabama` mit der Themenzeichenfolge `USA/Alabama` in Warteschlangenmanager `QMA`. Das würde zu folgenden Ergebnissen führen:

1. Der Themenbereich in `QMA` enthält jetzt auch das Themenobjekt `Alabama` und die Themenzeichenfolge `USA/Alabama`.
2. Eine Anwendung oder ein Administrator kann in `QMA` eine Subskription mit dem Themenobjektnamen `Alabama` erstellen.
3. Eine Anwendung kann in jedem Warteschlangenmanager in der Hierarchie eine Subskription für jedes Thema, einschließlich `USA/Alabama`, erstellen. Wenn `QMA` nicht lokal definiert wurde, wird das Thema `USA/Alabama` in das Themenobjekt `SYSTEM.BASE.TOPIC` aufgelöst.

**Zugehörige Konzepte**

„Veröffentlichungsumfang” auf Seite 111
In einem Publish/Subscribe-Cluster bzw. einer Publish/Subscribe-Hierarchie bestimmt auch der Bereich einer Veröffentlichung, ob Warteschlangenmanager die Veröffentlichung an ferne Warteschlangenmanager weiterleiten. Verwalten Sie den Bereich von Veröffentlichungen mit dem Themenattribut **PUBSCOPE**.

„Subskriptionsumfang” auf Seite 111
Der Bereich einer Subskription bestimmt, ob eine Subskription auf einem Warteschlangenmanager nur Veröffentlichungen von lokalen Publishern empfängt oder auch Veröffentlichungen, die auf einem anderen Warteschlangenmanager in einem Publish/Subscribe-Cluster oder einer Publish/Subscribe-Hierarchie erfolgen.

**Zugehörige Tasks**

Verteilte Publish/Subscribe-Netze konfigurieren

# IBM MQ Multicasting

IBM MQ Multicast bietet eine geringe Latenzzeit, eine hohe Ausgabefächerung und eine zuverlässige Multicasting-Nachrichtenübertragung.

Bei Multicast handelt es sich um eine effiziente Form der Publish/Subscribe-Nachrichtenübermittlung, da eine große Zahl von Subskribenten zugeordnet werden kann, ohne dass es zu negativen Auswirkungen beim Leistungsverhalten kommt. IBM MQ ermöglicht eine zuverlässige Multicasting-Nachrichtenübertragung, indem mithilfe von Empfangsbestätigungen, negativen Rückmeldungen und Folgenummern eine Nachrichtenübertragung mit geringer Latenzzeit und hoher Ausgabefächerung erzielt wird.

Die ausreichende Übermittlung von IBM MQ ermöglicht eine fast gleichzeitige Übermittlung, wodurch sichergestellt wird, dass kein Empfänger einen Vorteil hat. Da IBM MQ Multicast für die Übermittlung von Nachrichten ein Netz verwendet, wird für die Ausgabefächerung der Daten keine Publish/Subscribe-Engine benötigt. Nach der Zuordnung eines Themas zu einer Gruppenadresse ist kein Warteschlangenmanager erforderlich, da Publisher und Subskribenten in einem Peer-to-Peer-Modus arbeiten können. Dies ermöglicht die Reduzierung der Arbeitslast auf Warteschlangenmanager-Servern und der Warteschlangenmanager-Server ist keine potenzielle Fehlerquelle mehr.

## Ursprüngliche Multicasting-Konzepte

IBM MQ Multicast kann unter Verwendung des Kommunikationsinformationsobjekts COMMINFO problemlos in bestehende Systeme und Anwendungen integriert werden. Zwei TOPIC-Objektfelder ermöglichen die schnelle Konfiguration bestehender TOPIC-Objekte zur Unterstützung bzw. zum Ignorieren von Multicasting-Datenverkehr.

### Für Multicasting erforderliche Objekte

Nachfolgend finden Sie eine kurze Übersicht über die beiden Objekte, die für IBM MQ Multicast erforderlich sind:

**COMMINFO-Objekt**

Das COMMINFO-Objekt enthält die Attribute, die der Multicasting-Übertragung zugeordnet sind. Weitere Informationen zu den Parametern des Befehls COMMINFO finden Sie in DEFINE COMMINFO .

Das einzige COMMINFO-Feld, das angegeben werden MUSS, ist das Feld mit dem Namen des COMMINFO-Objekts. Mit diesem Namen wird das COMMINFO-Objekts für ein Thema ermittelt. Prüfen Sie das Feld **GRPADDR** des COMMINFO-Objekts, um sicherzustellen, dass der Wert eine gültige Multicastgruppenadresse ist.

**Themenobjekt**

Ein Thema ist der Gegenstand der Informationen, die in einer Publish/Subscribe-Nachricht veröffentlicht werden, und ein Thema wird definiert, indem ein TOPIC-Objekt erstellt wird. Weitere Informationen zu den Parametern des TOPIC-Objekts finden Sie unter DEFINE TOPIC.

Bestehende Themen können für das Multicasting verwendet werden, indem Sie die Werte der folgenden TOPIC-Objektparameter ändern: **COMMINFO** und **MCAST**.

- **COMMINFO** Dieser Parameter gibt den Namen des Multicastkommunikationsinformationsobjekts an.
- **MCAST** Dieser Parameter gibt an, ob Multicasting an dieser Position in der Themenstruktur zulässig ist. Standardmäßig ist **MCAST** auf ASPARENT gesetzt, d. h., das Multicasting-Attribut des Themas wird vom übergeordneten Element übernommen. Wenn Sie **MCAST** auf ENABLED setzen, ist auf diesem Knoten Multicasting-Datenverkehr möglich.

### Multicasting-Netze und -Themen

Nachfolgend finden Sie eine Übersicht darüber, was mit Subskriptionen mit unterschiedlichen Subskriptionstypen und Themendefinitionen geschieht. Alle folgenden Beispiele setzen voraus, dass der Parameter **COMMINFO** des TOPIC-Objekts auf den Namen eines gültigen COMMINFO-Objekts gesetzt wurde:

**Thema für Multicasting aktiviert (MCAST ENABLED)**
Wenn der Parameter der Themenzeichenfolge **MCAST** auf ENABLED gesetzt ist, sind Subskriptionen Multicasting-fähiger Clients zugelassen und eine Multicasting-Subskription wird vorgenommen – mit folgender Ausnahme:

- Es handelt sich um die permanente Subskription eines Multicasting-fähigen Clients.
- Es handelt sich um eine nicht verwaltete Subskription eines Multicasting-fähigen Clients.
- Es handelt sich um die Subskription eines nicht Multicasting-fähigen Clients.

In diesen Fällen wird eine andere Subskription als eine Multicast-Subskription vorgenommen und Subskriptionen werden in ein normales Publish/Subscribe herabgestuft.

**Thema für Multicasting inaktiviert**
Wenn der Parameter der Themenzeichenfolge **MCAST** auf DISABLED gesetzt ist, wird immer eine Nicht-Multicasting-Subskription vorgenommen und Subskriptionen werden in ein normales Publish/Subscribe herabgestuft.

**Thema nur für Multicasting aktiviert**
Wenn der Parameter der Themenzeichenfolge **MCAST** auf ONLY gesetzt ist, sind Subskriptionen Multicasting-fähiger Clients zugelassen und eine Multicasting-Subskription wird vorgenommen – mit folgender Ausnahme:

- Es handelt sich um eine permanente Subskription: Permanente Subskriptionen werden mit dem Ursachencode 2436 (0984) (RC2436): MQRC_DURABILITY_NOT_ALLOWED abgelehnt.
- Es handelt sich um eine nicht verwaltete Subskription: Nicht verwaltete Subskriptionen werden mit dem Ursachencode 2046 (07FE) (RC2046): MQRC_OPTIONS_ERROR abgelehnt.
- Es handelt sich um eine Subskription von einem nicht Multicast-fähigen Client: Diese Subskriptionen werden mit dem Ursachencode 2560 (0A00) (RC2560): MQRC_MULTICAST_ONLY abgelehnt.
- Es handelt sich um eine Subskription aus einer lokal gebundenen Anwendung: Diese Subskriptionen werden mit dem Ursachencode 2560 (0A00) (RC2560): MQRC_MULTICAST_ONLY abgelehnt.

## Windows Linux AIX Die Übersicht MQ Telemetry

MQ Telemetry umfasst einen Telemetrieservice (MQXR), der Bestandteil eines Warteschlangenmanagers ist, und Telemetrieclients, die Sie selbst schreiben oder kostenlos herunterladen können. Außerdem werden Befehlszeilenschnittstellen und Explorer-Verwaltungsschnittstellen zur Verfügung gestellt. Die Telemetrie bezeichnet die Erfassung von Daten aus zahlreichen fernen Einheiten und deren Verwaltung. Mit MQ Telemetry können Sie die Datensammlung und die Einheitensteuerung in Webanwendungen integrieren.

MQ Telemetry ist eine Komponente von IBM MQ. Ein Upgrade dieser Versionen besteht im Prinzip darin, eine höhere Version von IBM MQ zu installieren.

Beispielanwendungen sind über Eclipse Paho und MQTT.orgfrei verfügbar. Weitere Informationen finden Sie im Abschnitt IBM MQ Telemetry Transport-Beispielprogramme.

Da MQ Telemetry eine Komponente von IBM MQist, kann MQ Telemetry mit dem Hauptprodukt oder nach der Installation des Hauptprodukts installiert werden. Informationen zur Migration finden Sie unter Migration von MQ Telemetry unter Windows und Migration von MQ Telemetry unter Linux.

MQ Telemetry enthält folgende Komponenten:

**Telemetriekanäle**
Mithilfe von Telemetriekanälen können Sie die Verbindung zwischen MQTT-Clients und IBM MQ verwalten. Telemetriekanäle verwenden neue IBM MQ -Objekte, z. B. SYSTEM.MQTT.TRANSMIT.QUEUE, für die Interaktion mit IBM MQ.

**Telemetrieservice (MQXR)**
MQTT -Clients verwenden den Telemetrieservice `SYSTEM.MQXR.SERVICE` , um eine Verbindung zu Telemetriekanälen herzustellen.

**IBM MQ Explorer-Unterstützung für MQ Telemetry**
MQ Telemetry kann unter Verwendung von IBM MQ Explorer verwaltet werden.

**Dokumentation**
Die MQ Telemetry -Dokumentation ist in der IBM MQ -Standardproduktdokumentation enthalten. Die SDK-Dokumentation für Java und C-Clients wird in der Produktdokumentation sowie als Javadoc und im HTML-Format bereitgestellt.

## Telemetriekonzepte

Sie sammeln Informationen aus Ihrer Umgebung und entscheiden dann über Ihre Vorgehensweise. Als Verbraucher überprüfen Sie Ihre Vorräte und entscheiden dann, welche Lebensmittel Sie kaufen. Vor der Buchung einer Anschlussverbindung möchten Sie wissen, wie lange die Reise dauert, wenn Sie jetzt aufbrechen. Sie überprüfen Ihre Symptome und entscheiden dann, ob Sie einen Arzt konsultieren. Sie prüfen, wann ein Bus ankommen wird, und entscheiden dann, ob Sie warten. Die Informationen für diese Entscheidungen stammen direkt aus Zählern und Geräten, aus schriftlichen Unterlagen oder einem Bildschirm oder von Ihnen selbst. Sie sammeln Informationen, kombinieren diese, analysieren sie und handeln auf deren Grundlage - immer und überall.

Wenn die Informationsquellen weit verstreut oder nicht zugänglich sind, wird die Erfassung präzisester Informationen schwierig und aufwändig. Wenn Sie viele Änderungen vornehmen möchten oder es schwierig ist, diese Änderungen vorzunehmen, erfolgen keine Änderungen oder sie erfolgen an Stellen, an denen sie nicht so effektiv sind.

Was wäre, wenn sich der Aufwand der Erfassung von Informationen von den weit verstreuten Geräten und deren Steuerung erheblich verringern ließe, indem die Geräte mithilfe einer digitalen Technologie mit dem Internet verbunden würden? Die Informationen können mithilfe der Ressourcen des Internets und Unternehmens analysiert werden. Sie haben bessere Möglichkeiten für fundierte Entscheidungen und die Ergreifung der entsprechenden Maßnahmen.

Diese Änderungen werden durch technologische Trends und durch den umweltpolitischen und wirtschaftlichen Druck vorangetrieben:

1. Der Aufwand und die Kosten für die Verbindung und Steuerung von Sensoren und Aktuatoren können aufgrund der Standardisierung und Verbindung mit kostengünstigen digitalen Prozessoren reduziert werden.
2. Zur Verbindung von Geräten werden immer häufiger das Internet und Internettechnologien genutzt. In manchen Ländern werden für die Verbindung mit Internetanwendungen mehr Mobiltelefone als Personal Computer verwendet. Mit Sicherheit werden andere Geräte folgen.
3. Das Internet und die Internettechnologien erleichtern einer Anwendung den Abruf von Daten erheblich. Der schnelle Zugriff auf Daten fördert den Einsatz der Datenanalyse, damit Daten von Sensoren in Informationen umgewandelt werden können, die in zahlreichen anderen Lösungen hilfreich sind.
4. Der intelligente Einsatz von Ressourcen ist häufig der schnellere und kostengünstigere Weg zur Reduzierung von CO2-Emissionen und Kosten. Die Alternativen: Die Suche neuer Ressourcen oder die Entwicklung neuer Technologien zur Nutzung bestehender Ressourcen stellt vermutlich die langfristige Lösung dar. Im kurzfristigen Einsatz ist die Entwicklung neuer Technologien oder die Suche nach neuen Ressourcen oft riskanter, langsamer und teurer als die Verbesserung vorhandener Lösungen.

### Beispiel

Anhand eines Beispiels soll veranschaulicht werden, wie diese Trends neue Gelegenheiten für eine intelligente Interaktion mit der Umwelt bieten.

Das internationale Übereinkommen zum Schutz des menschlichen Lebens auf See (International Convention for the Safety of Life at Sea (SOLAS)) verlangt auf vielen Schiffen den Einsatz eines automatischen Identifikationssystems (AIS). Dieses System ist auf Handelsschiffen mit über 300 Tonnen und auf Passa-

gierschiffen erforderlich. AIS ist in erster Linie ein System zur Kollisionsvermeidung in der Küstenschifffahrt. Es wird von Schifffahrtsbehörden zur Überwachung und Kontrolle von Küstengewässern eingesetzt.

Enthusiasten auf der ganzen Welt implementieren kostengünstige AIS-Überwachungsstationen und stellen Informationen der Küstenschifffahrt in das Internet. Andere begeisterte Anhänger schreiben Anwendungen, die die Informationen aus dem AIS mit anderen Informationen aus dem Internet kombinieren. Die Ergebnisse werden auf Websites gestellt und mithilfe von Twitter und SMS veröffentlicht.

In einer Anwendung werden beispielsweise Informationen der AIS-Stationen in der Nähe von Southampton mit Informationen zum Schiffseigner und geografischen Daten kombiniert. Die Anwendung speist die Echtzeitdaten zu Ankunfts- und Abfahrtszeiten von Fähren in Twitter ein. Regelmäßige Pendler, die die Fähren zwischen Southampton und der Isle of Wight benutzen, abonnieren den Newsfeed mittels Twitter oder SMS. Wenn der Feed eine Verspätung der Fähre anzeigt, können sich Pendler entsprechend später auf den Weg machen und die Fähre pünktlich erreichen, obwohl diese nicht zur planmäßigen Ankunftszeit anlegt.

Weitere Beispiele finden Sie im Abschnitt „Telemetrieanwendungsfälle” auf Seite 119.

**Zugehörige Tasks**

Installieren von MQ Telemetry
MQ Telemetry verwalten
MQ Telemetry unter Windows migrieren
MQ Telemetry unter Linux migrieren
Anwendungen für MQ Telemetry entwickeln
Fehlerbehebung bei Problemen mit MQ Telemetry

**Zugehörige Verweise**

MQ Telemetry Referenz

Windows Linux AIX

# Einführung in MQ Telemetry

Privatpersonen, Unternehmen und Regierungen interessieren sich mehr und mehr für die Verwendung von MQ Telemetry, um schnell mit dem Lebens- und Arbeitsumfeld interagieren zu können. MQ Telemetry verbindet alle Arten von Geräten mit dem Internet und Unternehmen, wodurch sich die Kosten für die Erstellung von Anwendungen für Smart Devices verringern.

## Was ist MQ Telemetry?

- Diese Funktion von IBM MQ erweitert die universelle Messaging-Zentralverbindung, die IBM MQ einer Vielzahl von fernen Sensoren, Aktuatoren und Telemetriegeräten bereitstellt. MQ Telemetry erweitert IBM MQ, sodass intelligente Unternehmensanwendungen, Services und Entscheidungsträger mit Netzen aus instrumentierten Geräten verbunden werden können.
- Die Kernkomponenten von MQ Telemetry sind:

  **MQ Telemetry-Service (MQXR)**
  Dieser Service läuft auf dem IBM MQ-Server; er verwendet zur Kommunikation mit Telemetriegeräten das IBM MQ Telemetry Transport-(MQTT-)Protokoll.

  **MQTT-Anwendungen, die Sie schreiben**
  Diese Anwendungen steuern die Informationen, die zwischen den Telemetriegeräten und dem IBM MQ-Warteschlangenmanager übertragen werden, sowie alle Aktionen, die als Reaktion auf diese Informationen durchgeführt werden. Bei der Erstellung dieser Anwendungen helfen Ihnen die MQTT-Clientbibliotheken.

## Was kann die Software für mich tun?

- MQTT ist ein offenes Messaging-Protokoll, mit dem MQTT-Implementierungen für eine Vielzahl von Geräten erstellt werden können.
- MQTT-Clients können auch auf Geräten mit geringem Speicher und begrenzten Ressourcen ausgeführt werden.

- MQTT kann auf effiziente Weise in Netzen mit niedriger Bandbreite eingesetzt werden, bei denen das Versenden von Daten kostenintensiv ist oder die störanfällig sind.
- Die Nachrichtenübermittlung wird von der Anwendung sichergestellt und voneinander entkoppelt.
- Anwendungsprogrammierer müssen dazu keine Kenntnisse über das Programmieren von Kommunikationsroutinen besitzen.
- Nachrichten können mit anderen Messaging-Anwendungen ausgetauscht werden, zum Beispiel mit anderen Telemetrieanwendungen oder einer MQI-, JMS- oder Unternehmens-Messaging-Anwendung.

### Wozu wende ich die Software an?

- Es werden Beispielscripts bereitgestellt, die mit einem Beispiel für eine Clientanwendung von IBM MQ Telemetry Transport der Version 3 arbeiten (`mqttv3app.jar`). Siehe IBM MQ Telemetry Transport-Beispielprogramme.
- Mit dem IBM MQ Explorer und seinen Tools können Sie die Telemetriefunktion von IBM MQ verwalten.
- Mit den Clientbibliotheken können Sie MQTT-Anwendungen erstellen, die eine Verbindung zu einem Warteschlangenmanager herstellen und Publish/Subscribe-Messaging verwenden.
- Stellen Sie Ihre Anwendung und die Clientbibliothek auf dem Gerät bereit, auf dem die Anwendung ausgeführt werden soll.

### Wie funktioniert die Software?

- MQTT ist ein Publish/Subscribe-Protokoll. Eine MQTT-Clientanwendung kann Nachrichten an einen MQTT-Server veröffentlichen oder Nachrichten abonnieren, die von Anwendungen gesendet werden, die mit einem MQTT-Server verbunden sind.
- MQTT-Clientanwendungen verwenden Clientbibliotheken, die das MQTT-Messaging-Protokoll implementieren.
- Eine einfache MQTT-Clientanwendung funktioniert wie ein MQ-Standardclient, kann jedoch auf wesentlich mehr verschiedenen Plattformen und in deutlich mehr Netzen ausgeführt werden.
- Mit dem MQ Telemetry-Service (MQXR) wird aus einem IBM MQ-Warteschlangenmanager ein MQTT-Server.
- Wenn ein IBM MQ-Warteschlangenmanager als MQTT-Server agiert, können andere Anwendungen, die sich mit dem Warteschlangenmanager verbinden, die Nachrichten vom MQTT-Client abonnieren und empfangen.
- Der Warteschlangenmanager agiert als Router, der Nachrichten von veröffentlichenden Anwendungen an abonnierende Anwendungen verteilt.
- Die Nachrichten können zwischen verschiedenen Arten von Clientanwendungen verteilt werden. Dies ist beispielsweise zwischen Telemetrieclients und JMS-Clients möglich.

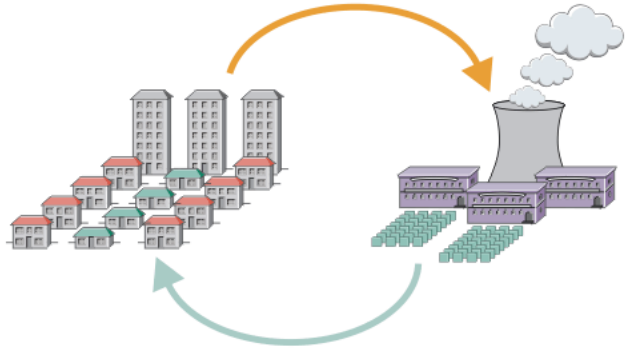
**Anmerkung:** MQ Telemetry ersetzt die SCADA-Knoten, die aus Version 7 von WebSphere Message Broker (jetzt "IBM Integration Bus") entfernt wurden, und wird unter Windows, Linux und AIX ausgeführt.

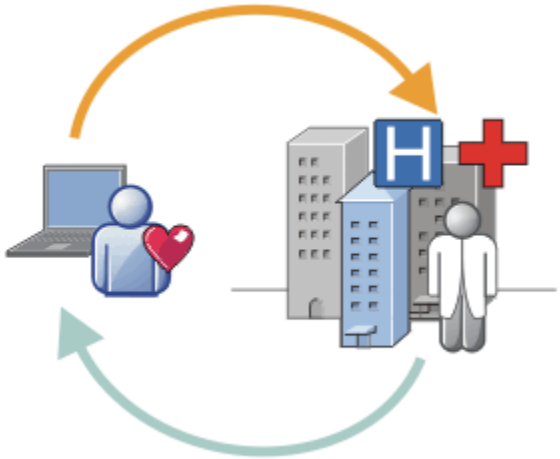
## Windows Linux AIX Telemetrieanwendungsfälle

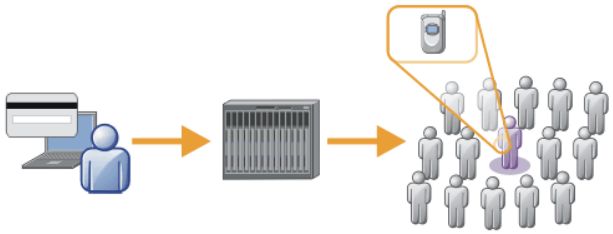
Telemetrie ist die automatisierte Erkennung und Messung von Daten ferner Geräte sowie die Steuerung dieser fernen Geräte. Der Schwerpunkt liegt in der Datenübertragung von fernen Geräten zu einem zentralen Kontrollpunkt. Die Telemetrie beinhaltet auch das Senden von Konfigurations- und Steuerinformationen an Einheiten.

MQ Telemetry verbindet kompakte Endgeräte über das MQTT protocol. Die Geräte werden unter Verwendung von IBM MQ mit anderen Anwendungen verbunden. MQ Telemetry bildet eine Brücke zwischen den Geräten und dem Internet und vereinfacht so die Erstellung von "smarten Lösungen". Smarte Lösungen ermöglichen Anwendungen, die Geräte überwachen und steuern, den Zugriff auf die Fülle von Informationen im Internet und in Unternehmensanwendungen.

Die folgenden Diagramme zeigen Beispiele zur typischen Verwendung von MQ Telemetry:

| Telemetry: Smart Electricity | |
|---|---|
|  | • MQTT-Nachricht mit den Energieverbrauchsdaten, die an den Service-Provider gesendet werden<br>• MQ Telemetry sendet Steuerbefehle auf Grundlage der analysierten Energieverbrauchsdaten.<br>• Weitere Informationen finden Sie im folgenden Anwendungsfall: „Anwendungsfall für Telemetry: Überwachung und Steuerung des Energieverbrauchs in Privathaushalten" auf Seite 122 |

| Telemetry: Smart Health Services | |
|---|---|
| • MQ Telemetry sendet Gesundheitsdaten an Ihr Krankenhaus & Ihren Arzt.<br>• MQTT-Rückmeldungen mit Warnungen oder Feedback können auf Basis der analysierten medizinischen Daten gesendet werden.<br>• Weitere Informationen finden Sie im folgenden Anwendungsfall: „Anwendungsfall für Telemetry: Überwachung nicht stationärer Patienten" auf Seite 120 |  |

| Telemetry: One in a Crowd | |
|---|---|
|  | • Eine einfache Kartentransaktion wird an den Server einer Bank gesendet.<br>• MQ Telemetry ermittelt die betreffende Kontaktperson und warnt den Kunden, dass seine Karte verwendet wurde.<br>• MQ Telemetry kann mithilfe einfachster Eingabeinformationen die betreffende Kontaktperson ausfindig machen. |

Die nachfolgenden Beispiele, die auf realen Anwendungsfällen beruhen, veranschaulichen einige Einsatzmöglichkeiten von Telemetry und zeigen einige der häufig auftretenden Probleme auf, die von der Telemetrietechnologie gelöst werden müssen.

## Windows Linux AIX Anwendungsfall für Telemetry: Überwachung nicht stationärer Patienten

Im Rahmen der Zusammenarbeit zwischen IBM und einem Anbieter des Gesundheitswesens bei einem Betreuungssystem für Herzpatienten kommuniziert ein implantierter Kardioverter-Defillibrator mit einem Krankenhaus. Die Daten des Patienten und implantierten Geräts werden mithilfe der RF-Telemetrie (RF = Radiofrequenz) an das MQTT-Gerät in der Wohnung des Patienten übertragen.

Für gewöhnlich erfolgt die Übertragung nachts an einen Sender, der sich neben dem Bett befindet. Der Sender überträgt die Daten auf sichere Weise über die Telefonanlage an das Krankenhaus, wo die Daten analysiert werden.

Durch dieses System verringert sich die Anzahl der Besuche des Patienten beim Arzt. Es erkennt, wenn Probleme beim Patienten oder Gerät auftreten, und benachrichtigt bei einem Notfall den diensthabenden Arzt.

Die Zusammenarbeit zwischen IBM und dem Anbieter des Gesundheitswesens weist Merkmale auf, die in verschiedenen Anwendungsfällen für Telemetry typisch sind:

**Unsichtbarkeit**

Für die Nutzung des Geräts muss der Anwender lediglich Strom und eine Telefonverbindung bereitstellen und sich zu bestimmten Tageszeiten in der Nähe des Geräts befinden. Ansonsten ist keinerlei Benutzereingriff erforderlich. Der Betrieb ist zuverlässig und die Bedienung einfach.

Damit der Patient das Gerät nicht einrichten muss, wird es vom Gerätelieferanten bereits vorkonfiguriert. Der Patient muss das Gerät lediglich einstecken. Dadurch, dass die Konfiguration nicht durch den Patienten erfolgt, wird der Gerätebetrieb vereinfacht und die Gefahr einer falschen Konfiguration verringert sich.

Der MQTT-Client ist in das Gerät integriert. Der Geräteentwickler integriert die MQTT-Clientimplementierung in das Gerät und der Entwickler bzw. Anbieter konfiguriert den MQTT-Client im Rahmen der Vorkonfiguration.

Der MQTT-Client wird als Java SE-JAR-Datei ausgeliefert. Diese JAR-Datei wird vom Entwickler in die zugehörige Java-Anwendung aufgenommen. In Umgebungen ohne Java (wie in diesem Fall) kann der Geräteentwickler einen Client unter Verwendung der veröffentlichten MQTT-Formate und des MQTT-Protokolls in einer anderen Sprache implementieren. Der Entwickler kann auch einen der Clients in der Programmiersprache C verwenden, die als gemeinsam genutzte Bibliotheken für Windows-, Linux- und ARM-Plattformen ausgeliefert werden.

**Ungleichmäßige Konnektivität**

Die Kommunikation zwischen dem Defibrillator und dem Krankenhaus ist mit den Herausforderungen einer ungleichmäßigen Konnektivität konfrontiert. Zwei verschiedene Netze werden zur Lösung der unterschiedlichen Problemstellung der Datenerfassung beim Patienten und der Datenübertragung an das Krankenhaus genutzt. Zwischen dem Patienten und dem MQTT-Gerät wird ein RF-Nahbereichsnetz mit geringer Leistung verwendet. Der Sender verbindet sich unter Verwendung einer TCP/IP-Verbindung im virtuellen privaten Netz über eine Telefonleitung mit geringer Bandbreite mit dem Krankenhaus.

Die direkte Verbindung jedes Geräts mit einem IP-Netz gestaltet sich häufig als aufwändig. Die Verwendung von zwei Netzen, die über einen Hub miteinander verbunden sind, ist eine bewährte Lösung für dieses Problem. Das MQTT-Gerät ist ein einfacher Hub, der Informationen des Patienten speichert und diese an das Krankenhaus weiterleitet.

**Sicherheit**

Der Arzt muss sich auf die Authentizität der Patientendaten verlassen können, während der Patient die Vertraulichkeit seiner Daten wünscht.

In einigen Anwendungsfällen reicht es aus, die Verbindung über ein virtuelles privates Netz (VPN) oder TLS zu verschlüsseln. In anderen Anwendungsfällen müssen die Daten auch nach ihrer Abspeicherung sicher aufbewahrt werden.

Gelegentlich lässt die Sicherheit des Telemetriegeräts zu wünschen übrig. Es könnte sich beispielsweise in einer Gemeinschaftsunterkunft befinden. Der Benutzer des Geräts muss authentifiziert werden, damit die Daten auf jeden Fall vom richtigen Patienten stammen. Das Gerät selbst kann über TLS beim Server authentifiziert werden. Die Authentifizierung in umgekehrter Richtung erfolgt ebenso.

Der Telemetriekanal zwischen dem Gerät und dem Warteschlangenmanager unterstützt JAAS (Java Authentication and Authorization Service) für die Benutzerauthentifizierung und TLS für die Verschlüs-

selung der Kommunikation sowie die Geräteauthentifizierung. Der Zugriff auf eine Veröffentlichung wird durch den Objektberechtigungsmanager in IBM MQ gesteuert.

Die Kennung, die für die Authentifizierung des Benutzers verwendet wird, kann einer anderen Kennung zugeordnet werden - beispielsweise einer allgemeinen Patientenidentität. Eine allgemeine Kennung vereinfacht die Konfiguration der Autorisierung für Veröffentlichungsthemen in IBM MQ.

**Konnektivität**

Der Verbindungsaufbau zwischen dem MQTT-Gerät und dem Krankenhaus erfolgt über eine Anwahl, wobei eine so niedrige Bandbreite wie 300 Baud genutzt werden kann.

Damit auch bei 300 Baud ein effektiver Betrieb möglich ist, fügt das MQTT protocol neben den TCP/IP-Headern lediglich eine geringe Anzahl an zusätzlichen Bytes zu einer Nachricht hinzu.

Das MQTT protocol bietet eine Nachrichtenübermittlung auf Basis des *fire-and-forget*-Prinzips mit einer Einzelübertragung, was die Latenzzeiten niedrig hält. Es kann aber auch mehrere Übertragungen verwenden, um die Zustellung des Typs *mindestens einmal* und *genau einmal* zu ermöglichen. Dies empfiehlt sich in Fällen, bei denen eine garantierte Übermittlung wichtiger als die Reaktionszeit ist. Zur Gewährleistung der Übermittlung werden die Nachrichten im Gerät gespeichert, bis sie erfolgreich zugestellt wurden. Wenn ein Gerät drahtlos verbunden ist, ist die garantierte Übermittlung besonders sinnvoll.

**Skalierbarkeit**

Telemetriegeräte werden für gewöhnlich in einer hohen Zahl implementiert - dies reicht von Zehntausenden bis hin zu Millionen.

Die Verbindung vieler Geräte mit einem System stellt hohe Anforderungen an die Lösung. Dies sind wirtschaftliche Anforderungen wie die Kosten für Geräte und die zugehörige Software und administrative Anforderungen wie die Verwaltung der Lizenzen, Geräte und Benutzer. Technische Anforderungen umfassen die Netz- und Serverauslastung.

Das Öffnen von Verbindungen belegt mehr Serverressourcen, als die Verbindungen offen zu halten. In einem Anwendungsfall wie dem vorliegenden, bei dem Telefonleitungen genutzt werden, bringen die Ausgaben für die Telefonverbindungen mit sich, dass die Verbindungen nur so lange offen bleiben, wie sie benötigt werden. Die Datenübertragungen erfolgen vorwiegend im Stapelbetrieb. Die Verbindungen können für die gesamte Nacht terminiert werden, um eine plötzliche Verbindungsspitze vor dem Schlafengehen zu vermeiden.

Auf dem Client wird die Skalierbarkeit von Clients durch den minimalen Aufwand für die Clientkonfiguration gefördert. Der MQTT-Client ist in das Gerät integriert. Es müssen keine Konfigurationsschritte oder Schritte zum Akzeptieren der MQTT-Clientlizenz zur Bereitstellung der Geräte für Patienten eingebunden werden.

Auf dem Server hat MQ Telemetry ein Anfangsziel von 50.000 offenen Verbindungen je Warteschlangenmanager.

Die Verbindungen werden mithilfe von IBM MQ Explorer verwaltet. Der IBM MQ Explorer filtert die anzuzeigenden Verbindungen auf eine Anzahl, deren Verwaltung bewältigt werden kann. Durch die geeignete Wahl eines Schemas, bei dem den Clients Kennungen zugeordnet werden, können Sie Verbindungen auf Basis der geografischen Lage oder alphabetisch nach Patientennamen filtern.

## Windows Linux AIX Anwendungsfall für Telemetry: Überwachung und Steuerung des Energieverbrauchs in Privathaushalten

Intelligente Zähler erfassen mehr Einzeldaten zum Energieverbrauch als herkömmliche Messgeräte.

Intelligente Zähler sind häufig mit einem lokalen Telemetrienetz verbunden, um einzelne Geräte in einem Privathaushalt zu überwachen und zu steuern. Einige davon sind auch remote zur Fernüberwachung und -steuerung verbunden.

Die Fernverbindung kann von einer Einzelperson, einem Elektrizitätswerk oder einer zentralen Kontrollstelle eingerichtet werden. Die ferne Kontrollstelle kann den Leistungsverbrauch ablesen und Nutzungs-

daten bereitstellen. Sie kann Daten bereitstellen, die die Nutzung beeinflussen, beispielsweise fortlaufende Preis- und Wetterdaten. Durch eine Lastbegrenzung kann die gesamte Stromerzeugungseffizienz gesteigert werden.

Die Implementierung intelligenter Zähler wird immer beliebter. Die britische Regierung steht beispielsweise in Verhandlungen über die Implementierung intelligenter Zähler in allen britischen Haushalten bis zum Jahr 2020.

Die Anwendungsfälle für die Messung in Privathaushalten weisen eine Reihe allgemeiner Merkmale auf:

**Unsichtbarkeit**

Das Messgerät erfordert keinen Benutzereingriff, es sei denn, der Benutzer möchte aktiv in die Energieersparnis eingreifen, die durch die Verwendung des Messgeräts ermöglicht wird. Es darf die Zuverlässigkeit der Energieversorgung einzelner Systeme nicht beeinträchtigen.

Ein MQTT-Client kann in die Software eingebettet werden, die mit dem Messgerät implementiert wird, und erfordert keine separate Installation oder Konfiguration.

**Ungleichmäßige Konnektivität**

Die Datenübertragung zwischen Geräten und dem intelligenten Zähler erfordert andere Konnektivitätsstandards als die Datenübertragung zwischen dem Zähler und dem Fernverbindungspunkt.

Die Verbindung vom intelligenten Zähler zu den Geräten muss hoch verfügbar sein und den Netzstandards für ein Heimnetz entsprechen.

Das ferne Netz nutzt wahrscheinlich verschiedene physische Verbindungen. Einige davon, beispielsweise Mobiltelefone, weisen hohe Übertragungskosten auf und sind möglicherweise nicht unterbrechungsfrei. Die MQTT v3-Spezifikation wurde für Fernverbindungen und Verbindungen zwischen lokalen Adaptern und dem intelligenten Zähler konzipiert.

Für die Verbindung zwischen den Netzsteckdosen und Geräten und dem Messgerät wird ein Heimnetz wie beispielsweise Zigbee verwendet. MQTT für Sensorennetze (MQTT-S) wurde für die Arbeit mit Zigbee und anderen Netzprotokollen mit geringer Bandbreite entwickelt. MQ Telemetry unterstützt MQTT-S nicht direkt. Es wird ein Gateway für die Verbindung von MQTT-S mit MQTT v3 benötigt.

Wie bei der Überwachung nicht stationärer Patienten erfordern auch die Lösungen für die Überwachung und Steuerung des Energieverbrauchs in Privathaushalten mehrere Netze, die unter Verwendung des intelligenten Zählers als Hub miteinander verbunden sind.

**Sicherheit**

Im Zusammenhang mit intelligenten Zählern bestehen einige Sicherheitsprobleme. Dazu gehören der fälschungssichere Herkunftsnachweis von Transaktionen, die Autorisierung eingeleiteter Steuerungsvorgänge und der Datenschutz der Energieverbrauchswerte.

Zur Gewährleistung des Datenschutzes können die Daten, die von MQTT zwischen dem Messgerät und der fernen Kontrollstelle übertragen werden, mithilfe von Transport Layer Security (TLS) verschlüsselt werden. Zur Sicherstellung, dass Steuerungsvorgänge tatsächlich autorisiert sind, kann die MQTT-Verbindung zwischen dem Messgerät und der fernen Kontrollstelle gegenseitig unter Verwendung von TLS authentifiziert werden.

**Konnektivität**

Die physische Spezifik des fernen Netzes kann beträchtlich variieren. Sie kann beispielsweise eine bestehende Breitbandverbindung oder ein Mobilnetz mit hohen Verbindungskosten und einer nicht unterbrechungsfreien Verfügbarkeit nutzen. Bei teuren, nicht unterbrechungsfreien Verbindungen ist MQTT ein effizientes und zuverlässiges Protokoll; siehe Abschnitt „Anwendungsfall für Telemetry: Überwachung nicht stationärer Patienten” auf Seite 120.

**Skalierbarkeit**

Energieversorgungsunternehmen oder zentrale Kontrollstellen planen die Implementierung von letztendlich Zig-Millionen intelligenter Zähler. Anfänglich bewegt sich die Anzahl der Messgeräte je Implementierung zwischen Zehn- und Hunderttausenden. Diese Zahl ist vergleichbar mit dem anfänglichen MQTT-Ziel von 50.000 offenen Clientverbindungen je Warteschlangenmanager.

Ein kritischer Aspekt in der Architektur bei der Überwachung und Steuerung des Energieverbrauchs in Privathaushalten ist die Verwendung des intelligenten Zählers als Netzkonzentrator. Jeder Geräteadapter ist ein separater Sensor. Durch deren Verbindung mit einem lokalen Hub unter Verwendung von MQTT kann der Hub die Datenflüsse in einer einzelnen TCP/IP-Sitzung mit der zentralen Kontrollstelle konzentrieren und außerdem die Nachrichten für einen kurzen Zeitraum speichern, um Sitzungsausfallzeiten entgegenzuwirken.

Fernverbindungen müssen in den Anwendungsfällen für die Energieverbrauchssteuerung in Privathaushalten aus zwei Gründen offen gehalten werden. Der erste Grund ist, dass das Öffnen von Verbindungen im Vergleich mit dem Senden von Anforderungen viel Zeit in Anspruch nimmt. Der zeitliche Aufwand für das Öffnen vieler Verbindungen zum Senden von Anforderungen zur "Lastbegrenzung" in einem kurzen Zeitraum ist zu hoch. Der zweite Grund ist, dass die Verbindung für den Empfang von Anforderungen zur Lastbegrenzung vom Energieversorgungsunternehmen zuerst vom Client geöffnet werden muss. Bei MQTT werden Verbindungen immer durch den Client aufgebaut und für den Empfang von Anforderungen zur Lastbegrenzung vom Energieversorgungsunternehmen muss die Verbindung offen gehalten werden.

Wenn die Geschwindigkeit des Öffnens von Verbindungen kritisch ist oder der Server zeitkritische Anforderungen initialisiert, unterhält die Lösung für gewöhnlich viele offene Verbindungen.

## Windows Linux AIX Anwendungsfälle für Telemetry: Radiofrequenzidentifikation (RFID)

RFID ist die Verwendung eines eingebetteten RFID-Tags zur Identifizierung und drahtlosen Überwachung eines Objekts. RFID-Tags können auf eine Entfernung von bis zu mehreren Metern gelesen werden und können sich außerhalb der Sichtweite der RFID-Leseeinheit befinden. Passive Tags werden von einer RFID-Leseeinheit aktiviert. Aktive Tags übertragen Daten ohne externe Aktivierung. Aktive Tags müssen über einen Versorgungsstromkreis verfügen. Passive Tags können einen Versorgungsstromkreis nutzen, um ihre Reichweite zu erhöhen.

RFID wird in vielen Anwendungen verwendet und die Anwendungsfälle sind vielfältig. RFID-Anwendungsfälle und die Anwendungsfälle für die Überwachung nicht stationärer Patienten sowie die Überwachung und Steuerung des Energieverbrauchs in Privathaushalten weisen einige Gemeinsamkeiten, aber auch Unterschiede auf.

**Unsichtbarkeit**

In vielen Anwendungsfällen werden die RFID-Leseeinheiten in hoher Zahl implementiert und müssen ohne Benutzereingriff funktionieren. Die Leseeinheit enthält einen integrierten MQTT-Client für die Kommunikation mit einem zentralen Kontrollpunkt.

In einem Vertriebslager verwendet die Leseeinheit beispielsweise einen Bewegungssensor zur Ortung einer Palette. Sie aktiviert die RFID-Tags der Artikel auf der Palette und sendet Daten und Anforderungen an zentrale Anwendungen. Mithilfe der Daten wird die Position des Warenbestands aktualisiert. Die Anforderungen steuern, was mit der Palette im nächsten Schritt passieren soll; sie könnte beispielsweise an eine bestimmte Position verschoben werden. Fluglinien und Gepäcksysteme in Flughäfen nutzen RFID beispielsweise auf diese Weise.

In einigen RFID-Anwendungsfällen verfügt die Leseeinheit über eine standardmäßige Datenverarbeitungsumgebung wie Java Platform, Micro Edition (Java ME). In diesen Fällen kann der MQTT-Client nach der Herstellung in einem eigenen Konfigurationsschritt implementiert werden.

**Ungleichmäßige Konnektivität**

Die RFID-Leseeinheiten können getrennt vom lokalen Steuergerät sein, das einen MQTT-Client enthält, oder jede Leseeinheit kann über einen integrierten MQTT-Client verfügen. Für gewöhnlich wird die gewählte Topologie von geografischen oder kommunikationstechnischen Faktoren bestimmt.

**Sicherheit**

Bei der Anlage von RFID-Tags stellen der Datenschutz und die Authentizität ein Sicherheitsproblem dar. RFID-Tags sind unauffällig und können heimlich überwacht, verfälscht oder manipuliert werden.

Die Lösung der RFID-Sicherheitsprobleme schafft Raum für die Implementierung neuer RFID-Lösungen. Auch wenn das Sicherheitsrisiko im RFID-Tag und in der lokalen Leseeinheit liegt, kann die Verarbeitung zentraler Daten Möglichkeiten für die Reaktion auf verschiedene Sicherheitsbedrohungen bieten. Eine Tag-Manipulation kann beispielsweise erkannt werden, indem der Warenbestand dynamisch mit Lieferungen und dem Versand korreliert wird.

**Konnektivität**

In RFID-Anwendungen waren für gewöhnlich sowohl stapelorientierte Store-and-forward-Verfahren für erfasste Informationen, die aus den RFID-Leseeinheiten stammen, als auch sofortige Abfragen involviert. Im Anwendungsfall des Vertriebslagers ist die RFID-Leseeinheit ständig verbunden. Wenn ein Tag gelesen wird, wird er gemeinsam mit Informationen zur Leseeinheit veröffentlicht. Die Warehousing-Anwendung veröffentlicht die Antwort wiederum bei der Leseeinheit.

In der Warehousing-Anwendung ist das Netz normalerweise zuverlässig und die sofortigen Anforderungen verwenden unter Umständen zur Verkürzung der Latenzzeit Nachrichten des Typs *fire and forget*. Die Daten des stapelorientierten Store-and-forward-Verfahrens nutzen möglicherweise eine Nachrichtenübertragung nach dem Prinzip *genau einmal*, um den Verwaltungsaufwand zu minimieren, der mit einem Datenverlust verbunden ist.

**Skalierbarkeit**

Wenn die RFID-Anwendung sofortige Antworten benötigt, also innerhalb von ein oder zwei Sekunden, müssen die RFID-Leseeinheiten verbunden bleiben.

## Windows Linux AIX Anwendungsfälle für Telemetry: Erfassung von Umweltdaten

Bei der Erfassung von Umweltdaten werden mithilfe der Telemetrie Informationen zum Wasserstand und zur Wasserqualität von Flüssen, zu Luftschadstoffen und zu sonstigen Umweltdaten gesammelt.

Die Sensoren befinden sich häufig an fernen Orten ohne Zugriff auf eine festnetzgebundene Kommunikation. Die drahtlose Bandbreite ist kostenintensiv und die Zuverlässigkeit kann gering sein. Für gewöhnlich sind mehrere Umgebungssensoren in einem kleinen geografischen Bereich mit einem lokalen Überwachungsgerät an einem sicheren Ort verbunden. Die lokalen Verbindungen können festnetzgebunden oder drahtlos sein.

**Unsichtbarkeit**

Die Sensoreinheiten sind in der Regel weniger zugänglich, haben eine niedrigere Leistung und werden in größerer Zahl implementiert als das zentrale Überwachungsgerät. Die Sensoren selbst sind gelegentlich "unintelligent" und das lokale Überwachungsgerät enthält Adapter für die Umwandlung und Speicherung der Sensordaten. Das Überwachungsgerät umfasst häufig einen Mehrzweckcomputer, der Java Platform, Standard Edition (Java SE) oder Java Platform, Micro Edition (Java ME) unterstützt. Bei der Konfiguration des MQTT-Clients spielt die Unsichtbarkeit für gewöhnlich eine untergeordnete Rolle.

**Ungleichmäßige Konnektivität**

Das Leistungsspektrum von Sensoren sowie die Kosten und Bandbreite der Fernverbindung führen für gewöhnlich dazu, dass ein lokaler Überwachungshub mit einem zentralen Server verbunden wird.

**Sicherheit**

Die Sicherheit spielt eine untergeordnete Rolle, es sei denn, die Lösung wird in einem militärischen Umfeld oder Verteidigungsszenario verwendet.

**Konnektivität**

Viele Einsatzgebiete erfordern keine kontinuierliche Überwachung oder sofortige Datenverfügbarkeit. Ausnahmedaten wie beispielsweise eine Warnung vor hohem Wasserstand müssen hingegen sofort weitergeleitet werden. Die Sensordaten werden beim lokalen Überwachungsprogramm gesammelt, um Verbindungs- und Kommunikationskosten zu reduzieren, und dann in einem festgelegten Zeitrahmen über Verbindungen übertragen. Ausnahmedaten werden direkt nach ihrer Ermittlung beim Überwachungsprogramm weitergeleitet.

**Skalierbarkeit**

Sensoren konzentrieren sich um lokale Hubs und die Sensordaten werden in Paketen gesammelt, die nach einem festgelegten Zeitplan übertragen werden. Diese Faktoren reduzieren die Arbeitslast auf dem zentralen Server, die bei der Verwendung von direkt verbundenen Sensoren bestehen würde.

## Windows Linux AIX Anwendungsfälle für Telemetry: Mobile Anwendungen

Mobile Anwendungen sind Anwendungen, die auf mobilen Endgeräten ausgeführt werden. Bei den Geräten handelt es sich entweder um generische Anwendungsplattformen oder kundenspezifische Geräte.

Allgemeine Plattformen umfassen mobile Endgeräte wie Telefone und elektronische Organizer sowie mobile Geräte wie Notebook-Computer. Kundenspezifische Geräte verwenden spezielle Hardware, die auf bestimmte Anwendungen zugeschnitten ist. Ein Gerät für die Erfassung einer Paketzustellung "mit Unterschrift" ist ein Beispiel für ein solches kundenspezifisches mobiles Gerät. Anwendungen auf kundenspezifischen mobilen Geräten basieren häufig auf einer generischen Softwareplattform.

**Unsichtbarkeit**

Die Implementierung kundenspezifischer mobiler Anwendungen wird verwaltet und kann die Konfiguration der MQTT-Clientanwendung beinhalten. Bei der Konfiguration des MQTT-Clients spielt die Unsichtbarkeit für gewöhnlich eine untergeordnete Rolle.

**Ungleichmäßige Konnektivität**

Im Gegensatz zu der lokalen Hub-Topologie der vorherigen Anwendungsfälle verbinden sich mobile Clients über Fernzugriff. Die Clientanwendungsschicht verbindet sich direkt mit einer Anwendung am zentralen Hub.

**Sicherheit**

Da nur wenig physische Sicherheit besteht, müssen das mobile Gerät und der mobile Benutzer authentifiziert werden. Zur Überprüfung der Identität des Geräts wird TLS verwendet, während die Benutzerauthentifizierung über JAAS erfolgt.

**Konnektivität**

Wenn die mobile Anwendung von einer drahtlosen Versorgung abhängt, muss sie in der Lage sein, offline zu agieren und auf effiziente Weise mit einer unterbrochenen Verbindung umgehen zu können. In dieser Umgebung besteht das Ziel darin, verbunden zu bleiben, die Anwendung muss jedoch Nachrichten im Store-and-forward-Verfahren verarbeiten können. Bei den Nachrichten handelt es sich häufig um Bestellungen oder Zustellnachweise, die von wichtigem geschäftlichen Nutzen sind. Sie müssen zuverlässig im Store-and-forward-Verfahren gespeichert und weitergeleitet werden.

**Skalierbarkeit**

Die Skalierbarkeit spielt in diesem Fall eine untergeordnete Rolle. In angepassten Anwendungsfällen mit mobilen Anwendungen ist es unwahrscheinlich, dass die Anzahl der Anwendungsclients Tausende oder Zehntausende von Clients überschreitet.

## Windows Linux AIX Telemetriegeräte mit einem Warteschlangenmanager verbinden

Telemetriegeräte verbinden sich unter Verwendung eines MQTT v3-Clients mit einem Warteschlangenmanager. Der MQTT v3-Client nutzt TCP/IP, um eine Verbindung zu einem TCP/IP-Empfangsprogramm, dem sogenannten Telemetrieservice (MQXR), herzustellen.

Wenn Sie ein Telemetriegerät mit einem Warteschlangenmanager verbinden, leitet der MQTT-Client mit der Methode `MqttClient.connect` eine TCP/IP-Verbindung ein. Wie IBM MQ-Clients muss auch ein MQTT-Client mit dem Warteschlangenmanager verbunden sein, damit Nachrichten gesendet und empfangen werden können. Die Verbindung wird auf dem Server unter Verwendung eines TCP/IP-Empfangsprogramms hergestellt, das im Rahmen von MQ Telemetry installiert und als Telemetrieservice (MQXR)

bezeichnet wird. Auf jedem Warteschlangenmanager wird höchstens ein Telemetrieservice (MQXR) ausgeführt.

Der Telemetrieservice (MQXR) nutzt zur Zuordnung der Verbindung zu einem Telemetriekanal die Adresse des fernen Sockets, die durch jeden Client in der Methode `MqttClient.connect` festgelegt wird. Eine Socketadresse ist eine Kombination aus TCP/IP-Hostnamen und Portnummer. Mehrere Clients, die dieselbe Adresse eines fernen Sockets verwenden, werden vom Telemetrieservice (MQXR) mit demselben Telemetriekanal verbunden.

Falls sich auf einem Server mehrere Warteschlangenmanager befinden, teilen Sie die Telemetriekanäle zwischen den Warteschlangenmanagern auf. Ordnen Sie die Adressen ferner Sockets den verschiedenen Warteschlangenmanagern zu. Definieren Sie jeden Telemetriekanal mit einer eindeutigen Adresse eines fernen Sockets. Eine Socketadresse kann nicht von zwei Telemetriekanälen verwendet werden.

Wenn dieselbe Adresse eines fernen Sockets für Telemetriekanäle auf mehreren Warteschlangenmanagern konfiguriert ist, wird der erste Telemetriekanal für die Verbindung genutzt. Nachfolgende Kanäle, die sich unter derselben Adresse verbinden, schlagen fehl.

Falls sich auf dem Server mehrere Netzadapter befinden, teilen Sie die Adressen ferner Sockets zwischen den Telemetriekanälen auf. Die Zuordnung der Socketadressen erfolgt völlig beliebig, sofern eine bestimmte Socketadresse in nur einem Telemetriekanal konfiguriert ist.

Konfigurieren Sie IBM MQ für die Verbindung von MQTT-Clients unter Verwendung der Assistenten, die in der MQ Telemetry-Ergänzung für IBM MQ Explorer bereitgestellt werden. Alternativ können Sie die Telemetrie auch manuell konfigurieren. Folgen Sie dazu den Anweisungen in den Abschnitten Warteschlangenmanager für Telemetrie unter Linux und AIX konfigurieren und Warteschlangenmanager für Telemetrie unter Windows konfigurieren.

**Zugehörige Verweise**

MQXR-Eigenschaften

## Windows Linux AIX Telemetry-Verbindungsprotokolle

MQ Telemetry unterstützt TCP/IP IPv4 und IPv6 sowie TLS.

## Windows Linux AIX Telemetrieservice (MQXR)

Beim Telemetrieservice (MQXR) handelt es sich um ein TCP/IP-Empfangsprogramm, das als IBM MQ-Service verwaltet wird. Erstellen Sie den Service mithilfe eines IBM MQ Explorer-Assistenten oder mit einem **runmqsc**-Befehl.

Der MQ Telemetry -Service (MQXR) heißt SYSTEM.MQXR.SERVICE

Der Assistent für die Telemetry-Beispielkonfiguration, der mit der MQ Telemetry-Funktion für IBM MQ Explorer bereitgestellt wird, erstellt den Telemetrieservice und einen Beispieltelemetriekanal; siehe Installation von MQ Telemetry mithilfe von IBM MQ Explorerüberprüfen.

Erstellen Sie die Beispielkonfiguration über die Befehlszeile; siehe Installation von MQ Telemetry über die Befehlszeile überprüfen.

Der Telemetrieservice (MQXR) wird gemeinsam mit dem Warteschlangenmanager automatisch gestartet und gestoppt. Steuern Sie den Service über den Ordner 'Services' in IBM MQ Explorer. Zum Anzeigen des Service müssen Sie auf das Symbol klicken, um das IBM MQ Explorer Herausfiltern von SYSTEM-Objekten aus der Anzeige zu stoppen.

Ein Beispiel für die manuelle Erstellung des Service finden Sie unter

- Linux AIX SYSTEM.MQXR.SERVICE unter Linuxerstellen.
- Windows SYSTEM.MQXR.SERVICE unter Windowserstellen.

V 9.3.0 Ab IBM MQ 9.3.0 werden die Optionen SYSTEM.MQXR.SERVICE unter Linuxerstellenund SYSTEM.MQXR.SERVICE unter Windowserstellen aktualisiert, um den Standardschlüssel anzugeben, da-

mit Kennphrasen für MQTT-TLS-Kanäle verschlüsselt werden müssen. Weitere Informationen finden Sie unter Verschlüsselung von Passphrasen für MQTT TLS-Kanäle.

## Windows Linux AIX Telemetriekanäle

Erstellen Sie Telemetriekanäle, um Verbindungen mit unterschiedlichen Eigenschaften wie einer JAAS- oder TLS-Authentifizierung (JAAS = Java Authentication and Authorization Service) zu erstellen oder Clientgruppen zu verwalten.

Erstellen Sie Telemetriekanäle mit dem Assistenten **New Telemetry Channel** , der in der Funktion MQ Telemetry für IBM MQ Explorerbereitgestellt wird. Konfigurieren Sie einen Kanal unter Verwendung des Assistenten für die Annahme von Verbindungen aus MQTT-Clients an einem bestimmten TCP/IP-Port. Seit IBM WebSphere MQ 7.1 kann MQ Telemetry mit dem Befehlszeilenprogramm **runmqsc** konfiguriert werden.

Erstellen Sie mehrere Telemetriekanäle an verschiedenen Ports, um die Verwaltung sehr vieler Clienttransaktionen durch die Aufteilung von Clients in Gruppen zu vereinfachen. Jeder Telemetriekanal hat einen anderen Namen.

Sie können Telemetriekanäle mit unterschiedlichen Sicherheitsattributen konfigurieren und so verschiedene Verbindungsarten erstellen. Erstellen Sie mehrere Kanäle, damit Clientverbindungen an verschiedenen TCP/IP-Adressen akzeptiert werden können. Verschlüsseln Sie Nachrichten mithilfe von TLS und authentifizieren Sie den Telemetriekanal und Client; siehe TLS-Konfiguration von MQTT-Clients und Telemetriekanälen. Geben Sie die Benutzer-ID zur einfacheren Berechtigung des Zugriffs auf IBM MQ-Objekte. Geben Sie eine JAAS-Konfiguration an, um den MQTT-Benutzer mit JAAS zu authentifizieren; siehe MQTT-Clientidentifikation, Autorisierung und Authentifizierung.

## Windows Linux AIX IBM MQ Telemetry Transport-Protokoll

Das IBM MQ Telemetry Transport (MQTT) v3 wurde für den Nachrichtenaustausch zwischen kompakten Endgeräten mit geringer Bandbreite oder kostenintensiven Verbindungen und für eine zuverlässige Nachrichtenübertragung entwickelt. Es verwendet TCP/IP.

Das MQTT protocol ist veröffentlicht (siehe IBM MQ Telemetry Transport-Format und -Protokoll). Version 3 des Protokolls nutzt Publish/Subscribe und unterstützt drei Servicequalitäten: *Fire and Forget*, *mindestens einmal* und *genau einmal*.

Die geringe Größe der Protokollheader und der Bytefeldgruppe der Nachrichtennutzdaten sorgt für Nachrichten mit geringem Umfang. Die Header umfassen einen festgelegten Header mit 2 Bytes und bis zu 12 Bytes zusätzliche variable Header. Das Protokoll verwendet variable Header mit 12 Bytes für Subskriptionen und Verbindungen und variable Header mit nur 2 Bytes für die meisten Veröffentlichungen.

Mit drei Servicequalitäten haben Sie Spielraum für einen guten Ausgleich zwischen niedrigen Latenzzeiten und Zuverlässigkeit; siehe Von einem MQTT-Client bereitgestellte Servicequalitäten. *Fire and Forget* verwendet keinen permanenten Gerätespeicher und nur eine Übertragung für das Senden oder Empfangen einer Veröffentlichung. *Mindestens einmal* und *genau einmal* erfordern einen permanenten Speicher im Gerät, um den Protokollstatus aufrechtzuerhalten und eine Nachricht bis zu ihrer Bestätigung zu speichern.

## Windows Linux AIX MQTT-Clients

Eine MQTT-Client-App ist für die Erfassung von Daten aus dem Telemetriegerät, die Verbindung zum Server und die Veröffentlichung der Informationen auf dem Server verantwortlich. Sie kann außerdem Themen abonnieren, Veröffentlichungen empfangen und das Telemetriegerät steuern.

Im Gegensatz zu IBM MQ-Clientanwendungen handelt es sich bei MQTT-Client-Apps um keine IBM MQ-Anwendungen. Sie geben keinen Warteschlangenmanager für die Verbindungsherstellung an. Sie sind nicht auf die Verwendung bestimmter IBM MQ-Programmierschnittstellen begrenzt. Stattdessen implementieren MQTT-Clients das MQTT 3-Protokoll. Sie können Ihre eigene Clientbibliothek als Schnittstelle zum MQTT protocol in der Programmiersprache und auf der Plattform Ihrer Wahl schreiben. Weitere Informationen finden Sie im Abschnitt IBM MQ Telemetry Transport-Format und -Protokoll.

Die Entwicklung von MQTT-Client-Apps erleichtern Sie sich durch die C-, Java- und JavaScript-Clientbibliotheken, in denen das MQTT protocol für eine Reihe von Plattformen enthalten ist. Wenn Sie diese Bibliotheken in Ihre MQTT-Apps einbinden, können Sie auch einen MQTT-Client mit vollem Funktionsumfang schreiben, der nicht länger als 15 Codezeilen ist. MQTT-Clientbibliotheken sind bei Eclipse Paho und MQTT.org kostenlos erhältlich. Weitere Informationen finden Sie im Abschnitt IBM MQ Telemetry Transport-Beispielprogramme.

Die MQTT-Client-App ist immer für den Aufbau einer Verbindung mit einem Telemetriekanal verantwortlich. Sobald die Verbindung hergestellt ist, kann entweder die MQTT-Client-App oder eine IBM MQ-Anwendung den Nachrichtenaustausch starten.

MQTT-Client-Apps und IBM MQ-Anwendungen veröffentlichen und abonnieren dieselbe Themengruppe. Eine IBM MQ-Anwendung kann auch eine Nachricht direkt an eine MQTT-Client-App senden, ohne dass die Client-App eine Subskription erstellen muss. Weitere Informationen finden Sie im Abschnitt Verteilte Steuerung von Warteschlangen zum Senden von Nachrichten an MQTT-Clients verwenden.

MQTT-Client-Apps werden unter Verwendung eines Telemetriekanals mit IBM MQ verbunden. Der Telemetriekanal agiert als Brücke zwischen den unterschiedlichen Nachrichtentypen, die von MQTT und IBM MQ verwendet werden. Er erstellt für die MQTT-Client-App Veröffentlichungen und Subskriptionen im Warteschlangenmanager. Der Telemetriekanal sendet Veröffentlichungen, die den Subskriptionen einer MQTT-Client-App entsprechen, vom Warteschlangenmanager an die MQTT-Client-App.

## Windows Linux AIX Nachricht an einen MQTT-Client senden

IBM MQ-Anwendungen können Nachrichten an MQTT v3-Clients senden, indem Sie Informationen in Subskriptionen veröffentlichen, die von Clients erstellt wurden, oder indem Sie die Nachrichten direkt senden. MQTT-Clients können sich gegenseitig Nachrichten senden, indem sie Informationen in Themen veröffentlichen, die von anderen Clients abonniert wurden.

### Ein MQTT-Client abonniert eine Veröffentlichung, die er von IBM MQ erhält

Führen Sie die Task „Nachricht über IBM MQ Explorer im MQTT -Clientdienstprogramm veröffentlichen” auf Seite 131 aus, um eine Veröffentlichung aus IBM MQ an einen MQTT-Client zu senden.

Üblicherweise erhält ein MQTT v3-Client Nachrichten, indem er eine Subskription für ein Thema oder eine Themengruppe erstellt. Im Beispielcode-Snippet Abbildung 44 auf Seite 130subskribiert der MQTT -Client die Topiczeichenfolge `"MQTT Examples"`. Eine IBM MQ C-Anwendung, Abbildung 45 auf Seite 130, veröffentlicht unter Verwendung der Themenzeichenfolge `"MQTT Examples"`im Thema. Im Codeausschnitt Abbildung 46 auf Seite 131 empfängt der MQTT-Client die Veröffentlichung in der Callback-Methode `messageArrived`.

Weitere Informationen zum Konfigurieren von IBM MQ für das Senden von Veröffentlichungen als Antwort auf Subskriptionen von MQTT -Clients finden Sie unter Nachricht als Antwort auf eine MQTT -Clientsubskription veröffentlichen.

### Eine IBM MQ-Anwendung sendet eine Nachricht direkt an einen MQTT-Client

Führen Sie die Task „Nachricht mithilfe von IBM MQ Explorer an einen MQTT -Client senden” auf Seite 136 aus, um eine Nachricht aus IBM MQ direkt an einen MQTT-Client zu senden.

Eine Nachricht, die auf diese Weise an einen MQTT-Client gesendet wird, wird als "nicht erwartete Nachricht" bezeichnet. MQTT v3-Clients empfangen nicht erwartete Nachrichten als Veröffentlichungen mit einem festgelegten Themennamen. Der Telemetrieservice (MQXR) setzt den Themennamen auf den Namen der fernen Warteschlange.

Weitere Informationen zur Konfiguration von IBM MQ für das direkte Senden von Nachrichten an MQTT -Clients finden Sie unter Nachricht direkt an einen Client senden.

## Ein MQTT-Client veröffentlicht eine Nachricht

Ein MQTT v3-Client kann eine Nachricht veröffentlichen, die von einem anderen MQTT v3-Client empfangen wird, er kann jedoch keine nicht erwartete Nachricht senden. Der Codeausschnitt Abbildung 47 auf Seite 131 zeigt, wie ein MQTT v3-Client, der in der Programmiersprache Java geschrieben wurde, eine Nachricht veröffentlicht.

Das typische Beispiel für das Senden einer Nachricht an einen bestimmten MQTT v3-Client sieht so aus, dass jeder Client eine Subskription für seinen eigenen `ClientIdentifier`-Eintrag erstellt. Führen Sie die Task „Nachricht an einen bestimmten MQTT v3-Client veröffentlichen" auf Seite 137 aus, um eine Nachricht von einem MQTT-Client für einen anderen MQTT-Client unter Verwendung von `ClientIdentifier` als Topic-Zeichenfolge zu veröffentlichen.

### Beispielcodeausschnitte

Das Code-Snippet in Abbildung 44 auf Seite 130 zeigt, wie ein in Java geschriebener MQTT -Client eine Subskription erstellt. Er benötigt außerdem eine Callback-Methode (`messageArrived`), um Veröffentlichungen für die Subskription zu empfangen.

```
String    clientId = String.format("%-23.23s",
                     System.getProperty("user.name") + "_" +
                     (UUID.randomUUID().toString())).trim()).replace('-', '_');
MqttClient  client = new MqttClient("localhost", clientId);
String topicString = "MQTT Examples";
int            QoS = 1;
client.subscribe(topicString, QoS);
```

*Abbildung 44. MQTT v3-Client-Subskribent*

Der Codeausschnitt in Abbildung 45 auf Seite 130 zeigt, wie eine IBM MQ-Anwendung, die in der Programmiersprache C geschrieben wurde, eine Veröffentlichung sendet. Der Codeausschnitt wird aus der Task Veröffentlichungskomponente für ein variables Thema erstellen extrahiert.

```
/* Define and set variables to.defaults */
/* Omitted lines declaring variables    */
char * topicName   = ""
char * topicString = "MQTT Examples"
char * publication = "Hello world!";
do {
  MQCONN(qMgrName, &Hconn, &CompCode, &Reason);
  if (CompCode != MQCC_OK) break;
  td.ObjectType = MQOT_TOPIC;    /* Object is a topic             */
  td.Version = MQOD_VERSION_4;   /* Descriptor needs to be V4     */
  strncpy(td.ObjectName, topicName,  MQ_TOPIC_NAME_LENGTH);
  td.ObjectString.VSPtr = topicString;
  td.ObjectString.VSLength = (MQLONG)strlen(topicString);
  MQOPEN(Hconn, &td, MQOO_OUTPUT | MQOO_FAIL_IF_QUIESCING, &Hobj, &CompCode, &Reason);
  if (CompCode != MQCC_OK) break;
  pmo.Options = MQPMO_FAIL_IF_QUIESCING | MQPMO_RETAIN;
  MQPUT(Hconn, Hobj, &md, &pmo, (MQLONG)strlen(publication)+1, publication, &CompCode, &Reason);
  if (CompCode != MQCC_OK) break;
  MQCLOSE(Hconn, &Hobj, MQCO_NONE, &CompCode, &Reason);
  if (CompCode != MQCC_OK) break;
  MQDISC(&Hconn, &CompCode, &Reason);
} while (0);
```

*Abbildung 45. IBM MQ-Publisher*

Wenn die Veröffentlichung ankommt, ruft der MQTT -Client die Methode `messageArrived` der Klasse `MqttCallback` des MQTT Anwendungsclients auf.

```
public class CallBack implements MqttCallback {
  public void messageArrived(MqttTopic topic, MqttMessage message) {
    try {
      System.out.println("Message arrived: \"" + message.toString()
          + "\" on topic \"" + topic.toString() + "\"");
    } catch (Exception e) {
      e.printStackTrace();
    }
  }
// ... Other callback methods
}
```

*Abbildung 46. `messageArrived`-Methode*

Abbildung 47 auf Seite 131 zeigt einen MQTT v3-Client, der eine Nachricht in der Subskription veröffentlicht, die in Abbildung 44 auf Seite 130 erstellt wurde.

---

```
      String          address = "localhost";
      String         clientId = String.format("%-23.23s",
                                System.getProperty("user.name") + "_" +
                                (UUID.randomUUID().toString())).trim()).replace('-', '_');
      MqttClient       client = new MqttClient(address, clientId);
      String      topicString = "MQTT Examples";
      MqttTopic         topic = client.getTopic(Example.topicString);
      String     publication  = "Hello world";
      MqttMessage     message = new MqttMessage(publication.getBytes());
      MqttDeliveryToken token = topic.publish(message);
```

*Abbildung 47. MQTT v3-Client-Publisher*

---

## Windows Linux AIX Nachricht über IBM MQ Explorer im MQTT -Clientdienstprogramm veröffentlichen

Führen Sie die Schritte in dieser Task aus, um eine Nachricht unter Verwendung von IBM MQ Explorer zu veröffentlichen und diese mit dem MQTT-Clientdienstprogramm zu abonnieren. In einer weiteren Task wird veranschaulicht, wie die Standardübertragungswarteschlange nicht auf `SYSTEM.MQTT.TRANSMIT.QUEUE` gesetzt, sondern ein Warteschlangenmanager-Aliasname konfiguriert wird.

### Vorbereitende Schritte

Bei der Task wird vorausgesetzt, dass Sie mit IBM MQ und dem IBM MQ Explorer vertraut sind, und dass IBM MQ und die Komponente MQ Telemetry installiert sind.

Der Benutzer, der die Warteschlangenmanagerressourcen für diese Task erstellt, muss über ausreichende Berechtigungen für diesen Vorgang verfügen. Zu Demonstrationszwecken wird davon ausgegangen, dass die IBM MQ Explorer-Benutzer-ID der Gruppe `mqm` angehört.

### Informationen zu diesem Vorgang

In dieser Task erstellen Sie ein Thema in IBM MQ und abonnieren das Thema unter Verwendung des MQTT-Clientdienstprogramms. Wenn Sie unter Verwendung von IBM MQ Explorer Informationen in dem Thema veröffentlichen, empfängt der MQTT-Client die Veröffentlichung.

### Vorgehensweise

Führen Sie eine der folgenden Tasks aus:

- Sie haben MQ Telemetry zwar installiert, aber noch nicht gestartet. Führen Sie diese Task aus: „Task ohne definierten Telemetrieservice (MQXR) starten” auf Seite 132.

- Sie haben IBM MQ Telemetry zuvor bereits ausgeführt, möchten für die Demonstration aber einen neuen Warteschlangenmanager verwenden. Führen Sie diese Task aus: „Task ohne definierten Telemetrieservice (MQXR) starten" auf Seite 132.
- Sie möchten die Task mit einem vorhandenen Warteschlangenmanager ausführen, auf dem keine Telemetrieressourcen definiert sind. Sie möchten den Assistenten **Beispielkonfiguration definieren** nicht ausführen.

  a. Führen Sie eine der folgenden Tasks aus, um die Telemetrie einzurichten:

     – Warteschlangenmanager für Telemetrie unter Linux und AIX konfigurieren
     – Warteschlangenmanager für Telemetrie unter Windows konfigurieren

  b. Führen Sie die folgende Task aus: „Task mit einem aktiven Telemetrieservice (MQXR) starten" auf Seite 133
- Wenn Sie die Task mit einem vorhandenen Warteschlangenmanager ausführen möchten, auf dem bereits Telemetrieressourcen definiert sind, führen Sie diese Task aus: „Task mit einem aktiven Telemetrieservice (MQXR) starten" auf Seite 133.

## Nächste Schritte

Führen Sie die Task „Nachricht mithilfe von IBM MQ Explorer an einen MQTT -Client senden" auf Seite 136 aus, um eine Nachricht direkt an das Clientdienstprogramm zu senden.

## *Task ohne definierten Telemetrieservice (MQXR) starten*

Erstellen Sie einen Warteschlangenmanager und führen Sie den Assistenten **Beispielkonfiguration definieren** aus, um Beispieltelemetrieressourcen für den Warteschlangenmanager zu definieren. Veröffentlichen Sie eine Nachricht unter Verwendung von IBM MQ Explorer und abonnieren Sie diese mit dem MQTT-Clientdienstprogramm.

## Informationen zu diesem Vorgang

Wenn Sie Beispieltelemetrieressourcen mithilfe des Assistenten **Beispielkonfiguration definieren** einrichten, legt der Assistent die Berechtigungen für die Gastbenutzer-ID fest. Überlegen Sie genau, ob Sie eine derartige Berechtigung der Gastbenutzer-ID wünschen. `guest` unter Windowsund `nobody` unter Linuxerhalten die Berechtigung zum Veröffentlichen und Subskribieren im Stammverzeichnis der Themenstruktur und zum Einreihen von Nachrichten in `SYSTEM.MQTT.TRANSMIT.QUEUE`.

Der Assistent setzt außerdem die Standardübertragungswarteschlange auf `SYSTEM.MQTT.TRANSMIT.QUEUE`, was zu Konflikten bei Anwendungen führen kann, die auf einem bestehenden Warteschlangenmanager ausgeführt werden. Es ist zwar aufwändig, jedoch möglich, die Telemetrie zu konfigurieren und nicht die Standardübertragungswarteschlange zu verwenden; führen Sie diese Folgetask aus: „Warteschlangenmanager-Aliasnamen verwenden" auf Seite 134. In dieser Task erstellen Sie einen Warteschlangenmanager, um einen möglichen Konflikt mit vorhandenen Standardübertragungswarteschlangen zu vermeiden.

## Vorgehensweise

1. Erstellen Sie unter Verwendung von IBM MQ Explorer einen neuen Warteschlangenmanager und starten Sie diesen.

   a) Klicken Sie mit der rechten Maustaste auf den Ordner `Queue Managers` und klicken Sie auf **Neu** > **Warteschlangenmanager ...**. Geben Sie einen Warteschlangenmanagername ein und klicken Sie auf **Fertigstellen**.

   Denken Sie sich einen Warteschlangenmanagernamen aus, z. B. `MQTTQMGR`.
2. Erstellen und starten Sie den Telemetrieservice (MQXR) und erstellen Sie einen Beispieltelemetriekanal.

   a) Öffnen Sie den Ordner `Queue Managers\`*`QmgrName`*`\Telemetry`.

   b) Klicken Sie auf **Beispielkonfiguration definieren ...** > **Fertigstellen**.

Lassen Sie das Kontrollkästchen **MQTT-Clientdienstprogramm starten** aktiviert.

3. Erstellen Sie mit dem MQTT -Clientdienstprogramm eine Subskription für `MQTT Example` .
   a) Klicken Sie auf **Verbinden**.

   Das **Clientprotokoll** enthält jetzt das Ereignis `Connected` (Verbunden).

   b) Geben Sie `MQTT Example` in das Feld **Subscription \ Topic** ein > **Subscribe**.

   Das **Clientprotokoll** enthält jetzt das Ereignis `Subscribed` (Verbunden).
4. Erstellen Sie `MQTTExampleTopic` in IBM MQ.
   a) Klicken Sie mit der rechten Maustaste auf den Ordner `Queue Managers\`*`QmgrName`*`\Topics` in **MQ Explorer** und klicken Sie auf **Neu** > **Thema**.
   b) Geben Sie `MQTTExampleTopic` als **Name** ein und klicken Sie auf **Weiter**.
   c) Geben Sie `MQTT Example` als **Themenzeichenfolge** > **Fertigstellen**ein.
   d) Klicken Sie auf **OK**, um das Bestätigungsfenster zu schließen.
5. Veröffentlichen Sie `Hello World!` mithilfe von IBM MQ Explorerim Thema `MQTT Example` .
   a) Klicken Sie im IBM MQ Explorer auf den Ordner `Queue Managers\`*`QmgrName`*`\Topics`.
   b) Klicken Sie mit der rechten Maustaste auf `MQTTExampleTopic` > **Testveröffentlichung...**
   c) Geben Sie `Hello World!` in das Feld **Nachrichtendaten** ein und klicken Sie auf **Nachricht veröffentlichen**. Wechseln zum Fenster des MQTT-Clientdienstprogramms.

   Das **Clientprotokoll** enthält jetzt das Ereignis `Received` (Verbunden).

### *Task mit einem aktiven Telemetrieservice (MQXR) starten*

Erstellen Sie einen Telemetriekanal und ein Thema. Berechtigen Sie den Benutzer für die Verwendung des Themas und der Übertragungswarteschlange der Telemetrie. Veröffentlichen Sie eine Nachricht unter Verwendung von IBM MQ Explorer und abonnieren Sie diese mit dem MQTT-Clientdienstprogramm.

## Vorbereitende Schritte

In dieser Version der Task ist der Warteschlangenmanager *QmgrName*definiert und aktiv. Ein Telemetrieservice (MQXR) ist definiert und aktiv. Der Telemetrieservice (MQXR) kann manuell oder durch Ausführung des Assistenten **Beispielkonfiguration definieren** erstellt worden sein.

## Informationen zu diesem Vorgang

In dieser Task konfigurieren Sie einen vorhandenen Warteschlangenmanager für das Senden einer Veröffentlichung an das MQTT-Clientdienstprogramm.

In Schritt „1” auf Seite 133 der Task wird außerdem die Standardübertragungswarteschlange auf `SYSTEM.MQTT.TRANSMIT.QUEUE` gesetzt, was zu Konflikten bei Anwendungen führen kann, die auf einem bestehenden Warteschlangenmanager ausgeführt werden. Es ist zwar aufwändig, jedoch möglich, die Telemetrie zu konfigurieren und nicht die Standardübertragungswarteschlange zu verwenden; führen Sie diese Folgetask aus: „Warteschlangenmanager-Aliasnamen verwenden” auf Seite 134.

## Vorgehensweise

1. Legen Sie `SYSTEM.MQTT.TRANSMIT.QUEUE` als Standardübertragungswarteschlange fest.
   a) Klicken Sie mit der rechten Maustaste auf `Queue Managers\`*`QmgrName`* `folder` und klicken Sie auf **Eigenschaften ...**
   b) Klicken Sie im Navigator auf **Kommunikation**.
   c) Klicken Sie auf **Auswählen ...** > Auswählen `SYSTEM.MQTT.TRANSMIT.QUEUE` > **OK** > **OK**.
2. Erstellen Sie den Telemetriekanal `MQTTExampleChannel`, um das MQTT-Clientdienstprogramm mit IBM MQ zu verbinden, und starten Sie das MQTT-Clientdienstprogramm.

a) Klicken Sie mit der rechten Maustaste auf den Ordner `Queue Managers\`*`QmgrName`*` \Telemetry\Channels` im **MQ Explorer** und anschließend auf **Neu** > **Telemetriekanal ...**.

b) Geben Sie `MQTTExampleChannel` in das Feld **Kanalname** ein und klicken Sie auf **Weiter** > **Weiter**.

c) Ändern Sie den Eintrag unter **Festgelegte Benutzer-ID** im Fenster für die Clientberechtigung in die Benutzer-ID, die für `MQTTExample` Veröffentlichungen und Subskriptionen vornehmen wird, und klicken Sie auf **Weiter**.

d) Lassen Sie die Option **Clientdienstprogramm starten** aktiviert und klicken Sie auf **Fertigstellen**.

3. Erstellen Sie mit dem MQTT -Clientdienstprogramm eine Subskription für `MQTT Example` .

a) Klicken Sie auf **Verbinden**.

Das **Clientprotokoll** enthält jetzt das Ereignis `Connected` (Verbunden).

b) Geben Sie `MQTT Example` in das Feld **Subscription \ Topic** ein > **Subscribe**.

Das **Clientprotokoll** enthält jetzt das Ereignis `Subscribed` (Verbunden).

4. Erstellen Sie `MQTTExampleTopic` in IBM MQ.

a) Klicken Sie mit der rechten Maustaste auf den Ordner `Queue Managers\`*`QmgrName`*`\Topics` in **MQ Explorer** und klicken Sie auf **Neu** > **Thema**.

b) Geben Sie `MQTTExampleTopic` als **Name** ein und klicken Sie auf **Weiter**.

c) Geben Sie `MQTT Example` als **Themenzeichenfolge** > **Fertigstellen**ein.

d) Klicken Sie auf **OK**, um das Bestätigungsfenster zu schließen.

5. Wenn Sie möchten, dass ein Benutzer, der nicht zur Gruppe `mqm` gehört, Veröffentlichungen und Subskriptionen für das Thema `MQTTExample` vornimmt, gehen Sie wie folgt vor:

a) Berechtigen Sie den Benutzer für die Veröffentlichung und Subskription im Zusammenhang mit dem Thema `MQTTExampleTopic`:

```
setmqaut -m qMgrName -t topic -n MQTTExampleTopic -p User ID -all +pub +sub
```

b) Berechtigen Sie den Benutzer zur Einreihung einer Nachricht in die Warteschlange `SYSTEM.MQTT.TRANSMIT.QUEUE`:

```
setmqaut -m qMgrName -t q -n SYSTEM.MQTT.TRANSMIT.QUEUE -p User ID -all +put
```

6. Veröffentlichen Sie `Hello World!` mithilfe von IBM MQ Explorerim Thema `MQTT Example` .

a) Klicken Sie im IBM MQ Explorer auf den Ordner `Queue Managers\`*`QmgrName`*`\Topics`.

b) Klicken Sie mit der rechten Maustaste auf `MQTTExampleTopic` > **Testveröffentlichung...**

c) Geben Sie `Hello World!` in das Feld **Nachrichtendaten** ein und klicken Sie auf **Nachricht veröffentlichen**. Wechseln zum Fenster des MQTT-Clientdienstprogramms.

Das **Clientprotokoll** enthält jetzt das Ereignis `Received` (Verbunden).

### *Warteschlangenmanager-Aliasnamen verwenden*

Veröffentlichen Sie eine Nachricht im MQTT -Clientdienstprogramm unter Verwendung von IBM MQ Explorer , ohne die Standardübertragungswarteschlange auf `SYSTEM.MQTT.TRANSMIT.QUEUE`zu setzen.

Diese Task ist eine Fortsetzung der vorherigen Task; dabei wird ein Warteschlangenmanager-Aliasname verwendet, damit die Standardübertragungswarteschlange nicht auf `SYSTEM.MQTT.TRANSMIT.QUEUE` gesetzt werden muss.

## Vorbereitende Schritte

Die Task „Task ohne definierten Telemetrieservice (MQXR) starten” auf Seite 132 oder „Task mit einem aktiven Telemetrieservice (MQXR) starten” auf Seite 133 muss vollständig ausgeführt worden sein.

## Informationen zu diesem Vorgang

Wenn ein MQTT-Client eine Subskription erstellt, sendet IBM MQ seine Antwort unter Verwendung von `ClientIdentifier` für den Namen des fernen Warteschlangenmanagers. In dieser Task wird `MyClient` als Client-ID (`ClientIdentifier`) verwendet.

Falls keine Übertragungswarteschlange oder kein Warteschlangenmanager-Aliasname namens `MyClient` vorhanden ist, wird die Antwort in die Standardübertragungswarteschlange gestellt. Wenn Sie die Standardübertragungswarteschlange auf `SYSTEM.MQTT.TRANSMIT.QUEUE`setzen, erhält der MQTT -Client die Antwort.

Wenn Sie Warteschlangenmanager-Aliasnamen verwenden, müssen Sie die Standardübertragungswarteschlange nicht auf `SYSTEM.MQTT.TRANSMIT.QUEUE` setzen. Sie müssen für jede Instanz von `ClientIdentifier` einen Warteschlangenmanager-Aliasnamen definieren. Für gewöhnlich können Warteschlangenmanager-Aliasnamen aufgrund der Vielzahl an Clients nicht verwendet werden. Häufig ist der `ClientIdentifier`-Eintrag unvorhersehbar, was eine derartige Konfiguration der Telemetrie unmöglich macht.

In manchen Situationen kann es jedoch vorkommen, dass Sie die Standardübertragungswarteschlange mit einem anderen Wert als `SYSTEM.MQTT.TRANSMIT.QUEUE` konfigurieren müssen. Mit den Schritten unter Vorgehensweise wird ein Warteschlangenmanager-Alias konfiguriert, anstatt die Standardübertragungswarteschlange auf `SYSTEM.MQTT.TRANSMIT.QUEUE`zu setzen.

## Vorgehensweise

1. Entfernen Sie `SYSTEM.MQTT.TRANSMIT.QUEUE` als Standardübertragungswarteschlange.
   a) Klicken Sie mit der rechten Maustaste auf `Queue Managers\`*`QmgrName`* `folder` und klicken Sie auf **Eigenschaften ...**
   b) Klicken Sie im Navigator auf **Kommunikation**.
   c) Entfernen Sie `SYSTEM.MQTT.TRANSMIT.QUEUE` aus dem Feld **Standardübertragungswarteschlange** und klicken Sie auf **OK**.
2. Vergewissern Sie sich, dass mit dem MQTT-Clientdienstprogramm keine Erstellung einer Subskription mehr möglich ist:
   a) Klicken Sie auf **Verbinden**.

      Das **Clientprotokoll** enthält jetzt das Ereignis `Connected` (Verbunden).
   b) Geben Sie `MQTT Example` in das Feld **Subscription \ Topic** ein > **Subscribe**.

   Im **Clientprotokoll** werden die Ereignisse `Subscribe failed` (Subskription fehlgeschlagen) und `Connection lost` (Verbindung nicht mehr vorhanden) aufgezeichnet.
3. Erstellen Sie einen Warteschlangenmanager-Aliasnamen für den `ClientIdentifier`-Eintrag `MyClient`.
   a) Klicken Sie mit der rechten Maustaste auf den Ordner `Queue Managers\`*`QmgrName`*`\Queues` und klicken Sie anschießend auf **Neu** > **Definition der fernen Warteschlange**.
   b) Geben Sie der Definition den Namen `MyClient` > und klicken Sie auf **Weiter**.
   c) Geben Sie `MyClient` in das Feld **Ferner Warteschlangenmanager** ein.
   d) Geben Sie `SYSTEM.MQTT.TRANSMIT.QUEUE` im Feld **Übertragungswarteschlange** und klicken Sie auf **Fertigstellen**.
4. Stellen Sie wieder eine Verbindung zum MQTT-Clientdienstprogramm her.
   a) Vergewissern Sie sich, dass die **Client-ID** auf `MyClient` gesetzt ist.
   b) **Verbindung herstellen**

   Das **Clientprotokoll** enthält jetzt das Ereignis `Connected` (Verbunden).
5. Erstellen Sie mit dem MQTT -Clientdienstprogramm eine Subskription für `MQTT Example` .
   a) Klicken Sie auf **Verbinden**.

Das **Clientprotokoll** enthält jetzt das Ereignis `Connected` (Verbunden).

b) Geben Sie `MQTT Example` in das Feld **Subscription \ Topic** ein > **Subscribe**.

Das **Clientprotokoll** enthält jetzt das Ereignis `Subscribed` (Verbunden).

6. Veröffentlichen Sie `Hello World!` mithilfe von IBM MQ Explorerim Thema `MQTT Example` .
   a) Klicken Sie im IBM MQ Explorer auf den Ordner `Queue Managers\`*`QmgrName`*`\Topics`.
   b) Klicken Sie mit der rechten Maustaste auf `MQTTExampleTopic` > **Testveröffentlichung...**
   c) Geben Sie `Hello World!` in das Feld **Nachrichtendaten** ein und klicken Sie auf **Nachricht veröffentlichen**. Wechseln zum Fenster des MQTT-Clientdienstprogramms.

   Das **Clientprotokoll** enthält jetzt das Ereignis `Received` (Verbunden).

## Windows Linux AIX Nachricht mithilfe von IBM MQ Explorer an einen MQTT -Client senden

Senden Sie eine Nachricht an das MQTT -Clientdienstprogramm, indem Sie eine Nachricht mithilfe von IBM MQ Explorerin eine IBM MQ -Warteschlange stellen. Diese Task veranschaulicht die Konfiguration einer Definition einer fernen Warteschlange zum direkten Senden einer Nachricht an einen MQTT-Client.

### Vorbereitende Schritte

Führen Sie den im Abschnitt „Nachricht über IBM MQ Explorer im MQTT -Clientdienstprogramm veröffentlichen” auf Seite 131 beschriebenen Schritt aus. Behalten Sie die Verbindung des MQTT-Clientdienstprogramms bei.

### Informationen zu diesem Vorgang

Die Task veranschaulicht das Senden einer Nachricht an einen MQTT-Client, indem sie nicht in einem Thema veröffentlicht, sondern in eine Warteschlange gestellt wird. Sie erstellen keine Subskription im Client. Schritt „2” auf Seite 136 in der Task veranschaulicht, dass die vorherige Subskription gelöscht wurde.

### Vorgehensweise

1. Verwerfen Sie alle vorhandenen Subskriptionen, indem Sie die Verbindung des MQTT-Clientdienstprogramms trennen und wiederherstellen.

   Sofern Sie die Standardeinstellungen nicht geändert haben, wird die Subskription verworfen, da das MQTT-Clientdienstprogramm mit einer bereinigten Sitzung eine Verbindung herstellt (siehe Sitzungen bereinigen).

   Geben Sie zur Vereinfachung der Task Ihren eigenen `ClientIdentifier`-Eintrag ein, statt den generierten `ClientIdentifier`-Eintrag zu verwenden, der vom MQTT-Clientdienstprogramm erstellt wurde.

   a) Klicken Sie auf **Verbindung trennen**, um die Verbindung zwischen dem MQTT-Clientdienstprogramm und dem Telemetriekanal zu trennen.

      Das **Clientprotokoll** enthält jetzt das Ereignis `Disconnected` (Verbindung getrennt).

   b) Ändern Sie die **Client-ID** in `MyClient`.

   c) Klicken Sie auf **Verbinden**.

   Das **Clientprotokoll** enthält jetzt das Ereignis `Connected` (Verbindung getrennt).
2. Stellen Sie sicher, dass das MQTT-Clientdienstprogramm keine Veröffentlichungen für das Thema `MQTTExampleTopic` mehr empfängt.
   a) Klicken Sie im IBM MQ Explorer auf den Ordner `Queue Managers\`*`QmgrName`*`\Topics`.
   b) Klicken Sie mit der rechten Maustaste auf `MQTTExampleTopic` > **Testveröffentlichung...**

c) Geben Sie `Hello World!` in das Feld **Nachrichtendaten** ein und klicken Sie auf **Nachricht veröffentlichen**. Wechseln zum Fenster des MQTT-Clientdienstprogramms.

Im **Clientprotokoll** wird kein Ereignis erfasst.

3. Erstellen Sie eine Definition einer fernen Warteschlange für den Client.

   Legen Sie den `ClientIdentifier`-Eintrag `MyClient` für den Namen des fernen Warteschlangenmanagers in der Definition einer fernen Warteschlange fest. Für den Namen der fernen Warteschlange kann ein beliebiger Name verwendet werden. Der Name der fernen Warteschlange wird als Themenname an einen MQTT-Client übergeben.

   a) Klicken Sie mit der rechten Maustaste auf den Ordner `Queue Managers\`*`QmgrName`*`\Queues` und klicken Sie anschießend auf **Neu** > **Definition der fernen Warteschlange**.

   b) Geben Sie der Definition den Namen `MyClientRemoteQueue` und klicken Sie auf **Weiter**.

   c) Geben Sie `MQTTExampleQueue` in das Feld **Ferne Warteschlange** ein.

   d) Geben Sie `MyClient` in das Feld **Ferner Warteschlangenmanager** ein.

   e) Geben Sie `SYSTEM.MQTT.TRANSMIT.QUEUE` im Feld **Übertragungswarteschlange** und klicken Sie auf **Fertigstellen**.

4. Reihen Sie eine Testnachricht in `MyClientRemoteQueue` ein.

   a) Klicken Sie mit der rechten Maustaste auf **MyClientRemoteQueue** > **Testnachricht einreihen...**

   b) Geben Sie `Hello queue!` in das Feld für Nachrichtendaten ein > **Nachricht einreihen** > **Schließen**

   Das **Clientprotokoll** enthält jetzt das Ereignis `Received` (Empfangen).

5. Entfernen Sie `SYSTEM.MQTT.TRANSMIT.QUEUE` als Standardübertragungswarteschlange.

   a) Klicken Sie mit der rechten Maustaste auf `Queue Managers\`*`QmgrName`* `folder` und klicken Sie auf **Eigenschaften ...**

   b) Klicken Sie im Navigator auf **Kommunikation**.

   c) Entfernen Sie `SYSTEM.MQTT.TRANSMIT.QUEUE` aus dem Feld **Standardübertragungswarteschlange** und klicken Sie auf **OK**.

6. Wiederholen Sie den Schritt „4” auf Seite 137.

   `MyClientRemoteQueue` ist eine Definition einer fernen Warteschlange, in der die Übertragungswarteschlange explizit genannt wird. Sie müssen keine Standardübertragungswarteschlange definieren, um eine Nachricht an `MyClient` zu senden.

## Nächste Schritte

Wenn die Standardübertragungswarteschlange nicht mehr auf `SYSTEM.MQTT.TRANSMIT.QUEUE`gesetzt ist, kann das MQTT -Clientdienstprogramm keine neue Subskription mehr erstellen, es sei denn, für `ClientIdentifier` `MyClient`ist ein Warteschlangenmanager-Aliasname definiert. Setzen Sie die Standardübertragungswarteschlange wieder auf `SYSTEM.MQTT.TRANSMIT.QUEUE`.

# Windows Linux AIX Nachricht an einen bestimmten MQTT v3-Client veröffentlichen

Veröffentlichen Sie eine Nachricht von einem MQTT v3-Client an einen anderen und verwenden Sie dabei `ClientIdentifier` als Themennamen und IBM MQ als Publish/Subscribe-Broker.

## Vorbereitende Schritte

Führen Sie den im Abschnitt „Nachricht über IBM MQ Explorer im MQTT -Clientdienstprogramm veröffentlichen” auf Seite 131 beschriebenen Schritt aus. Behalten Sie die Verbindung des MQTT-Clientdienstprogramms bei.

## Informationen zu diesem Vorgang

Die Task veranschaulicht zwei Punkte:

1. Die Subskription eines Themas in dem einen MQTT-Client und dem Empfang einer Veröffentlichung von einem anderen MQTT-Client.
2. Die Einrichtung von "Punkt-zu-Punkt"-Subskriptionen unter Verwendung von `ClientIdentifier` als Themenzeichenfolge.

## Vorgehensweise

1. Verwerfen Sie alle vorhandenen Subskriptionen, indem Sie die Verbindung des MQTT-Clientdienstprogramms trennen und wiederherstellen.

   Sofern Sie die Standardeinstellungen nicht geändert haben, wird die Subskription verworfen, da das MQTT-Clientdienstprogramm mit einer bereinigten Sitzung eine Verbindung herstellt (siehe Sitzungen bereinigen).

   Geben Sie zur Vereinfachung der Task Ihren eigenen `ClientIdentifier`-Eintrag ein, statt den generierten `ClientIdentifier`-Eintrag zu verwenden, der vom MQTT-Clientdienstprogramm erstellt wurde.

   a) Klicken Sie auf **Verbindung trennen**, um die Verbindung zwischen dem MQTT-Clientdienstprogramm und dem Telemetriekanal zu trennen.

      Das **Clientprotokoll** enthält jetzt das Ereignis `Disconnected` (Verbindung getrennt).

   b) Ändern Sie die **Client-ID** in `MyClient`.

   c) Klicken Sie auf **Verbinden**.

   Das **Clientprotokoll** enthält jetzt das Ereignis `Connected` (Verbindung getrennt).
2. Erstellen Sie eine Subskription für das Thema `MyClient`.

   `MyClient` ist der `ClientIdentifier`-Eintrag dieses Clients.

   a) Geben Sie `MyClient` in das Feld **Subscription\Topic** ein und klicken Sie auf **Subskribieren**.

   Das **Clientprotokoll** enthält jetzt das Ereignis `Subscribed` (Verbunden).
3. Starten Sie ein anderes MQTT-Clientdienstprogramm.

   a) Öffnen Sie den Ordner `Queue Managers\`*`QmgrName`*`\Telemetry\channels`.

   b) Klicken Sie mit der rechten Maustaste auf den Kanal **PlainText** und klicken Sie auf **MQTT-Clientdienstprogramm ausführen...**

   c) Klicken Sie auf **Verbinden**.

   Das **Clientprotokoll** enthält jetzt das Ereignis `Connected` (Verbindung getrennt).
4. Veröffentlichen Sie `Hello MyClient!` im Thema `MyClient`.

   a) Kopieren Sie das Subskriptionsthema `MyClient` aus dem MQTT-Clientdienstprogramm, das mit dem `ClientIdentifier`-Eintrag `MyClient` ausgeführt wird.

   b) Fügen Sie `MyClient` in das Feld **Publication\Topic** aller MQTT-Clientdienstprogramminstanzen ein.

   c) Geben Sie `Hello MyClient!` in das Feld **Publication\message** ein.

   d) Klicken Sie in beiden Instanzen auf **Veröffentlichen**.

## Ergebnisse

Das **Clientprotokoll** im MQTT-Clientdienstprogramm mit dem `ClientIdentifier`-Eintrag `MyClient` enthält zwei Ereignisse des Typs **Received** (Empfangen) und ein Ereignis des Typs **Published** (Veröffentlicht). Die andere Instanz des MQTT-Clientdienstprogramms dokumentiert ein Ereignis des Typs **Published** (Veröffentlicht).

Wenn Sie nur ein Ereignis des Typs **Received** (Empfangen) sehen, prüfen Sie folgende mögliche Ursachen:

1. Ist die Standardübertragungswarteschlange für den Warteschlangenmanager auf SYSTEM.MQTT.TRANSMIT.QUEUE gesetzt?
2. Haben Sie bei der Durchführung der anderen Übungen Warteschlangenmanager-Aliasnamen oder Definitionen einer fernen Warteschlange erstellt, die auf `MyClient` verweisen? Falls ein Konfigurationsproblem vorliegt, löschen Sie alle Ressourcen, die auf `MyClient` verweisen (beispielsweise Warteschlangenmanager-Aliasnamen oder Übertragungswarteschlangen). Trennen Sie die Verbindung der Clientdienstprogramme, stoppen Sie den Telemetrieservice (MQXR) und starten Sie ihn erneut.

## Windows Linux AIX Nachricht von einem MQTT -Client an eine IBM MQ -Anwendung senden

Eine IBM MQ-Anwendung kann eine Nachricht von einem MQTT v3-Client empfangen, indem sie ein Thema abonniert. Der MQTT-Client verbindet sich mit IBM MQ über einen Telemetriekanal und sendet eine Nachricht an die IBM MQ-Anwendung, indem er Informationen in demselben Thema veröffentlicht.

Führen Sie die Task „Nachricht von einem MQTT -Client in IBM MQ veröffentlichen” auf Seite 139 aus, um zu lernen, wie eine Veröffentlichung von einem MQTT-Client an eine Subskription gesendet wird, die in IBM MQ definiert ist.

Falls das Thema in Gruppen zusammengefasst ist oder unter Verwendung der Publish/Subscribe-Hierarchie verteilt wird, kann sich die Subskription auf einem anderen Warteschlangenmanager befinden als dem Warteschlangenmanager, mit dem der MQTT-Client verbunden ist.

## Windows Linux AIX Nachricht von einem MQTT -Client in IBM MQ veröffentlichen

Erstellen Sie eine Subskription für ein Thema unter Verwendung von IBM MQ Explorer und veröffentlichen Sie Informationen im Thema unter Verwendung des MQTT-Clientdienstprogramms.

### Vorbereitende Schritte

Führen Sie den im Abschnitt „Nachricht über IBM MQ Explorer im MQTT -Clientdienstprogramm veröffentlichen” auf Seite 131 beschriebenen Schritt aus. Behalten Sie die Verbindung des MQTT-Clientdienstprogramms bei.

### Informationen zu diesem Vorgang

Die Task veranschaulicht die Veröffentlichung einer Nachricht mit einem MQTT-Client und den Empfang der Veröffentlichung über eine nicht verwaltete permanente Subskription, die unter Verwendung von IBM MQ Explorer erstellt wird.

### Vorgehensweise

1. Erstellen Sie eine permanente Subskription für die Themenzeichenfolge `MQTT Example`.

   Führen Sie die folgenden Schritte zur Erstellung der Warteschlange und Subskription unter Verwendung von IBM MQ Explorer aus.

   a) Klicken Sie mit der rechten Maustaste auf den Ordner `Queue Managers\`*`QmgrName`*`\Queues` unter IBM MQ Explorer > **Neu** > **Lokale Warteschlange...**.

   b) Geben Sie `MQTTExampleQueue` als Warteschlangenname ein > **Fertigstellen**.

   c) Klicken Sie mit der rechten Maustaste auf den Ordner `Queue Managers\`*`QmgrName`*`\Subscriptions` unter IBM MQ Explorer > **Neu** > **Subskription...**.

   d) Geben Sie `MQTTExampleSubscription` als Warteschlangenname ein und klicken Sie auf **Weiter**.

   e) Klicken Sie auf **Auswählen...** > `MQTTExampleTopic` > **OK**.

   Sie haben das Thema `MQTTExampleTopic` bereits in Schritt „4” auf Seite 133 von „Nachricht über IBM MQ Explorer im MQTT -Clientdienstprogramm veröffentlichen” auf Seite 131 erstellt.

   f) Geben Sie `MQTTExampleQueue` als Bestimmungsname ein und klicken Sie auf **Fertigstellen**.

2. Sie können wahlweise die Warteschlange zur Verwendung eines anderen Benutzers ohne mqm-Berechtigung einrichten.

   Wenn Sie die Konfiguration für Benutzer mit einer niedrigeren Berechtigung als mqm einrichten, müssen Sie für MQTTExampleQueue die Berechtigungen put und get erteilen. Der Zugriff auf das Thema und die Übertragungswarteschlange wurde in „Nachricht über IBM MQ Explorer im MQTT -Clientdienstprogramm veröffentlichen" auf Seite 131 konfiguriert.

   a) Berechtigen Sie einen Benutzer für die Einreihung in die Warteschlange MQTTExampleQueue sowie für den Abruf aus dieser Warteschlange:

   ```
   setmqaut -m qMgrName -t queue -n MQTTExampleQueue -p User ID -all +put +get
   ```

3. Veröffentlichen Sie Hello IBM MQ! mithilfe des MQTT -Clientdienstprogramms im Thema MQTT Example .

   Wenn die Verbindung zum MQTT-Clientdienstprogramm nicht mehr besteht, klicken Sie mit der rechten Maustaste auf den **PlainText**-Kanal und wählen **MQTT-Clientdienstprogramm ausführen...** > **Verbinden** aus.

   a) Geben Sie MQTT Example in das Feld **Publication\Topic** ein.

   b) Geben Sie Hello IBM MQ! in das Feld **Publication\Message** ein und klicken Sie auf **Veröffentlichen**.

4. Öffnen Sie den Ordner Queue Managers*QmgrName*\Queues und suchen Sie den Eintrag MQTTExampleQueue.

   Das Feld **Aktuelle Warteschlangenlänge** hat den Wert 1

5. Klicken Sie mit der rechten Maustaste auf MQTTExampleQueue > **Nachrichten durchsuchen ...** und sehen Sie sich die Veröffentlichung an.

## Windows Linux AIX MQTT-Publish/Subscribe-Anwendungen

Verwenden Sie die themenbasierte Publish/Subscribe-Funktion zum Schreiben von MQTT-Anwendungen.

Wenn der MQTT-Client verbunden ist, fließen Veröffentlichungen zwischen dem Client und Server in beide Richtungen. Die Veröffentlichungen werden vom Client gesendet, sobald Informationen auf dem Client veröffentlicht werden. Veröffentlichungen werden beim Client empfangen, sobald eine Nachricht in einem Thema veröffentlicht wird, das einer vom Client erstellten Subskription entspricht.

Der Publish/Subscribe-Broker von IBM MQ verwaltet die Themen und Subskriptionen, die von MQTT -Clients erstellt wurden. Die von MQTT-Clients erstellten Themen nutzen denselben Themenbereich wie Themen, die von IBM MQ-Anwendungen erstellt werden.

Veröffentlichungen, die der Themenzeichenfolge in einer MQTT -Clientsubskription entsprechen, werden in SYSTEM.MQTT.TRANSMIT.QUEUE gestellt, wobei der Name des fernen Warteschlangenmanagers auf die ClientIdentifier des Clients gesetzt ist. Der Telemetrieservice (MQXR) leitet die Veröffentlichungen an den Client weiter, von dem die Subskription erstellt wurde. Er verwendet den ClientIdentifier-Eintrag, der als Name des fernen Warteschlangenmanagerrs festgelegt wurde, zur Identifizierung des Clients.

Für gewöhnlich muss SYSTEM.MQTT.TRANSMIT.QUEUE als Standardübertragungswarteschlange definiert werden. Es ist zwar aufwendig, aber möglich, MQTT so zu konfigurieren, dass nicht die Standardübertragungswarteschlange verwendet wird (siehe Verteilte Steuerung von Warteschlangen zum Senden von Nachrichten an MQTT-Clients verwenden).

Ein MQTT-Client kann eine persistente Sitzung erstellen (siehe „Statusunabhängige und statusbehaftete MQTT-Sitzungen" auf Seite 144). Subskriptionen, die in einer persistenten Sitzung erstellt werden, sind permanent. Veröffentlichungen, die für einen Client mit einer persistenten Sitzung eingehen, werden in der Warteschlange SYSTEM.MQTT.TRANSMIT.QUEUE gespeichert und an den Client weitergeleitet, wenn er die Verbindung wiederherstellt.

Ein MQTT-Client kann auch in ständigen Veröffentlichungen Informationen veröffentlichen und abonnieren (siehe Ständige Veröffentlichungen und MQTT-Clients). Ein Subskribent eines Themas in einer ständigen Veröffentlichung empfängt die neueste Veröffentlichung des Themas. Der Subskribent empfängt die ständige Veröffentlichung, wenn er eine Subskription erstellt oder die Verbindung zu seiner früheren Sitzung wiederherstellt.

## Windows Linux AIX Telemetrieanwendungen

Schreiben Sie Telemetrieanwendungen unter Verwendung von IBM MQ- oder IBM Integration Bus-Nachrichtenflüssen.

Verwenden Sie JMS, MQI oder sonstige IBM MQ-Programmierschnittstellen, um Telemetrieanwendungen in IBM MQ zu programmieren.

Der Telemetrieservice (MQXR) nimmt eine Konvertierung zwischen MQTT v3-Nachrichten und IBM MQ-Nachrichten vor. Er erstellt Subskriptionen und Veröffentlichungen für MQTT-Clients und leitet Veröffentlichungen an MQTT-Clients weiter. Eine Veröffentlichung sind die Nutzdaten einer MQTT v3-Nachricht. Die Nutzdaten umfassen Nachrichtenheader und eine Bytefeldgruppe im Format `jms-bytes`. Der Telemetrieserver ordnet die Header zwischen einer MQTT v3-Nachricht und einer IBM MQ-Nachricht zu (siehe „Integration von MQ Telemetry mit Warteschlangenmanagern” auf Seite 141).

Verwenden Sie die Publication-, MQInput- und JMSInput-Knoten, um Veröffentlichungen zwischen IBM Integration Bus und MQTT-Clients zu senden und zu empfangen.

Mithilfe von Nachrichtenflüssen können Sie die Telemetrie in Websites mit HTTP sowie in andere Anwendungen, die IBM MQ- und WebSphere-Adapter verwenden, integrieren.

## Windows Linux AIX Integration von MQ Telemetry mit Warteschlangenmanagern

Der MQTT-Client ist als Publish/Subscribe-Anwendung in IBM MQ integriert. Er kann Themen in IBM MQ entweder veröffentlichen oder abonnieren, neue Themen erstellen oder vorhandene Themen verwenden. Er empfängt Veröffentlichungen von IBM MQ, wenn MQTT-Clients (ihn selbst eingeschlossen) oder andere IBM MQ-Anwendungen Veröffentlichungen zu den Themen seiner Subskriptionen veröffentlichen. Bei der Festlegung der Attribute einer Veröffentlichung gelten bestimmte Regeln.

Viele der Attribute, die den von IBM MQ bereitgestellten Themen, Veröffentlichungen, Subskriptionen und Nachrichten zugeordnet sind, werden nicht unterstützt. Unter „MQTT-Client an IBM MQ-Publish/Subscribe-Broker” auf Seite 141 und „IBM MQ an einen MQTT-Client” auf Seite 143 wird beschrieben, wie Attribute von Veröffentlichungen festgelegt werden. Die Einstellungen hängen davon ab, ob die Veröffentlichung an den IBM MQ-Publish/Subscribe-Broker geleitet wird oder von diesem stammt.

In IBM MQ-Publish/Subscribe sind Themen mit administrativen Themenobjekten verknüpft. Dies gilt auch für Themen, die von MQTT-Clients erstellt werden. Wenn ein MQTT-Client eine Themenzeichenfolge für eine Veröffentlichung erstellt, ordnet der IBM MQ-Publish/Subscribe-Broker diese einem administrativen Themenobjekt zu. Der Broker ordnet die Themenzeichenfolge in der Veröffentlichung dem nächsten übergeordneten administrativen Themenobjekt zu. Die Zuordnung erfolgt auf die gleiche Weise wie bei IBM MQ-Anwendungen. Wenn kein Thema vorhanden ist, das von einem Benutzer erstellt wurde, wird das Veröffentlichungsthema `SYSTEM.BASE.TOPIC` zugeordnet. Die Attribute der Veröffentlichung leiten sich aus dem Themenobjekt ab.

Wenn eine IBM MQ-Anwendung oder ein Administrator eine Subskription erstellt, erhält die Subskription einen Namen. Listen Sie Subskriptionen mit IBM MQ Exploreroder mit **runmqsc** -oder PCF-Befehlen auf. Alle MQTT-Clientsubskriptionen haben einen Namen. Sie erhalten einen Namen im folgenden Format: *ClientIdentifier:Topic name*

### MQTT-Client an IBM MQ-Publish/Subscribe-Broker

Ein MQTT-Client hat eine Veröffentlichung an IBM MQ gesendet. Der Telemetrieservice (MQXR) konvertiert die Veröffentlichung in eine IBM MQ-Nachricht. Die IBM MQ-Nachricht enthält drei Teile:

1. MQMD
2. RFH2
3. Nachricht

MQMD-Eigenschaften werden auf ihre Standardwerte gesetzt, es sei denn, dies ist in Tabelle 9 auf Seite 142 anders angegeben.

*Tabelle 9. Einstellungen für MQMD-Eigenschaften*

| **MQMD-Feld** | **Typ** | **Wert** |
|---|---|---|
| **Format** | MQCHAR8 | MQFMT_RF_HEADER_2 |
| **UserIdentifier** | MQCHAR12 | Legen Sie einen der folgenden Werte fest:<br>MqttClient.ClientIdentifier<br>MqttConnectOptions.UserName<br>Eine Benutzer-ID, die vom IBM MQ-Administrator für den Telemetriekanal festgelegt wird |
| **Priority** | MQLONG | MQPRI_PRIORITY_AS_Q_DEF (Standardeinstellung bei IBM MQ, während JMS den Standardwert 4 hat) |
| **Persistence** | MQLONG | QoS=0→MQPER_NOT_PERSISTENT<br>QoS=1→MQPER_PERSISTENT<br>QoS=2→MQPER_PERSISTENT |

Der Header RFH2 enthält keinen <msd>-Ordner für die Definition des Typs der JMS-Nachricht. Der Telemetrieservice (MQXR) erstellt die IBM MQ-Nachricht als JMS-Standardnachricht. Der JMS-Standardnachrichtentyp ist eine jms-bytes-Nachricht. Eine Anwendung kann in Form von Nachrichteneigenschaften auf die zusätzlichen Kopfzeileninformationen zugreifen; siehe Nachrichteneigenschaften.

RFH2-Werte werden wie in Tabelle 10 auf Seite 142 dargestellt festgelegt. Die Eigenschaft 'Format' wird im festen RFH2-Header festgelegt, während die übrigen Werte in RFH2-Ordnern festgelegt werden.

*Tabelle 10. Einstellungen für RFH2-Eigenschaften*

| **RFH2-Eigenschaft** | **Typ/Ordner** | **Header** |
|---|---|---|
| Format | MQCHAR8 | MQFMT_NONE |
| ClientIdentifier | mqtt/clientId | Kopieren Sie MqttClient.ClientIdentifier mit einer Länge von 1...23 Bytes. |
| QoS | mqtt/qos | Kopieren Sie QoS aus der eingehenden MQTT-Nachricht. |
| Nachrichten-ID | mqtt/msgid | Kopieren Sie die Nachrichten-ID (Message ID) aus der eingehenden MQTT-Nachricht, wenn QoS den Wert 1 oder 2 hat. |
| MQIsRetained | mqps/Ret | Festgelegt, wenn die ursprüngliche MQTT-Veröffentlichung mit der Einstellung der Eigenschaft RETAIN gesendet wurde und die Nachricht als ständige Veröffentlichung empfangen wird. |
| MQTopicString | mqps/Top | Das Thema, in dem die MQTT-Nachricht veröffentlicht wurde. |

Die Nutzdaten in einer MQTT-Veröffentlichung werden dem Inhalt einer IBM MQ-Nachricht zugeordnet:

| *Tabelle 11. Zuordnung der Nutzdaten in einer MQTT-Veröffentlichung zu den IBM MQ-Nachrichteninhalten* | | |
|---|---|---|
| **Nachrichteninhalt(e)** | **Typ** | **Inhalt der IBM MQ-Nachricht** |
| `Buffer` | `MQBYTE` *n* | Kopie der Bytes der eingehenden MQTT-Nachricht. Die Länge kann null sein. |

## IBM MQ an einen MQTT-Client

Ein Client hat ein Veröffentlichungsthema abonniert. Eine IBM MQ -Anwendung hat zu dem Thema veröffentlicht, sodass eine Veröffentlichung vom IBM MQ Publish/Subscribe-Broker an den MQTT -Subskribenten gesendet wird. Möglich ist auch, dass eine IBM MQ-Anwendung eine nicht erwartete Nachricht direkt an einen MQTT-Client gesendet hat. In Tabelle 12 auf Seite 143 wird beschrieben, wie die festgelegten Nachrichtenheader in der Nachricht festgelegt werden, die an den MQTT-Client gesendet wird. Alle anderen Daten im IBM MQ-Nachrichtenheader sowie alle anderen Header werden gelöscht. Die Nachrichtendaten in der IBM MQ-Nachricht werden unverändert als Nachrichtennutzdaten in der MQTT-Nachricht gesendet. Die MQTT-Nachricht wird vom Telemetrieservice (MQXR) an den MQTT-Client gesendet.

| *Tabelle 12. Festlegen von festgelegten Nachrichtenheadern in einer IBM MQ -Nachricht, die an den MQTT -Client gesendet wird* | | |
|---|---|---|
| **Feld MQTT** | **Typ** | **Wert** |
| **DUP** | `Boolesch` | Festgelegt, wenn der Wert `QoS = 1` oder 2 eingestellt wurde und die Nachricht in einer früheren Übertragung an diesen Client gesendet, jedoch nach einer gewissen Zeit noch nicht bestätigt wurde. |
| **QoS** | `int` | Wie der Wert von `QoS` in einer ausgehenden Veröffentlichung vom Publish/Subscribe-Broker in IBM MQ festgelegt wird, hängt von der eingehenden Veröffentlichung ab. Hierbei ist relevant, ob die eingehende Veröffentlichung von einem MQTT-Client oder von einer IBM MQ-Anwendung gesendet wurde.<br>**MQTT**<br>Legen Sie in der eingehenden Veröffentlichung einen niedrigeren QoS-Wert (Servicequalität) fest. Dasselbe gilt für die vom Subskribenten angeforderte QoS.<br>**IBM MQ**<br>Verringern Sie den Wert der von der eingehenden Veröffentlichung abgeleiteten QoS:<br>`MQPER_NOT_PERSISTENT→QoS=0`<br>`MQPER_PERSISTENT→QoS=2`<br>Dasselbe gilt für die vom Subskribenten angeforderte QoS. Wenn die Nachricht ohne Subskription an den Client gesendet wird, wird die Servicequalität (QoS) standardmäßig auf 2 gesetzt. Ein Client kann diesen Wert ändern, indem er `DEFAULT.QoS` mit einer anderen QoS abonniert. |
| **RETAIN** | `Boolesch` | Festgelegt, wenn für die eingehende Veröffentlichung die Eigenschaft 'retained' (ständig) festgelegt wurde. |

In Tabelle 13 auf Seite 144 wird beschrieben, wie die variablen Nachrichtenheader in der MQTT-Nachricht festgelegt werden, die an den MQTT-Client gesendet wird.

*Tabelle 13. Wie variable MQTT-Headereigenschaften in einer an den MQTT-Client gesendeten MQTT-Nachricht festgelegt werde*

| Feld MQTT | Typ | Wert |
|---|---|---|
| **Topic name** | Zeichenfolge | Die Themenzeichenfolge, mit der die Nachricht veröffentlicht wurde. |
| **Message ID** | Zeichenfolge | Die letzten 2 Bytes der Eigenschaft MQMD.MsgId der Veröffentlichung, wenn sie in SYSTEM.MQTT.TRANSMIT.QUEUE gestellt wird. |
| **Payload** | byte[] | Direktkopie der Bytes aus der eingehenden Veröffentlichung an den Publish/Subscribe-Broker. Die Länge kann null sein. |

## Windows Linux AIX Statusunabhängige und statusbehaftete MQTT-Sitzungen

MQTT-Clients können eine statusbehaftete Sitzung mit dem Warteschlangenmanager erstellen. Wenn ein statusbehafteter MQTT-Client getrennt wird, behält der Warteschlangenmanager die vom Client erstellten Subskriptionen und unvollständig verarbeitete Nachrichten. Sobald der Client die Verbindung wiederherstellt, löst er unvollständig verarbeitete Nachrichten auf. Er sendet alle Nachrichten, die für die Zustellung eingereiht wurden und empfängt Nachrichten, die für seine Subskriptionen veröffentlicht wurden, solange er getrennt war.

Wenn ein MQTT-Client eine Verbindung zu einem Telemetriekanal herstellt, startet er entweder eine neue Sitzung oder setzt eine alte Sitzung fort. Eine neue Sitzung verfügt weder über noch nicht bestätigte, ausstehende Nachrichten, noch über Subskriptionen oder Veröffentlichungen, die zur Zustellung anstehen. Wenn ein Client eine Verbindung herstellt, gibt er an, ob der Start mit einer neuen Sitzung erfolgen oder eine bestehende Sitzung fortgesetzt werden soll (siehe Sitzungen bereinigen).

Wenn der Client eine bestehende Sitzung fortsetzt, erfolgt die Verarbeitung, als ob die Verbindung niemals unterbrochen worden wäre. Veröffentlichungen, die zur Zustellung anstehen, werden an den Client gesendet und alle noch nicht festgeschriebenen Nachrichtenübertragungen werden ausgeführt. Wenn ein Client in einer persistenten Sitzung die Verbindung zum Telemetrieservice (MQXR) trennt, bleiben alle Subskriptionen, die vom Client erstellt wurden, erhalten. Veröffentlichungen für die Subskriptionen werden an den Client gesendet, sobald dieser die Verbindung wiederherstellt. Wenn er die Verbindung wiederherstellt, ohne die alte Sitzung fortzusetzen, werden die Veröffentlichungen vom Telemetrieservice (MQXR) gelöscht.

Die Informationen zum Sitzungsstatus werden vom Warteschlangenmanager in der Warteschlange SYSTEM.MQTT.PERSISTENT.STATE gespeichert.

Der IBM MQ-Administrator kann eine Sitzung trennen und löschen.

## Windows Linux AIX MQTT-Client ist nicht verbunden

Wenn ein Client nicht verbunden ist, kann der Warteschlangenmanager für ihn weiterhin Veröffentlichungen empfangen. Diese werden an den Client weitergeleitet, sobald er die Verbindung wiederherstellt. Ein Client kann ein "Last Will and Testament" erstellen, das vom Warteschlangenmanager für den Client veröffentlicht wird, wenn die Verbindung des Clients unerwartet getrennt wird.

Wenn Sie bei einer unerwarteten Verbindungstrennung des Clients benachrichtigt werden möchten, können Sie eine Veröffentlichung des Typs 'Last Will and Testament' registrieren (siehe Veröffentlichung 'Last Will and Testament'. Sie wird vom Telemetrieservice (MQXR) gesendet, wenn dieser feststellt, dass die Verbindung zum Client ohne eine entsprechende Trennungsanforderung des Clients unterbrochen wurde.

Ein Client kann jederzeit eine ständige Veröffentlichung veröffentlichen (siehe Ständige Veröffentlichungen und MQTT-Clients). In einer neuen Subskription eines Themas kann angefordert werden, dass alle ständige Veröffentlichungen im Zusammenhang mit diesem Thema gesendet werden. Wenn Sie das Last

Will and Testament als ständige Veröffentlichung erstellen, können Sie mit dessen Hilfe den Status eines Clients überwachen.

Der Client veröffentlicht beispielsweise eine ständige Veröffentlichung, wenn er sich verbindet, in der seine Verfügbarkeit bekannt gemacht wird. Gleichzeitig erstellt er eine ständige Veröffentlichung des Typs 'Last Will and Testament', die seine Nichtverfügbarkeit ankündigt. Außerdem veröffentlicht er seine Nichtverfügbarkeit als ständige Veröffentlichung unmittelbar vor einem geplanten Verbindungsabbau. Um festzustellen, ob der Client verfügbar ist, müssen Sie das Thema der ständigen Veröffentlichung abonnieren. Sie erhalten in diesem Fall immer eine der drei Veröffentlichungen.

Wenn der Client veröffentlichte Nachrichten erhalten soll, wenn er getrennt ist, müssen Sie den Client mit seiner vorherigen Sitzung erneut verbinden; siehe „Statusunabhängige und statusbehaftete MQTT-Sitzungen” auf Seite 144. Seine Subskriptionen sind aktiv, bis sie gelöscht werden oder bis der Client eine neue (bereinigte) Sitzung erstellt.

## Windows Linux AIX Lose Kopplung zwischen MQTT-Clients und IBM MQ-Anwendungen

Der Fluss von Veröffentlichungen zwischen MQTT-Clients und IBM MQ-Anwendungen ist lose gekoppelt. Veröffentlichungen können entweder von einem MQTT-Client oder einer IBM MQ-Anwendung stammen und haben keine festgelegte Reihenfolge. Die Veröffentlichungskomponenten und Subskribenten sind lose verbunden. Sie interagieren indirekt über Veröffentlichungen und Subskriptionen. Sie können Nachrichten auch direkt von einer IBM MQ -Anwendung an einen MQTT -Client senden.

MQTT-Clients und IBM MQ-Anwendungen sind auf zwei Arten lose gekoppelt:

1. Veröffentlichungskomponenten und Subskribenten sind durch die Zuordnung einer Veröffentlichung und einer Subskription zu einem Thema lose verbunden. Veröffentlichungskomponenten und Subskribenten haben normalerweise keine Kenntnis von der Adresse oder Identität der anderen Quelle einer Veröffentlichung oder Subskription.
2. MQTT-Clients veröffentlichen, abonnieren, empfangen Veröffentlichungen und verarbeiten Empfangsbestätigungen in separaten Threads.

Eine MQTT-Clientanwendung wartet nicht, bis eine Veröffentlichung zugestellt wurde. Die Anwendung übergibt eine Nachricht an den MQTT-Client und setzt dann den eigenen Thread fort. Die Anwendung wird über ein Zustellungstoken mit der Zustellung einer Veröffentlichung synchronisiert (siehe Zustellungstokens).

Nach der Übergabe einer Nachricht an den MQTT-Client hat die Anwendung die Option, im Rahmen des Zustellungstokens zu warten. Statt zu warten, kann der Client eine Callback-Methode bereitstellen, die aufgerufen wird, wenn die Veröffentlichung an IBM MQ zugestellt wird. Das Zustellungstoken kann auch ignoriert werden.

Abhängig von der Servicequalität, die der Nachricht zugeordnet ist, wird das Zustellungstoken sofort oder möglicherweise auch nach einer langen Zeit an die Callback-Methode zurückgegeben. Das Zustellungstoken kann sogar zurückgegeben werden, nachdem die Verbindung des Clients getrennt und wiederhergestellt wurde. Bei der Servicequalität *fire and forget* wird das Zustellungstoken sofort zurückgegeben. In den anderen beiden Fällen wird das Zustellungstoken nur zurückgegeben, wenn der Client die Bestätigung erhält, dass die Veröffentlichung an Subskribenten gesendet wurde.

Veröffentlichungen, die als Folge einer Clientsubskription an einen MQTT-Client gesendet werden, werden an die Callback-Methode `messageArrived` übermittelt. `messageArrived` wird in einem anderen Thread als die Hauptanwendung ausgeführt.

### Nachrichten direkt an einen MQTT-Client senden

Sie können eine Nachricht auf eine von zwei Arten an einen bestimmten MQTT-Client senden.

1. Eine IBM MQ-Anwendung kann eine Nachricht ohne Subskription direkt an einen MQTT-Client senden (siehe Nachricht direkt an einen Client senden).

2. Eine alternative Methode ist die Verwendung Ihrer `ClientIdentifier`-Namenskonvention. Veranlassen Sie, dass alle MQTT-Subskribenten Subskriptionen unter Verwendung ihres eindeutigen `ClientIdentifier`-Eintrags als Thema verwenden. Nehmen Sie Veröffentlichungen für *ClientIdentifier* vor. Die Veröffentlichung wird an den Client gesendet, der das Thema *ClientIdentifier* abonniert hat. Mit diesem Verfahren können Sie eine Veröffentlichung an einen bestimmten MQTT-Subskribenten senden.

## Windows Linux AIX MQ Telemetry Sicherheit

Der Schutz von Telemetriegeräten kann wichtig sein, da die Geräte in der Regel tragbar sind und an Orten eingesetzt werden, die nicht genau kontrolliert werden können. Sie können die Verbindung zwischen dem MQTT-Gerät und dem Telemetrieservice (MQXR) mithilfe eines virtuellen privaten Netzes (VPN) schützen. MQ Telemetry bietet die beiden Sicherheitsmechanismen Transport Layer Security (TLS) und Java Authentication and Authorization Service (JAAS).

TLS wird in erster Linie zur Verschlüsselung der Kommunikation zwischen dem Gerät und dem Telemetriekanal und für die Authentifizierung der Geräteverbindung mit dem richtigen Server verwendet (siehe Telemetriekanalauthentifizierung mit TLS. Sie können mithilfe von TLS auch überprüfen, ob sich das Clientgerät mit dem Server verbinden darf (siehe MQTT-Clientauthentifizierung mit TLS).

Mit JAAS wird hauptsächlich überprüft, ob die Serveranwendung von dem Gerätebenutzer verwendet werden darf (siehe MQTT-Clientauthentifizierung mit einem Kennwort). JAAS kann in Verbindung mit LDAP verwendet werden, um ein Kennwort über ein Verzeichnis für einmalige Anmeldungen zu überprüfen.

Wenn TLS und JAAS kombiniert werden, wird die Authentifizierung über zwei Faktoren abgesichert. Sie können die von TLS verwendeten Chiffrierwerte auf Chiffrierwerte beschränken, welche die Federal Information Processing Standards erfüllen.

Mit Zehntausenden von Benutzern oder mehr ist die Bereitstellung einzelner Sicherheitsprofile nicht immer in der Praxis umsetzbar. Außerdem ist es häufig umständlich, die Profile für die Autorisierung einzelner Benutzer für den IBM MQ-Objekte zu verwenden. Gruppieren Sie stattdessen Benutzer in Klassen für die Autorisierung von Veröffentlichungen und Subskriptionen von Themen und für das Senden von Veröffentlichungen an Client.

Konfigurieren Sie jeden Telemetriekanal so, dass Clients zu allgemeinen Client-Benutzer-IDs zugeordnet werden. Verwenden Sie eine allgemeine Benutzer-ID für jeden Client, der sich auf einem bestimmten Kanal verbindet (siehe MQTT-Clientidentität und Autorisierung.

Die Autorisierung von Benutzergruppen umfasst keine Authentifizierung aller Einzelpersonen. Jeder einzelne Benutzer kann beim Client oder Server durch seinen `Benutzernamen` und das zugehörige `Kennwort` authentifiziert und dann mithilfe einer allgemeine Benutzer-ID beim Server autorisiert werden.

## Windows Linux AIX MQ Telemetry-Globalisierung

Die Nachrichtennutzdaten im MQTT v3-Protokoll werden als Bytefeldgruppe codiert. Für gewöhnlich erstellen Anwendungen, die den Text verarbeiten, die Nachrichtennutzdaten in UTF-8. Der Telemetriekanal beschreibt die Nachrichtennutzdaten als UTF-8, nimmt jedoch keine Codepagekonvertierungen vor. Die Themenzeichenfolge der Veröffentlichung muss UTF-8 sein.

Die Anwendung ist für die Konvertierung alphabetischer Daten in die richtige Codepage und für die Konvertierung numerischer Daten in die richtige Zahlencodierung verantwortlich.

Der MQTT Java-Client verfügt über eine geeignete `MqttMessage.toString`-Methode. Die Methode behandelt die Nachrichtennutzdaten, als ob sie in dem Standardzeichensatz der lokalen Plattform codiert wären; dies ist für gewöhnlich UTF-8. Sie konvertiert die Nutzdaten in eine Java-Zeichenfolge (String). Java verfügt über die Zeichenfolgemethode `getBytes` zur Konvertierung einer Zeichenfolge in eine Bytefeldgruppe, die unter Verwendung des Standardzeichensatzes der lokalen Plattform codiert wird. Zwei MQTT Java-Programme, die Text in den Nachrichtennutzdaten zwischen Plattformen mit demselben Standardzeichensatz austauschen, können hierbei auf einfache und effiziente Weise UTF-8 nutzen.

Wenn eine der Plattformen nicht den Standardzeichensatz UTF-8 hat, müssen die Anwendungen eine Konvention für den Austausch von Nachrichten erstellen. Die Veröffentlichungskomponente gibt beispielsweise eine Konvertierung aus einer Zeichenfolge in UTF-8 an, wobei die Methode `getBytes("UTF8")` verwendet wird. Zum Empfang des Nachrichtentextes geht der Subskribent davon aus, dass die Nachricht im Zeichensatz UTF-8 codiert ist.

Der Telemetrieservice (MQXR) beschreibt die Codierung aller eingehenden Veröffentlichungen von MQTT-Client als UTF-8. Er legt `MQMD.CodedCharSetId` zu UTF-8und `RFH2.CodedCharSetId` zu `MQCCSI_INHERIT` ; Siehe „Integration von MQ Telemetry mit Warteschlangenmanagern” auf Seite 141. Da das Format der Veröffentlichung auf `MQFMT_NONE` gesetzt ist, kann keine Konvertierung durch Kanäle oder `MQGET` erfolgen.

## Windows Linux AIX Leistung und Skalierbarkeit von MQ Telemetry

Beachten Sie die folgenden Faktoren bei der Verwaltung einer großen Anzahl an Clients und der Verbesserung der Skalierbarkeit von MQ Telemetry.

### Kapazitätsplanung

Informationen zu Leistungsberichten für MQ Telemetry finden Sie unter MQ Performance documents.

### Verbindungen

Verbindungen bringen unter anderem folgenden Aufwand mit sich:

- Den Einrichtungsaufwand für die Verbindung selbst im Hinblick auf die Prozessorbelegung und die Dauer
- Den Netzaufwand
- Die Speicherbelegung, wenn eine Verbindung offen gehalten, jedoch nicht verwendet wird.

Wenn Clients verbunden bleiben, besteht eine zusätzliche Belastung. Wird eine Verbindung offen gehalten, verwenden TCP/IP-Verarbeitungsabläufe und MQTT-Nachrichten das Netz, um zu prüfen, ob die Verbindung immer noch besteht. Außerdem wird für jede Clientverbindung, die offen gehalten wird, Serverspeicher belegt.

Wenn Sie mehr als eine Nachricht pro Minute senden, halten Sie Ihre Verbindung offen, um den Aufwand für den Aufbau einer neuen Verbindung zu vermeiden. Wenn Sie weniger als eine alle 10 bis 15 Minuten senden, sollten Sie erwägen, die Verbindung zu trennen, um den Aufwand zu vermeiden, der entsteht, wenn sie offen gehalten wird. Es kann sinnvoll sein, über längere Zeiträume eine TLS-Verbindung offen, aber inaktiv zu halten, da ihr Aufbau sehr aufwändig ist.

Berücksichtigen Sie außerdem die Funktionalität des Clients. Wenn der Client über die Funktion eines Store-and-forward-Verfahrens verfügt, kann es hilfreich sein, Nachrichten zu stapeln und die Verbindung zwischen dem Versenden der Stapel zu trennen. Wird die Verbindung des Clients jedoch getrennt, kann er keine Nachrichten vom Server empfangen. Der Zweck Ihrer Anwendung beeinflusst daher Ihre Entscheidung.

Wenn Ihr System über einen Client verfügt, der viele Nachrichten sendet (beispielsweise bei Dateiübertragungen), warten Sie nicht bei jeder Nachricht auf eine Serverantwort. Senden Sie stattdessen alle Nachrichten und prüfen Sie am Ende, ob alle empfangen wurden. Sie können auch die Methode der Servicequalität (Quality of Service, QoS) nutzen.

Sie können die Servicequalität je nach Nachricht variieren, indem unwichtige Nachrichten mit der Servicequalität 0 und wichtige Nachrichten mit der Servicequalität 2 übermittelt werden. Der Nachrichtendurchsatz kann bei der Servicequalität 0 etwa doppelt so hoch sein wie bei der Servicequalität 2.

### Namenskonventionen

Wenn Sie Ihre Anwendung für viele Clients entwerfen, implementieren Sie eine effektive Namenskonvention. Damit jedem Client die richtige `Client-ID` zugeordnet wird, verwenden Sie eine aussagekräftige

`Client-ID`. Durch eine gute Namenskonvention lässt sich für den Administrator leichter feststellen, welche Clients gerade ausgeführt werden. Eine Namenskonvention ermöglicht dem Administrator das Filtern einer langen Liste mit Clients in IBM MQ Explorer und vereinfacht die Problembestimmung (siehe Client-ID).

## Durchsatz

Die Länge der Themennamen wirkt sich auf die Anzahl der Bytes aus, die über das Netz fließen. Beim Veröffentlichen oder Abonnieren kann die Anzahl der Bytes in einer Nachricht eine wichtige Rolle spielen. Begrenzen Sie daher die Anzahl der Zeichen in einem Themennamen. Wenn ein MQTT-Client ein Thema abonniert, gibt ihm IBM MQ einen Namen in folgendem Format:

```
ClientIdentifier: TopicName
```

Mit dem IBM MQ MQSC **DISPLAY** -Befehl können Sie alle Subskriptionen für einen MQTT -Client anzeigen:

```
DISPLAY SUB(' ClientID1:*')
```

## Ressourcen in IBM MQ zur Verwendung durch MQTT-Clients definieren

Ein MQTT-Client stellt eine Verbindung zu einem fernen IBM MQ-Warteschlangenmanager her. Es gibt zwei grundlegende Methoden, mit denen eine IBM MQ -Anwendung Nachrichten an einen MQTT -Client senden kann: Setzen Sie die Standardübertragungswarteschlange auf `SYSTEM.MQTT.TRANSMIT.QUEUE` oder verwenden Sie Warteschlangenmanager-Aliasnamen. Definieren Sie bei sehr vielen MQTT-Clients die Standardübertragungswarteschlange eines Warteschlangenmanagers. Durch die Einstellung einer Standardübertragungswarteschlange verringert sich der Verwaltungsaufwand (siehe Verteilte Steuerung von Warteschlangen zum Senden von Nachrichten an MQTT-Clients konfigurieren).

## Skalierbarkeit durch das Vermeiden von Subskriptionen verbessern

Wenn ein MQTT V3-Client ein Thema abonniert, wird vom Telemetrieservice (MQXR) in IBM MQ eine Subskription erstellt. Die Subskription leitet Veröffentlichungen für den Client an die Warteschlange `SYSTEM.MQTT.TRANSMIT.QUEUE` weiter. Der Name des fernen Warteschlangenmanagers im Übertragungsheader jeder Veröffentlichung wird auf die `Client-ID` des MQTT-Clients gesetzt, der die Subskription erstellt hat. Wenn viele Clients vorhanden sind, die alle ihre eigenen Subskriptionen erstellen, führt dies dazu, dass viele Proxy-Subskriptionen im gesamten Publish/Subscribe-Cluster oder in der Hierarchie von IBM MQ verwaltet werden müssen. Im Abschnitt Nachricht direkt an einen Client senden finden Sie Informationen darüber, wie Sie anstelle von Publish/Subscribe eine Punkt-zu-Punkt-basierte Lösung verwenden können.

## Sehr viele Clients verwalten

Wenn Sie viele gleichzeitig verbundene Clients unterstützen müssen, erhöhen Sie den Speicher, der für den Telemetrieservice (MQXR) zur Verfügung steht. Legen Sie hierfür die JVM-Parameter **-Xms** und **-Xmx** fest. Führen Sie folgende Schritte aus:

1. Suchen Sie die Datei `java.properties` im Konfigurationsverzeichnis des Telemetrieservice (siehe Konfigurationsverzeichnis des Telemetrieservice (MQXR) unter Windows bzw. Konfigurationsverzeichnis des Telemetrieservice unter Linux).
2. Folgen Sie den Anweisungen in der Datei; ein Heapspeicher von 1 GB reicht für 50.000 gleichzeitig verbundene Clients aus.

   ```
   # Heap sizing options - uncomment the following lines to set the heap to 1G
   #-Xmx1024m
   #-Xms1024m
   ```

3. Fügen Sie in der Datei `java.properties` weitere Befehlszeilenargumente für die Übergabe an die Java Virtual Machine, auf der der Telemetrieservice (MQXR) ausgeführt wird, hinzu (siehe JVM-Parameter an den Telemetrieservice (MQXR) übergeben).

Wenn Sie die Anzahl der offenen Dateideskriptoren unter Linux erhöhen möchten, fügen Sie `/etc/security/limits.conf/` folgende Zeilen hinzu und melden Sie sich wieder an.

```
@mqm soft nofile 65000
@mqm hard nofile 65000
```

Für jeden Socket ist ein Dateideskriptor erforderlich. Da der Telemetrieservice einige zusätzliche Dateideskriptoren benötigt, muss diese Zahl größer als die Anzahl der erforderlichen offenen Sockets sein.

Der Warteschlangenmanager verwendet für jede nicht dauerhafte Subskription eine Objektkennung. Wenn Sie viele aktive, nicht dauerhafte Subskriptionen unterstützen möchten, erhöhen Sie die maximale Anzahl aktiver Kennungen im Warteschlangenmanager. Beispiel:

```
echo ALTER QMGR MAXHANDS(999999999) | runmqsc qMgrName
```

*Abbildung 48. Maximale Anzahl an Kennungen unter Windows ändern*

```
echo "ALTER QMGR MAXHANDS(999999999)" | runmqsc qMgrName
```

*Abbildung 49. Maximale Anzahl an Kennungen unter Linux ändern*

### Weitere Überlegungen

Berücksichtigen Sie bei der Planung Ihrer Systemvoraussetzungen den Zeitaufwand, der für den Neustart des Systems benötigt wird. Die geplante Nichtverfügbarkeit kann sich auf die Anzahl gestauter Nachrichten auswirken, die zur Verarbeitung anstehen. Konfigurieren Sie das System so, dass die Nachrichten in einer angemessenen Zeit erfolgreich verarbeitet werden können. Überprüfen Sie den Plattenspeicher, Hauptspeicher und die Verarbeitungskapazität. Bei einigen Clientanwendungen können bei einer Wiederherstellung der Clientverbindung möglicherweise Nachrichten gelöscht werden. Wenn Nachrichten gelöscht werden sollen, legen Sie `CleanSession` in den Parametern der Clientverbindung fest (siehe Sitzungen bereinigen). Alternativ können Sie die Veröffentlichung und Subskription mit der bestmöglichen Servicequalität ( `0`) in einem MQTT -Client durchführen (siehe Servicequalität). Verwenden Sie `nicht persistente` Nachrichten, wenn Sie Nachrichten von IBM MQsenden. Nachrichten mit diesen Servicequalitäten werden bei einem Neustart des Systems oder Verbindung nicht wiederhergestellt.

## Windows Linux AIX Von MQ Telemetry unterstützte Geräte

MQTT-Clients können auf einer ganzen Reihe von Geräten ausgeführt werden, von Sensoren über Aktuatoren bis hin zu Handbediengeräten und Fahrzeugsystemen.

MQTT-Clients sind klein und werden auf Geräten ausgeführt, die durch wenig Speicherplatz und niedrige Verarbeitungskapazität eingeschränkt sind. Das MQTT protocol ist zuverlässig und verfügt über kompakte Header, wodurch es sich für Netze eignet, die durch eine geringe Bandbreite, hohen Aufwand und nicht unterbrechungsfreie Verfügbarkeit eingeschränkt sind.

MQ Telemetry kommuniziert über MQTT-Clientanwendungen mit Telemetriegeräten. Diese Anwendungen verwenden die folgenden Ressourcen, die alle das MQTT v3-Protokoll implementieren:

- Die folgenden Clientbibliotheken:
  - *MQTT client for Java* für die Erstellung nativer Anwendungen für beispielsweise Android-, OS X-, Linux- oder Windows-Geräte. Anwendungen, die diese Clientbibliothek verwenden, können alle Varianten von Java von der kleinsten CLDC (Connected Limited Device Configuration)/MIDP (Mobile Information Device Profile) über CDC (Connected Device Configuration)/Foundation, J2SE Java Platform, Standard Edition) und J2EE Java Platform, Enterprise Edition) ausführen. Die angepasste jclRM-

Klassenbibliothek von IBM wird ebenfalls unterstützt. Die Java ME-Plattform wird im Allgemeinen auf kompakten Endgeräten wie Aktuatoren, Sensoren, Mobiltelefonen und sonstigen integrierten Einheiten verwendet. Die Java SE-Plattform ist im Allgemeinen auf integrierten High-End-Einheiten wie Desktop-Computern und Servern installiert.

- *MQTT client for C* für die Erstellung nativer Anwendungen für beispielsweise iOS-, OS X-, Linux- oder Windows-Geräte. Diese Clientbibliothek stellt eine Referenzimplementierung in der Programmiersprache C in Kombination mit dem vordefinierten nativen Client für Windows- und Linux-Systeme bereit. Die Referenzimplementierung in der Programmiersprache C ermöglicht die Portierung von MQTT auf eine breite Palette an Geräten und Plattformen. Einige Windows-Systeme auf Intel-Plattformen, einschließlich Windows 7, RedHat, Ubuntu, und einige Linux-Systeme auf ARM-Plattformen wie Eurotech Viper implementieren Versionen von Linux, die den C-Client ausführen, aber IBM stellt für die Plattformen keine Serviceunterstützung bereit. Sie müssen Probleme im Zusammenhang mit dem Client auf einer unterstützten Plattform reproduzieren, wenn Sie sich an Ihr IBM Support Center wenden möchten.
- *MQTT client for Java* für die Erstellung browserbasierter Webanwendungen.

MQTT-Clientbibliotheken sind bei Eclipse Paho und MQTT.org kostenlos erhältlich. Weitere Informationen finden Sie im Abschnitt IBM MQ Telemetry Transport-Beispielprogramme.

# Sicherheit in IBM MQ

In IBM MQ gibt es mehrere Möglichkeiten, für Sicherheit zu sorgen: die Berechtigungsserviceschnittstelle, Kanalexits (benutzerdefiniert oder von einem anderen Hersteller), Kanalsicherheit mit Transport Layer Security (TLS), Kanalauthentifizierungsdatensätze und Nachrichtensicherheit.

## Schnittstelle für Berechtigungsservice

Die Berechtigung für die Verwendung von MQI-Aufrufen und Befehlen und den Zugriff auf Objekte wird vom **Objektberechtigungsmanager** erteilt, der standardmäßig aktiviert ist. Der Zugriff auf IBM MQ-Entitäten wird über IBM MQ-Benutzergruppen und den Objektberechtigungsmanager gesteuert. Administratoren können Berechtigungen über eine Befehlszeilenschnittstelle nach Bedarf erteilen oder entziehen.

Weitere Informationen zum Erstellen von Berechtigungsservicekomponenten finden Sie unter Sicherheit auf AIX, Linux, and Windows-Systemen einrichten.

## Benutzerdefinierte Kanalexits und Kanalexits anderer Anbieter

Kanäle können benutzerdefinierte Kanalexits oder Kanalexits anderer Anbieter verwenden. Weitere Informationen finden Sie im Abschnitt Kanalexitprogramme für Messaging-Kanäle.

## Kanalsicherheit mit TLS

Das TLS-Protokoll (Transport Layer Security) sorgt für standardisierte Kanalsicherheit, indem es Schutz vor Abhören, Manipulation und Identitätsvortäuschung bietet.

Bei TLS werden der Nachrichtenschutz sowie Integrität und gegenseitige Authentifizierung mithilfe von öffentlichen Schlüsseln und symmetrischer Verschlüsselung implementiert.

Umfassende Informationen zur Sicherheit in IBM MQ, darunter auch ausführliche Hinweise zu TLS, finden Sie im Abschnitt Sicherheit. Eine Übersicht über TLS mit Hinweisen zu den in diesem Abschnitt beschriebenen Befehlen finden Sie im Abschnitt Kryptografische Sicherheitsprotokolle: TLS.

## Kanalauthentifizierungsdatensätze

Mit Kanalauthentifizierungsdatensätzen können Sie den Zugriff auf Systeme, die eine Verbindung herstellen können, auf Kanalebene steuern. Weitere Informationen finden Sie im Abschnitt Kanalauthentifizierungsdatensätze.

### Nachrichtensicherheit

Verwenden Sie Advanced Message Security eine gesondert installierte und lizenzierte Komponente von IBM MQ, um mit IBM MQ gesendete und empfangene Nachrichten mit einem kryptografischen Schutz zu versehen. Weitere Informationen finden Sie in Advanced Message Security.

**Zugehörige Tasks**

Sicherung

Sicherheitsanforderungen planen

## TLS-Unterstützung für verwalteten IBM MQ.NET-Client

Der vollständig verwaltete IBM MQ.NET-Client bietet TLS-Unterstützung (Transport Layer Security) basierend auf dem Kit 'Microsoft.NET SSLStreams'. Dies unterscheidet sich von den anderen IBM MQ -Clients, die auf IBM Global Security Kit (GSKit)basieren.

Sie können IBM MQ.NET-Anwendungen für die Ausführung im verwalteten und im nicht verwalteten Modus entwickeln.

- Im verwalteten Modus agieren .NET-Anwendungen in der .NET-CLR (Common Language Runtime) ohne plattformübergreifenden Aufruf wie beispielsweise den Aufruf des Message Queue Interface in der Programmiersprache C.
- Im nicht verwalteten Modus wird das Message Queue Interface (MQI) in der Programmiersprache C für die zugrunde liegenden MQI-Operationen aufgerufen. Im Wesentlichen umfasst die Schnittstelle im nicht verwalteten Modus die .NET-Wrapperklassen, die auf dem Message Queue Interface in der Programmiersprache C aufsetzen.

Der verwaltete IBM MQ.NET-Client verwendet die Microsoft.NET-Frameworkbibliotheken für die Implementierung von TLS-Secure-Socket-Protokollen. Die Klasse 'System.NET.Security.SSLStream' von Microsoft wird für die Implementierung der Sicherheit (TLS) in IBM MQ.NET verwendet.

Der nicht verwaltete IBM MQ.NET -Clientmodus unterstützt bereits die TLS-Funktion, die auf C MQI (und GSKit) basiert. TLS-Operationen werden also vom Message Queue Interface in der Programmiersprache C verarbeitet. In diesem Fall implementiert GSKit die sicheren TLS-Socketprotokolle.

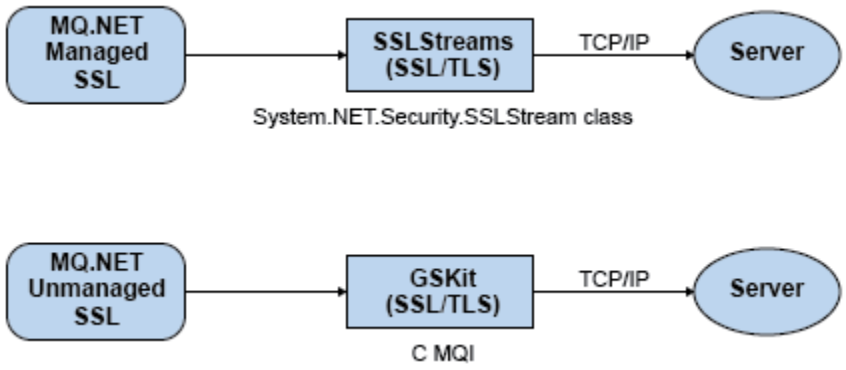



*Abbildung 50. Vergleich von IBM MQ.NET-TLS im verwalteten und nicht verwalteten Modus*

Die folgende Tabelle enthält eine Zusammenfassung der Unterschiede zwischen den verwalteten und nicht verwalteten Implementierungen:

| *Tabelle 14. Unterschiede zwischen verwalteten und nicht verwalteten Implementierungen* | | | |
|---|---|---|---|
| **Modus** | **Protokolle** | **Implementierung** | **Kommentare** |
| IBM MQ.NET SSL verwaltet | TLS | Klasse System.NET.Security.SSLStream<br>Die SSLStream-Klasse agiert als Datenstrom über ein verbundenes TCP-Socket. | TLS 1.0<br>TLS 1.2 (nur mit Microsoft.NET Framework v4.5) |
| IBM MQ.NET SSL nicht verwaltet | TLS | Global Security Kit und Message Queue Interface in der Programmiersprache C | Sichere TLS-Socketprotokolle |

**Zugehörige Konzepte**

Unterstützung von Secure Sockets Layer (SSL) und Transport Layer Security (TLS) für .NET

# IBM MQ MQI clients

Ein IBM MQ MQI client ist eine Komponente des IBM MQ-Produkts, die auf einem System installiert werden kann, auf dem kein Warteschlangenmanager aktiv ist.

Ein IBM MQ MQI-*Client* ist eine Komponente, die es einer auf einem System aktiven Anwendung ermöglicht, MQI-Aufrufe an Warteschlangenmanager auf einem anderen System abzusetzen. Die Ausgabe eines Aufrufs wird an den Client zurückgesendet, der sie an die Anwendung weitergibt.

Mit einem IBM MQ MQI client kann eine Anwendung, die auf demselben System wie der Client aktiv ist, eine Verbindung zu einem Warteschlangenmanager auf einem anderen System herstellen. Dadurch kann die Anwendung MQI-Aufrufe an diesen Warteschlangenmanager absetzen. Diese Anwendungen werden als IBM MQ MQI clientanwendungen, der Warteschlangenmanager als *Server-Warteschlangenmanager* bezeichnet.

Ein IBM MQ-*Server* ist ein Warteschlangenmanager, der für einen oder mehrere Clients Services zur Steuerung von Warteschlangen bereitstellt. Alle IBM MQ-Objekte (wie beispielsweise Warteschlangen) sind nur auf der Warteschlangenmanager-Maschine (IBM MQ-Servermaschine) vorhanden, nicht auf dem Client. Ein IBM MQ-Server kann auch lokale IBM MQ-Anwendungen unterstützen.

Ein IBM MQ-Server unterscheidet sich von einem normalen Warteschlangenmanager darin, dass zwischen dem Server und den einzelnen Clients dedizierte Kommunikationsverbindungen bestehen. Weitere Informationen zur Erstellung von Kanälen für Clients und Server finden Sie im Abschnitt Verteilte Steuerung von Warteschlangen konfigurieren.

IBM MQ MQI clientanwendungen und Server-Warteschlangenmanager kommunizieren über *MQI-Kanäle* miteinander. Ein MQI-Kanal wird gestartet, wenn die Clientanwendung einen **MQCONN**- oder **MQCONNX**-Aufruf absetzt, um eine Verbindung zum Warteschlangenmanager herzustellen, und endet, wenn die Clientanwendung einen **MQDISC**-Aufruf ausgibt, um die Verbindung zum Warteschlangenmanager zu beenden. Die Eingabeparameter eines MQI-Aufrufs werden im MQI-Kanal in eine Richtung, die Ausgabeparameter in die entgegengesetzte Richtung übertragen.

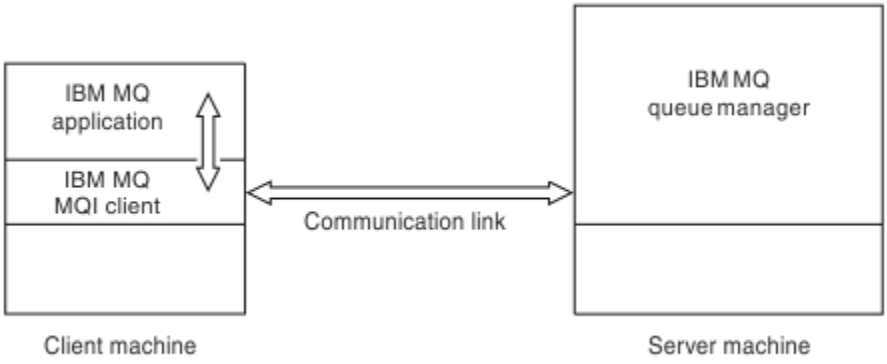



*Abbildung 51. Verbindung zwischen Client und Server*

Folgende Plattformen können eingesetzt werden. Die zulässigen Kombinationen hängen von dem von Ihnen verwendeten IBM MQ-Produkt ab und sind im Abschnitt „Plattformunterstützung für IBM MQ-Clients" auf Seite 154 beschrieben.

| **IBM MQ MQI client** | **IBM MQ-Server** |
|---|---|
| AIX and Linux<br>Windows<br>IBM i | AIX and Linux<br>Windows<br>IBM i<br>z/OS |

Die MQI ist für alle auf der Clientplattform aktiven Anwendungen verfügbar; die Warteschlangen und andere IBM MQ-Objekte befinden sich in einem von Ihnen auf einem Server installierten Warteschlangenmanager.

Eine Anwendung, die in der IBM MQ MQI clientumgebung eingesetzt werden soll, muss zunächst mit der entsprechenden Clientbibliothek verbunden werden. Wenn die Anwendung einen MQI-Aufruf ausgibt, leitet der IBM MQ MQI client die Anforderung an einen Warteschlangenmanager weiter, von dem sie verarbeitet und eine Antwort an den IBM MQ MQI client zurückgegeben wird.

Die Verbindung zwischen Anwendung und IBM MQ MQI client wird zur Ausführungszeit dynamisch hergestellt.

Sie können Clientanwendungen auch mithilfe von IBM MQ classes for .NET, IBM MQ classes for Java oder IBM MQ classes for Java Message Service (JMS) entwickeln. Sie können Java- und JMS-Clients auf folgenden Plattformen verwenden:

- IBM i
- AIX
- Linux
- Windows

Auf die Verwendung von Java und JMS wird hier nicht näher eingegangen. Vollständige Informationen zur Installation, Konfiguration und Verwendung von IBM MQ classes for Java und IBM MQ classes for JMS finden Sie in den Abschnitten IBM MQ classes for Java verwenden und IBM MQ classes for JMS verwenden.

## IBM MQ-Anwendungen in einer Client/Server-Umgebung

Wenn eine Verbindung zu einem Server besteht, können IBM MQ-Clientanwendungen die meisten MQI-Aufrufe auf dieselbe Weise wie lokale Anwendungen ausgeben. Die Clientanwendung gibt einen MQCONN-

Aufruf aus, um eine Verbindung zu einem angegebenen Warteschlangenmanager herzustellen. Alle weiteren MQI-Aufrufe, in denen die von der Verbindungsanforderung zurückgegebene Verbindungskennung angegeben ist, werden von diesem Warteschlangenmanager verarbeitet.

Sie müssen die Anwendungen mit den entsprechenden Clientbibliotheken verbinden. Weitere Informationen finden Sie unter Anwendungen für IBM MQ MQI clients erstellen.

**Zugehörige Konzepte**

„Verwendung von IBM MQ-Clients" auf Seite 154
Mit IBM MQ-Clients können IBM MQ-Nachrichtenübertragung und -Warteschlangensteuerung auf effiziente Weise implementiert werden.

„Was ist ein erweiterter Transaktionsclient?" auf Seite 156
Ein erweiterter transaktionsorientierter IBM MQ-Client kann unter der Steuerung eines externen Transaktionsmanagers Ressourcen aktualisieren, die von einem anderen Ressourcenmanager verwaltet werden.

„Verbindung vom Client zum Server herstellen" auf Seite 157
Clients stellen über MQCONN- oder MQCONNX-Aufrufe eine Verbindung zum Server her; die Kommunikation erfolgt über einen Kanal.

„Transaktionsmanagement und -unterstützung" auf Seite 158
Dieser Abschnitt gibt eine Einführung in das Transaktionsmanagement und es wird erläutert, wie Transaktionen in IBM MQ unterstützt werden.

„Funktionen des Warteschlangenmanagers erweitern" auf Seite 160
Sie können die Funktionen eines Warteschlangenmanagers durch Benutzerexits, API-Exits oder installierbare Services erweitern.

**Zugehörige Informationen**

IBM MQ MQI client einrichten

# Verwendung von IBM MQ-Clients

Mit IBM MQ-Clients können IBM MQ-Nachrichtenübertragung und -Warteschlangensteuerung auf effiziente Weise implementiert werden.

Sie können eine Anwendung einsetzen, die eine MQI und einen Warteschlangenmanager verwendet, die sich beide (physisch oder virtuell) auf unterschiedlichen Maschinen befinden. Hier die Vorteile eines solchen Szenarios:

- Es ist keine vollständige IBM MQ-Implementierung auf der Clientmaschine erforderlich.
- Die Hardwarevoraussetzungen für das Clientsystem werden reduziert.
- Die Anforderungen bezüglich der Systemverwaltung werden reduziert.
- Eine auf einem Client aktive IBM MQ-Anwendung kann eine Verbindung zu mehreren Warteschlangenmanagern auf unterschiedlichen Systemen herstellen.
- Es können alternative Kanäle mit unterschiedlichen Übertragungsprotokollen verwendet werden.

## Plattformunterstützung für IBM MQ-Clients

IBM MQ auf allen unterstützten Serverplattformen akzeptiert Clientverbindungen von IBM MQ MQI clients auf einer Reihe von Plattformen.

IBM MQ, das als *Basisprodukt und Server* auf allen unterstützten Serverplattformen installiert ist, akzeptiert Verbindungen von IBM MQ MQI clients auf folgenden Plattformen:

- 

IBM i
- 

AIX
- 

Linux
- 

Windows

CCSID (ID des codierten Zeichensatzes) und Kommunikationsprotokoll der Clientverbindungen hängen von der jeweiligen Plattform ab.

## Auf einem IBM MQ MQI client ausführbare Anwendungen

In der Clientumgebung wird die MQI vollständig unterstützt. Damit können nahezu alle IBM MQ-Anwendungen für die Ausführung auf IBM MQ MQI clientsystemen konfiguriert werden, indem die Anwendung auf dem IBM MQ MQI client nicht mit der MQI-, sondern mit der MQIC-Bibliothek verbunden wird. Die Ausnahmen sind:

- MQGET mit Signal.
- Anwendungen, bei denen eine Synchronisationspunktkoordination mit anderen Ressourcenmanagern erforderlich ist, benötigen einen erweiterten transaktionsorientierten Client.

Wenn Vorauslesen aktiviert ist, um den Durchsatz beim nicht permanenten Messaging zu optimieren, stehen nicht alle MQGET-Optionen zur Verfügung. In der folgenden Tabelle sind alle zulässigen Optionen aufgeführt; außerdem ist angegeben, ob sie zwischen MQGET-Aufrufen geändert werden können.

*Tabelle 15. Zulässige MQGET-Optionen bei aktiviertem Vorauslesen*

| **Werte** | **Bei aktiviertem Vorauslesen zulässig und kann zwischen MQGET-Aufrufen ausgetauscht werden** | **Bei aktiviertem Vorauslesen zulässig, kann aber nicht zwischen MQGET-Aufrufen ausgetauscht werden[1]** | **MQGET-Optionen, die bei aktiviertem Vorauslesen nicht zulässig sind[2]** |
|---|---|---|---|
| MQGET MD-Werte | MsgId[3]<br>CorrelId [3] | Encoding<br>CodedCharSetId | |
| MQGET MQGMO-Optionen | MQGMO_WAIT<br>MQGMO_NO_WAIT<br>MQGMO_FAIL_IF_QUIESCING<br>MQGMO_BROWSE_FIRST[4]<br>MQGMO_BROWSE_NEXT[4]<br>MQGMO_BROWSE_MESSAGE _UNDER_CURSOR [4] | MQGMO_SYNCPOINT_IF _PERSISTENT<br>MQGMO_NO_SYNCPOINT<br>MQGMO_ACCEPT_TRUNCATED_MSG<br>MQGMO_CONVERT<br>MQGMO_LOGICAL_ORDER<br>MQGMO_COMPLETE_MSG<br>MQGMO_ALL_MSGS_AVAILABLE<br>MQGMO_ALL_SEGMENTS_ AVAILABLE<br>MQGMO_MARK_BROWSE_HANDLE<br>MQGMO_MARK_BROWSE_CO_OP<br>MQGMO_UNMARK_BROWSE_CO_OP<br>MQGMO_UNMARK_BROWSE_HANDLE<br>MQGMO_UNMARKED_BROWSE_MSG<br>MQGMO_PROPERTIES_FORCE_MQRFH2<br>MQGMO_NO_PROPERTIES<br>MQGMO_PROPERTIES_IN_HANDLE<br>MQGMO_PROPERTIES_COMPATIBILITY | MQGMO_SET_SIGNAL<br>MQGMO_SYNCPOINT<br>MQGMO_MARK_SKIP _BACKOUT<br>MQGMO_MSG_UNDER _CURSOR[4]<br>MQGMO_LOCK<br>MQGMO_UNLOCK |
| MQGMO-Werte | | MsgHandle | |

1. Wenn diese Optionen zwischen MQGET-Aufrufen ausgetauscht werden, wird der Ursachencode MQRC_OPTIONS_CHANGED zurückgegeben.
2. Wenn diese Optionen im ersten MQGET-Aufruf angegeben werden, wird das Vorauslesen inaktiviert. Werden diese Optionen in einem nachfolgenden MQGET-Aufruf angegeben, wird Ursachencode MQRC_OPTIONS_ERROR zurückgegeben.
3. Clientanwendungen müssen erkennen können, dass bei einer Änderung der Werte für 'MsgId' und 'CorrelId' zwischen MQGET-Aufrufen Nachrichten mit den früheren Werten möglicherweise bereits an den Client gesendet wurden und im Vorauslesepuffer des Clients verbleiben, bis sie verarbeitet (oder automatisch gelöscht) werden.

4. Der erste MQGET-Aufruf bestimmt, ob Nachrichten aus einer Warteschlange angezeigt oder abgerufen werden, wenn Vorauslesen aktiviert ist. Wenn die Anwendung versucht, Anzeige und Abruf zu kombinieren, wird Ursachencode MQRC_OPTIONS_CHANGED zurückgegeben.
5. MQGMO_MSG_UNDER_CURSOR ist bei aktiviertem Vorauslesen nicht möglich. Nachrichten können angezeigt oder abgerufen werden, wenn Vorauslesen aktiviert ist, aber eine Kombination der beiden Funktionen ist nicht möglich.

Eine auf einem IBM MQ MQI client aktive Anwendung kann parallel Verbindungen zu mehreren Warteschlangenmanagern herstellen oder in einem MQCONN- oder MQCONNX-Aufruf einen Warteschlangenmanagernamen mit einem Stern (*) verwenden (siehe Beispiele im Abschnitt IBM MQ MQI client-Anwendungen mit Warteschlangenmanagern verbinden).

## Was ist ein erweiterter Transaktionsclient?

Ein erweiterter transaktionsorientierter IBM MQ-Client kann unter der Steuerung eines externen Transaktionsmanagers Ressourcen aktualisieren, die von einem anderen Ressourcenmanager verwaltet werden.

Wenn Sie nicht mit den Grundlagen des Transaktionsmanagements vertraut sind, lesen Sie den Abschnitt „Transaktionsmanagement und -unterstützung” auf Seite 158.

Der transaktionsorientierte XA-Client wird jetzt mit IBM MQ bereitgestellt.

Eine Clientanwendung kann an einer Arbeitseinheit beteiligt sein, die von einem Warteschlangenmanager verwaltet wird, mit dem sie verbunden ist. Innerhalb der Arbeitseinheit kann die Clientanwendung Nachrichten in die Warteschlangen dieses Warteschlangenmanagers einreihen bzw. Nachrichten aus diesen Warteschlangen abrufen. Anschließend kann die Clientanwendung die Arbeitseinheit mit dem **MQCMIT**-Aufruf festschreiben oder mit dem **MQBACK**-Aufruf zurücksetzen. Die Clientanwendung kann allerdings nicht innerhalb einer Arbeitseinheit die Ressourcen (z. B. die Tabellen einer Db2-Datenbank) eines anderen Ressourcenmanagers aktualisieren. Bei Verwendung eines erweiterten transaktionsorientierten IBM MQ-Clients besteht diese Einschränkung nicht.

Ein erweiterter transaktionsorientierter IBM MQ-Client ist ein IBM MQ MQI client mit zusätzlichen Funktionen. Mithilfe dieser Funktionen kann eine Clientanwendung innerhalb derselben Arbeitseinheit Folgendes ausführen:

- Nachrichten in Warteschlangen einreihen, die dem Warteschlangenmanager zugeordnet sind, mit dem die Anwendung verbunden ist, bzw. Nachrichten aus diesem Warteschlangenmanager abrufen.
- Die Ressourcen eines Ressourcenmanagers, bei dem es sich nicht um einen IBM MQ-Warteschlangenmanager handelt, verwalten.

Diese Arbeitseinheit muss von einem externen Transaktionsmanager verwaltet werden, der auf demselben System wie die Clientanwendung aktiv ist. Die Arbeitseinheit darf nicht von dem Warteschlangenmanager verwaltet werden, mit dem die Clientanwendung verbunden ist. Das heißt, dass der Warteschlangenmanager nur als Ressourcenmanager, nicht als Transaktionsmanager dienen kann. Es heißt auch, dass die Clientanwendung die Arbeitseinheit nur über die Anwendungsprogrammierschnittstelle (API) des externen Transaktionsmanagers festschreiben bzw. zurücksetzen kann. Die Clientanwendung kann daher nicht die MQI-Aufrufe (**MQBEGIN**, **MQCMIT** und **MQBACK**) verwenden.

Der externe Transaktionsmanager kommuniziert mit dem Warteschlangenmanager als Ressourcenmanager über denselben MQI-Kanal, der auch von der Clientanwendung verwendet wird, die mit dem Warteschlangenmanager verbunden ist. Bei einer Wiederherstellung im Anschluss an einen Ausfall, wenn keine Anwendungen aktiv sind, kann der Transaktionsmanager jedoch über einen dedizierten MQI-Kanal alle unvollständigen Arbeitseinheiten wiederherstellen, an denen der Warteschlangenmanager zum Zeitpunkt des Ausfalls beteiligt war.

In diesem Abschnitt werden IBM MQ MQI clients ohne die erweiterte transaktionsorientierte Funktion als IBM MQ-Basisclients bezeichnet. Ein erweiterter transaktionsorientierter IBM MQ-Client ist also ein IBM MQ-Basisclient mit erweiterter transaktionsorientierter Funktion.

**Anmerkung:** IBM i Von IBM MQ MQI client on IBM i wird die erweiterte transaktionsorientierte IBM MQ-Funktion nicht unterstützt.

### Plattformunterstützung für erweiterte Transaktionsclients

Multi

Erweiterte transaktionsorientierte Clients sind für alle Multiplatforms verfügbar, die einen Basisclient unterstützen. Die Clients sind nicht für z/OS verfügbar.

Eine Clientanwendung, die einen erweiterten transaktionsorientierten Client verwendet, kann nur eine Verbindung zu einem Warteschlangenmanager der folgenden IBM MQ 9.0oder höher herstellen:

- AIX IBM MQ for AIX
- IBM i IBM MQ for IBM i
- Linux IBM MQ fürLinux
- Windows IBM MQ for Windows

z/OS Zwar gibt es keine erweiterten transaktionsorientierten Clients, die unter z/OS ausgeführt werden, jedoch können Clientanwendungen, die einen erweiterten transaktionsorientierten Client verwenden, eine Verbindung zu einem Warteschlangenmanager herstellen, der unter z/OS läuft.

Auf allen Plattformen gelten für erweiterte transaktionsorientierte Clients dieselben Hardware- und Softwarevoraussetzungen wie für den IBM MQ-Basisclient. Programmiersprachen, die vom IBM MQ-Basisclient und dem von Ihnen verwendeten Transaktionsmanager unterstützt werden, werden auch von einem erweiterten transaktionsorientierten Client unterstützt.

Informationen zu den externen Transaktionsmanagern für alle Plattformen finden Sie unter Systemvoraussetzungen für IBM MQ.

## Verbindung vom Client zum Server herstellen

Clients stellen über MQCONN- oder MQCONNX-Aufrufe eine Verbindung zum Server her; die Kommunikation erfolgt über einen Kanal.

Eine in der IBM MQ-Clientumgebung aktive Anwendung muss eine aktive Verbindung zwischen Client- und Servermaschinen aufrechterhalten.

Die Verbindung wird über einen MQCONN- oder MQCONNX-Aufruf hergestellt, den die Anwendung absetzt. Clients und Server kommunizieren über *MQI-Kanäle* miteinander oder (bei Verwendung gemeinsam genutzter Verbindungen) über Verbindungen, bei denen eine MQI-Kanalinstanz gemeinsam genutzt wird. Ist der Aufruf erfolgreich, wird die MQI-Kanalinstanz oder Verbindung aufrechterhalten, bis die Anwendung einen MQDISC-Aufruf absetzt. Dies gilt für jeden Warteschlangenmanager, zu dem eine Anwendung eine Verbindung herstellen muss.

### Client und Warteschlangenmanager auf einem System

Anwendungen können auch in der IBM MQ MQI client-Umgebung ausgeführt werden, wenn auf demselben System zusätzlich noch ein Warteschlangenmanager installiert ist.

In diesem Fall können Sie eine Verbindung zu den Warteschlangenmanagerbibliotheken oder den Clientbibliotheken herstellen; allerdings müssen auch bei einer Verbindung zu den Clientbibliotheken die Kanalverbindungen definiert werden. Dies kann bei der Entwicklung einer Anwendung hilfreich sein. Sie können das Programm auf der eigenen Maschine ohne Abhängigkeiten von anderen testen und sicher sein, dass es auch dann arbeitet, wenn es in einer unabhängigen IBM MQ MQI clientumgebung eingesetzt wird.

### Clients auf verschiedenen Plattformen

In diesem Beispiel kommuniziert die Servermaschine mit drei IBM MQ MQI clients auf verschiedenen Plattformen.

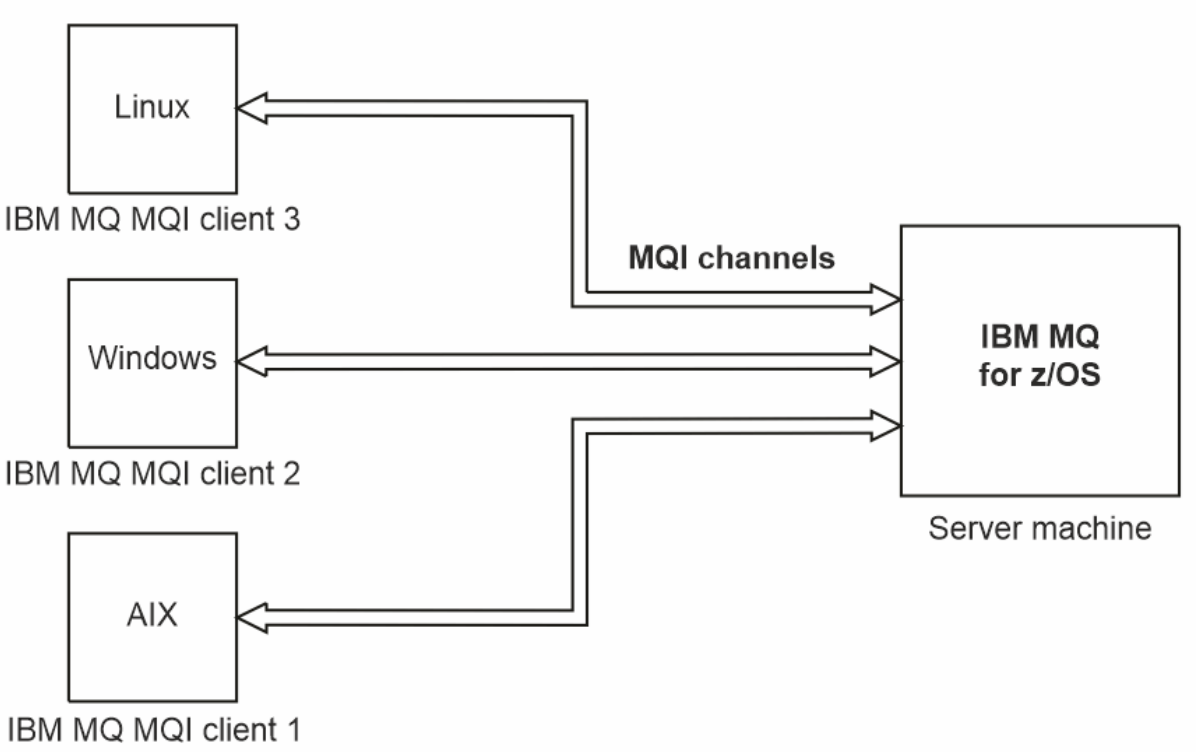



*Abbildung 52. Verbindung eines IBM MQ-Servers mit Clients auf unterschiedlichen Plattformen*

Es sind weitere komplexere Umgebungen möglich. So kann beispielsweise ein IBM MQ-Client mit mehreren Warteschlangenmanagern verbunden werden oder mit einer beliebigen Anzahl von Warteschlangenmanagern, die als Teil einer Gruppe mit gemeinsamer Warteschlange miteinander verbunden sind.

### Verschiedene Versionen von Client- und Server-Software verwenden

Bei Verwendung älterer Versionen von IBM MQ-Produkten müssen Sie sicherstellen, dass der Server die Konvertierung der CCSID (ID des codierten Zeichensatzes) des Clients unterstützt.

Ein IBM MQ -Client kann eine Verbindung zu allen unterstützten Versionen des Warteschlangenmanagers herstellen. Wenn Sie eine Verbindung mit einem Warteschlangenmanager einer früheren Version herstellen, können Sie Funktionen und Strukturen aus einer späteren Version des Produkts in Ihrer IBM MQ-Anwendung auf dem Client nicht verwenden.

Ein IBM MQ -Warteschlangenmanager kann mit Clients unterschiedlicher Versionen mit sich selbst kommunizieren, indem er bis zur höchsten gegenseitig unterstützten Protokollebene aushandelt. Dies bedeutet, dass ältere Clients mit höheren Warteschlangenmanagerversionen verwendet werden können. Es wird empfohlen, dass sowohl der Client als auch der Server IBM MQ -Versionen haben, die derzeit unterstützt werden, um die Problemdiagnose zu vereinfachen und die Unterstützung durch IBMzu aktivieren.

Weitere Informationen finden Sie in den unterstützten Programmiersprachen unter Anwendungen entwickeln.

## Transaktionsmanagement und -unterstützung

Dieser Abschnitt gibt eine Einführung in das Transaktionsmanagement und es wird erläutert, wie Transaktionen in IBM MQ unterstützt werden.

Ein *Ressourcenmanager* ist ein Computersubsystem, das Eigner der Ressourcen ist, die von ihm verwaltet werden; Anwendungen können auf diese Ressourcen zugreifen und sie aktualisieren. Hier einige Beispiele für Ressourcenmanager:

- Ein IBM MQ-Warteschlangenmanager; bei den Ressourcen handelt es sich um Warteschlangen
- Eine Db2-Datenbank; bei den Ressourcen handelt es sich um die Datenbanktabellen

Wenn eine Anwendung die Ressourcen auf einem oder auch mehreren Ressourcenmanagern aktualisiert, ist möglicherweise eine Geschäftsanforderung implementiert, mit der sichergestellt wird, dass bestimmte Aktualisierungen entweder zusammen erfolgreich oder gar nicht ausgeführt werden. Dies ist sinnvoll, da die Geschäftsdaten einen inkonsistenten Status hätten, wenn einige Aktualisierungen erfolgreich wären, andere hingegen nicht.

Bei dieser Art von Aktualisierung von Ressourcen, die auf diese Weise verwaltet werden, spricht man von einer *Arbeitseinheit* bzw. *Transaktion*. Ein Anwendungsprogramm kann eine Reihe von Aktualisierungen zu einer solchen Arbeitseinheit zusammenfassen.

Während einer solchen Arbeitseinheit sendet die Anwendung an die Ressourcenmanager Anforderungen zur Aktualisierung der Ressourcen. Die Arbeitseinheit endet, wenn von der Anwendung die Anforderung abgesetzt wird, alle Aktualisierungen festzuschreiben (Commit). Bis zu ihrer Festschreibung bleiben die Aktualisierungen für andere Anwendungen, die auf dieselben Ressourcen zugreifen, unsichtbar. Ebenso kann eine Anwendung die Arbeitseinheit auch abbrechen und eine Anforderung absetzen, alle bis dahin angeforderten Aktualisierungen zurückzusetzen. In diesem Fall werden die Aktualisierung nie sichtbar für andere Anwendungen. Diese Aktualisierungen sind in der Regel logisch miteinander verknüpft und müssen alle erfolgreich ausgeführt werden, damit die Datenintegrität gewährleistet ist. Die Datenintegrität geht verloren, wenn einige Aktualisierungen erfolgreich verlaufen, andere nicht.

Wenn eine Arbeitseinheit erfolgreich abgeschlossen ist, spricht man von einer *Festschreibung*. Nach der Festschreibung sind alle innerhalb einer Arbeitseinheit vorgenommenen Aktualisierungen permanent und können nicht mehr zurückgesetzt werden. Schlägt die Arbeitseinheit fehl, werden alle Aktualisierungen wieder *zurückgesetzt*. Dieser Prozess, bei dem Arbeitseinheiten unter Wahrung der Datenintegrität festgeschrieben oder zurückgesetzt werden, wird als *Synchronisationspunktkoordination* bezeichnet.

Der Zeitpunkt, zu dem alle Aktualisierungen innerhalb einer Arbeitseinheit festgeschrieben oder zurückgesetzt werden, wird als *Synchronisationspunkt* bezeichnet. Bei einer Aktualisierung innerhalb einer Arbeitseinheit spricht man davon, dass sie *synchronisationspunktgesteuert* ausgeführt wird. Fordert eine Anwendung eine *nicht synchronisationspunktgesteuerte* Aktualisierung an, schreibt der Ressourcenmanager die Aktualisierung sofort fest, auch wenn eine Arbeitseinheit noch nicht abgeschlossen wurde und die Aktualisierung nicht mehr zurückgesetzt werden kann.

Das Computersubsystem, das Arbeitseinheiten verwaltet, wird als *Transaktionsmanager* oder *Punktkoordinator* bezeichnet.

Eine *lokale* Arbeitseinheit ist eine Arbeitseinheit, in der nur die Ressourcen des IBM MQ-Warteschlangenmanagers aktualisiert werden. Die Synchronisationspunktkoordination wird vom Warteschlangenmanager selbst mit einer einphasigen Festschreibung vorgenommen.

Bei einer *globalen* Arbeitseinheit werden auch die Ressourcen anderer Ressourcenmanager, beispielsweise einer XA-konformen Datenbank, aktualisiert. Hier wird eine zweiphasige Festschreibung durchgeführt; die Arbeitseinheit kann vom Warteschlangenmanager selbst oder von einem anderen XA-konformen Transaktionsmanager wie IBM TXSeries oder BEA Tuxedo koordiniert werden.

Der Transaktionsmanager muss dafür sorgen, dass entweder alle Aktualisierungen, die an Ressourcen vorgenommen werden, erfolgreich innerhalb einer Arbeitseinheit abgeschlossen werden, oder keine. Anwendungen senden Anforderungen zum Festschreiben oder Zurücksetzen einer Arbeitseinheit an den Transaktionsmanager. Beispiele für Transaktionsmanager sind CICS und WebSphere Application Server, auch wenn beide Produkte noch über andere Funktionen verfügen.

Einige Ressourcenmanager stellen eine eigene Funktion für das Transaktionsmanagement bereit. So kann beispielsweise ein IBM MQ-Warteschlangenmanager Arbeitseinheiten verwalten, bei denen die eigenen Ressourcen und Db2-Tabellen aktualisiert werden. Der Warteschlangenmanager benötigt hierfür keinen separaten Transaktionsmanager; wenn dies vom Benutzer gefordert wird, kann jedoch zusätzlich ein Transaktionsmanager eingesetzt werden. Ein solcher separater Transaktionsmanager wird als *externer Transaktionsmanager* bezeichnet.

Damit eine Arbeitseinheit von einem externen Transaktionsmanager verwaltet werden kann, ist zwischen dem Transaktionsmanager und allen an der Arbeitseinheit beteiligten Ressourcenmanagern eine

Standardschnittstelle erforderlich. Über diese Schnittstellen können der Transaktionsmanager und die Ressourcenmanager miteinander kommunizieren. Eine dieser Schnittstellen ist die *XA-Schnittstelle*, eine Standardschnittstelle, die von einer Reihen von Transaktions- und Ressourcenmanagern unterstützt wird. Die XA-Schnittstelle ist von The Open Group in *Distributed Transaction Processing: The XA Specification* veröffentlicht.

Wenn mehrere Ressourcenmanager an einer Arbeitseinheit beteiligt sind, muss der Transaktionsmanager ein Protokoll für die *zweiphasige Festschreibung* verwenden; damit wird sichergestellt, dass entweder alle Aktualisierungen innerhalb einer Arbeitseinheit erfolgreich ausgeführt werden oder keine, und dies auch im Fall eines Systemausfalls. Wenn der Transaktionsmanager von einer Anwendung eine Anforderung zum Festschreiben einer Arbeitseinheit erhält, wird Folgendes durchgeführt:

**Phase 1 (Vorbereitung für die Festschreibung)**
Der Transaktionsmanager fragt die einzelnen Ressourcenmanager ab, die an dieser Arbeitseinheit beteiligt sind, um sicherzustellen, dass alle Informationen zu den beabsichtigten Aktualisierungen an den Ressourcen einen wiederherstellbaren Status haben. Dazu schreibt ein Ressourcenmanager die Informationen in der Regel in ein Protokoll und stellt sicher, dass diese Informationen auf der Festplatte gespeichert werden. Phase 1 ist abgeschlossen, wenn der Transaktionsmanager von jedem Ressourcenmanager benachrichtigt wurde, dass die Informationen zu den beabsichtigten Aktualisierungen an den Ressourcen einen wiederherstellbaren Status haben.

**Phase 2 (Festschreibung)**
Nach Abschluss der Phase 1 bereitet der Transaktionsmanager den unwiderruflichen Schritt vor, mit dem die Arbeitseinheit festgeschrieben wird. Dazu fordert er die einzelnen Ressourcenmanager, die an der Arbeitseinheit beteiligt sind, auf, die Aktualisierungen an ihren Ressourcen festzuschreiben. Wenn ein Ressourcenmanager diese Anforderung empfängt, muss er die Aktualisierungen festschreiben. Zu diesem Zeitpunkt ist es für ihn nicht mehr möglich, die Aktualisierungen zurückzusetzen. Phase 2 ist abgeschlossen, wenn der Transaktionsmanager von jedem Ressourcenmanager benachrichtigt wird, dass die Aktualisierungen an den Ressourcen festgeschrieben wurden.

Die XA-Schnittstelle verwendet ein solches Protokoll für die zweiphasige Festschreibung.

Weitere Informationen finden Sie im Abschnitt Szenarios zur Transaktionsunterstützung.

IBM MQ stellt auch Unterstützung für Microsoft Transaction Server (COM +) bereit. Im Abschnitt Microsoft Transaction Server (COM+) verwenden wird beschrieben, wie IBM MQ für die Nutzung der COM+-Unterstützung konfiguriert wird.

# Funktionen des Warteschlangenmanagers erweitern

Sie können die Funktionen eines Warteschlangenmanagers durch Benutzerexits, API-Exits oder installierbare Services erweitern.

## Benutzerexits

Benutzerexits ermöglichen es Ihnen, eigenen Code in eine Warteschlangenmanagerfunktion einzufügen. Zu den unterstützten Benutzerexits gehören folgende:

**Kanalexits**
Über diese Exits kann die Betriebsweise von Kanälen geändert werden. Kanalexits werden im Abschnitt Kanalexitprogramme für Messaging-Kanäle beschrieben.

**Datenkonvertierungsexits**
Diese Exits erstellen Quellencodefragmente, die in Anwendungsprogramme eingefügt werden, um Daten in ein anderes Format zu konvertieren. Datenkonvertierungsexits werden im Abschnitt Datenkonvertierungsexits schreiben beschrieben.

**Exit für Clusterauslastung**
Die Funktion dieses Exits wird vom Exit-Provider definiert. Informationen zur Aufrufdefinition finden Sie im Abschnitt MQ_CLUSTER_WORKLOAD_EXIT - Beschreibung des Aufrufs.

### API-Exits

API-Exits ermöglichen das Schreiben von Code zur Änderung des Verhaltens von IBM MQ-API-Aufrufen wie MQPUT und MQGET, und fügen diesen Code dann unmittelbar vor oder nach diesen Aufrufen ein. Das Einfügen erfolgt automatisch; der Exit-Code wird dann vom Warteschlangenmanager an den registrierten Punkten ausgeführt. Weitere Informationen zu API-Exits finden Sie im Abschnitt API-Exits schreiben und verwenden.

### Installierbare Services

Installierbare Services haben formalisierte Schnittstellen (eine API) mit mehreren Eingangspunkten.

Ein implementierter installierbarer Service wird als *Servicekomponente* bezeichnet. Sie können entweder die zusammen mit IBM MQ bereitgestellten Komponenten verwenden oder eigene Komponenten für die für Ihre Umgebung erforderlichen Funktionen erstellen.

Derzeit werden die folgenden installierbaren Services bereitgestellt:

**Berechtigungsservice**
Mithilfe des Berechtigungsservice können Sie eine eigene Sicherheitsfunktion erstellen.

Die Standardservicekomponente, mit der der Service implementiert wird, ist der Objektberechtigungsmanager (Object Authority Manager; OAM). Der Objektberechtigungsmanager ist standardmäßig aktiviert; Sie müssen nichts tun, um ihn zu konfigurieren. Mithilfe der Berechtigungsserviceschnittstelle können Sie andere Komponenten erstellen, um den Objektberechtigungsmanager zu ersetzen oder aber zu erweitern. Weitere Informationen zu OAM finden Sie unter Sicherheit auf AIX, Linux, and Windows-Systemen einrichten.

**Namensservice**
Mit dem Namensservice können Anwendungen Warteschlangen gemeinsam nutzen, indem ferne Warteschlangen wie lokale Warteschlangen angegeben werden.

Sie können eine eigene Namensservicekomponente erstellen. Dies ist beispielsweise sinnvoll, wenn der Namensservice zusammen mit IBM MQ verwendet werden soll. Um den Namensservice zu verwenden, muss entweder eine benutzerdefinierte Komponente oder eine Komponente von einem anderen Softwareanbieter vorhanden sein. Der Namensservice ist standardmäßig inaktiv.

**Zugehörige Konzepte**
Benutzerexits, API-Exits und installierbare IBM MQ-Services

## IBM MQ Java-Sprachenschnittstellen

IBM MQ stellt drei Anwendungsprogrammierschnittstellen (APIs) für die Verwendung in Java -Anwendungen bereit: IBM MQ classes for Jakarta Messaging, IBM MQ classes for JMSund IBM MQ classes for Java.

IBM unterstützt und ist ein aktiver Teilnehmer an offenen Standards.

- Ab IBM MQ 8.0implementiert das Produkt den JMS 2.0 -Standard, der eine neue vereinfachte API zusammen mit Features wie gemeinsam genutzte Subskriptionen eingeführt hat.
- V 9.3.0 V 9.3.0 Ab IBM MQ 9.3.0wird auch Jakarta Messaging 3.0 unterstützt.
- Außerdem unterstützt WebSphere Liberty JMS 2.0 und Jakarta Messaging 3.0 mit IBM MQ.

In IBM MQ gibt es drei APIs für die Verwendung in Java -Anwendungen:

JM 3.0 V 9.3.0 V 9.3.0 **IBM MQ classes for Jakarta Messaging**
IBM MQ classes for Jakarta Messaging ist ein Jakarta Messaging-Provider, der die Jakarta Messaging-Schnittstellen für IBM MQ als Messaging-System implementiert. Jakarta Connectors Architecture bietet eine Standardmethode zum Verbinden von Anwendungen, die in einer Jakarta EE -Umgebung ausgeführt werden, mit einem unternehmensweiten Informationssystem (EIS) wie IBM MQ oder Db2.

JMS 2.0 **IBM MQ classes for JMS**
IBM MQ classes for JMS ist ein JMS-Provider, der die JMS-Schnittstellen für IBM MQ als Messaging-System implementiert. Die Java Platform, Enterprise Edition Connector Architecture (JCA) stellt eine Standardmethode für die Verbindung von Anwendungen, die in einer Java EE-Umgebung aktiv sind, mit einem Enterprise Information System (EIS) wie IBM MQ oder Db2 bereit.

**IBM MQ classes for Java**
IBM MQ classes for Java ermöglichen Ihnen die Verwendung von IBM MQ in einer Java-Umgebung. IBM MQ classes for Java ermöglichen einer Java-Anwendung eine Verbindung mit IBM MQ als IBM MQ-Client oder eine direkte Verbindung mit einem IBM MQ-Warteschlangenmanager.

**Anmerkung:**

- V 9.3.0 V 9.3.0 JMS 2.0 wurde durch Jakarta Messagingersetzt. IBM MQ classes for JMS unterstützt weiterhin den JMS 2.0 -Standard, aber zukünftige Erweiterungen für das Java -Messaging treten nur in Jakarta Messagingauf, daher in IBM MQ classes for Jakarta Messaging. IBM MQ classes for JMS wird nur für die Verwaltung und Erweiterung vorhandener JMS 2.0 -Anwendungen empfohlen. IBM MQ classes for Jakarta Messaging sollte die bevorzugte Technologie für die Neuentwicklung sein.
- Stabilized IBM MQ classes for Java werden funktional auf der mit IBM MQ 8.0 bereitgestellten Stufe eingefroren. Bestehende Anwendungen, die IBM MQ classes for Java verwenden, werden weiterhin vollständig unterstützt, diese API wird jedoch eingefroren, d. h., es werden keine Änderungen mehr daran vorgenommen (auch nicht auf Kundenanfrage). Vollständig unterstützt bedeutet, dass Fehler behoben und daraus erforderliche Änderungen an den IBM MQ-Systemvoraussetzungen vorgenommen werden.

JM 3.0 V 9.3.0 V 9.3.0 Ab IBM MQ 9.3werden die IBM MQ classes for Java, IBM MQ classes for JMSund IBM MQ classes for Jakarta Messaging mit Java 8erstellt. Java -Laufzeitumgebungen mit diesen Ebenen oder höher müssen verwendet werden, um Anwendungen unter Verwendung dieser Schnittstellen auszuführen.

**Zugehörige Konzepte**
Zugriff auf IBM MQ über Java -Auswahl der API
V 9.3.0 V 9.3.0 Warum sollte ich IBM MQ -Klassen für Jakarta Messaging verwenden?
Gründe für die Verwendung von IBM MQ classes for JMS
Gründe für die Verwendung von IBM MQ classes for Java

## IBM MQ classes for JMS/Jakarta Messaging

IBM MQ classes for JMS und IBM MQ classes for Jakarta Messaging sind die Messaging-Provider, die mit IBM MQbereitgestellt werden. Jeder dieser Provider stellt außerdem zwei Gruppen von Erweiterungen für die Messaging API bereit. Diese Messaging-Provider können sowohl von Java Platform, Standard Edition -Anwendungen ( Java SE) als auch von Java Platform, Enterprise Edition -Anwendungen ( Java EE) verwendet werden.

JM 3.0 V 9.3.0 V 9.3.0 IBM MQ 9.3.0 bietet jetzt Unterstützung für Jakarta Messaging 3.0. JMS 2.0 wird weiterhin vollständig unterstützt.

Die Spezifikationen JMS und Jakarta Messaging definieren eine Gruppe von Schnittstellen, die Anwendungen für Messaging-Operationen verwenden können. Ab IBM MQ 8.0 unterstützt das Produkt die JMS 2.0-Version des JMS-Standards. Diese Implementierung bietet alle Funktionen der klassischen API, erfordert jedoch weniger Schnittstellen und ist einfacher zu verwenden. Weitere Informationen hierzu finden Sie im Abschnitt „Modell JMS und Jakarta Messaging” auf Seite 166 und in der Spezifikation zu JMS 2.0 auf Java.net. JM 3.0 Ab IBM MQ 9.3.0wird auch Jakarta Messaging unterstützt.

Das Paket jakarta.jms (Jakarta Messaging 3.0) oder javax.jms (JMS 2.0) gibt die Details der Messaging-Schnittstellen an, und ein Messaging-Provider implementiert diese Schnittstellen für ein bestimmtes Messaging-Produkt. For example:

- IBM MQ classes for JMS ist ein JMS-Provider, der die JMS-Schnittstellen für IBM MQ implementiert und darüber hinaus die beiden folgenden Erweiterungssätze für die JMS-API bereitstellt:
  - IBM MQ JMS-Erweiterungen
  - IBM JMS-Erweiterungen
- Eine Verbindungsfactory, eine Warteschlange oder ein Themenobjekt, die bzw. das mit javax.dimsoder jakarta.jmserstellt wurde, Schnittstellen oder eine Gruppe von JMS -Erweiterungen können mit jeder dieser APIs adressiert werden, d. h., sie können in jede der Schnittstellen umgesetzt werden. Um die Portierbarkeit Ihrer Anwendungen zu erleichtern, sollten Sie stets die allgemeinste API verwenden, die für Ihre Anforderungen gerade noch geeignet ist.

Da JMS und Jakarta Messaging viel gemeinsam genutzt werden, können weitere Verweise auf JMS in diesem Abschnitt als Verweise auf beides betrachtet werden. Alle Unterschiede werden nach Bedarf hervorgehoben.

### IBM MQ JMS-Erweiterungen

IBM MQ classes for JMS stellt Erweiterungen für die JMS-API bereit. IBM MQ classes for JMS enthält Erweiterungen, die in MQConnectionFactory-, MQQueue-und MQTopic-Objekten implementiert sind. Diese Objekte haben IBM MQ-spezifische Eigenschaften und Methoden. Bei den Objekten kann es sich um verwaltete Objekte handeln oder eine Anwendung kann die Objekte dynamisch während der Laufzeit erstellen. Diese Erweiterungen werden als IBM MQ JMS -Erweiterungen bezeichnet. In dieser Dokumentation werden Objekte, die dynamisch von einer Anwendung zur Laufzeit erstellt werden, nicht als verwaltete Objekte betrachtet.

### IBM JMS-Erweiterungen

Zusätzlich zu den IBM MQ JMS -Erweiterungen bietet IBM MQ classes for JMS eine allgemeinere Gruppe von Erweiterungen für die JMS -API oder Java als Programmiersprache. Diese Erweiterungen werden als IBM JMS -Erweiterungen bezeichnet und haben die folgenden allgemeinen Ziele:

- Um eine größere Konsistenz zwischen IBM JMS -Providern bereitzustellen.
- Um das Schreiben einer Brückenanwendung zwischen zwei IBM -Messaging-Systemen zu vereinfachen.
- Um das Portieren einer Anwendung von einem IBM JMS -Provider auf einen anderen zu vereinfachen

Der Hauptschwerpunkt dieser Erweiterungen liegt auf der dynamischen Erstellung und Konfiguration von Verbindungsfactorys und Zielen zur Laufzeit, die Erweiterungen bieten jedoch auch Funktionen, die nicht direkt mit Messaging verbunden sind, z. B. Funktionen für die Problembestimmung.

**Zugehörige Tasks**

IBM MQ -Klassen für JMS/Jakarta Messaging verwenden

JMS-und Jakarta Messaging-Ressourcen konfigurieren

## JM 3.0 V 9.3.0 V 9.3.0 IBM MQ classes for Jakarta Messaging: Übersicht

IBM MQ 9.3.0 führt Unterstützung für Jakarta Messagingein. Für Jakarta Messaging 3.0die Steuerung der JMS -Spezifikation, die von Oracle in den Java -Community-Prozess verschoben wurde Oracle behält jedoch die Steuerung des Namens "javax", der in anderen Java -Technologien verwendet wird. Obwohl Jakarta Messaging 3.0 funktional äquivalent zu JMS 2.0ist, gibt es einige Unterschiede bei der Benennung. Der offizielle Name für Version 3.0 ist Jakarta Messaging und nicht Java Message Serviceund die Paket-und Konstantennamen haben das Präfix `jakarta` und nicht `javax`.

### Hintergrund

Die Java -Plattform wird seit vielen Jahren in zwei Formaten bereitgestellt: Standard Edition und Enterprise Edition.

Java Platform, Standard Edition (manchmal abgekürzt Java SE) ist die Kernsprache und die Klassenbibliotheken, die in einem eigenständigen Kontext ausgeführt werden können. Die meisten Java -Pakete in Java SE haben Namen, die mit "java." beginnen.

Java Platform, Enterprise Edition (Java EE) erweitert dies und fügt Funktionalität wie Messaging, verschiedene Beans, Transaktionalität usw. hinzu. Einige dieser Technologien können auch in einem Java SE -Kontext verwendet werden. Die meisten Java -Pakete in Java EE haben in der Vergangenheit Namen, die mit "javax." beginnen. Es gibt jedoch eine Querverbindung, sodass einige Java SE -Pakete "javax" enthalten. als Präfix für ihren Namen.

Java Message Service (JMS) ist Teil von Java Platform, Enterprise Edition. Java EE 7 enthält JMS 2.0.

Bis Java EE 7standen die Technologien unter der Verantwortung von Oracle.

Die Java EE -Technologien wurden kürzlich von der Verwaltung von Oracle zu einem Community-Prozess verschoben, der von der Eclipse Foundation überwacht wird.

Als "javax." Name konnte nicht in das neue Projekt verschoben werden, neue Benennung wurde übernommen-alle Pakete und Eigenschaftsnamen haben jetzt das Präfix "jakarta". Java Platform, Enterprise Edition wird in Zukunft als "Jakarta EE" bezeichnet. Die Versionsnummerierung wurde fortgesetzt: Version 8 war eine Zwischenversion, die weitgehend ignoriert werden kann, und Jakarta EE 9 ist der Punkt, an dem "jakarta". Präfix wird wirksam.

Die Jakarta EE -Haupttechnologie, die im IBM MQ -Kontext angewendet wird, ist Jakarta Messaging 3.0 -der Nachfolger von Java Message Service (JMS) 2.0. Jakarta EE 9 enthält also Jakarta Messaging 3.0.

IBM MQ unterstützt weiterhin Java EE 7 und JMS 2.0und bietet Unterstützung für Jakarta EE 9 und Jakarta Messaging 3.0.

## Was wird bereitgestellt: Java SE

Für Java Platform, Standard Editionstellt IBM MQ 9.3.0 zusätzlich zu IBM MQ classes for JMS (die JMS 2.0 -Operationen mit IBM MQunterstützen) IBM MQ classes for Jakarta Messagingbereit. Diese Klassen stellen einen Jakarta Messaging 3.0 -Provider bereit, der in IBM MQintegriert ist und die Verwendung von IBM MQ -Warteschlangenmanagern zur Vereinfachung von Jakarta Messaging -Operationen ermöglicht.

Diese werden als Standard-JAR-Datei `com.ibm.mq.jakarta.client.jar`im Unterverzeichnis `java/lib` der IBM MQ -Installation bereitgestellt.

Für die Verwendung in OSGi-Containern wie Apache Felix oder Eclipse Equinox stellt IBM MQ auch ein Paar OSGi-Bundles bereit:

- `com.ibm.mq.osgi.jms30.clientprereqs_`*`V.R.M.F`*`.jar`
- `com.ibm.mq.osgi.jms30.client_`*`V.R.M.F`*`.jar`

Dabei ist *V.R.M.F* stellt die Version von IBM MQdar, z. B. 9.3.0.0. Diese Bundles befinden sich im Unterverzeichnis `java/lib/OSGi` der IBM MQ -Installation.

## Was wird bereitgestellt: Jakarta EE 9

Zur Unterstützung des IBM MQ-basierten Messaging in einem Jakarta EE 9 -kompatiblen Anwendungsserver stellt IBM MQ einen Jakarta EE 9-kompatiblen Ressourcenadapter bereit: `wmq.jakarta.jmsra.rar`. Diese Datei befindet sich im Unterverzeichnis `java/lib/jca` der IBM MQ -Installation.

IBM MQ stellt weiterhin einen Java EE 7 -kompatiblen Ressourcenadapter ( `wmq.jmsra.rar`) im Unterverzeichnis `java/lib/jca` der IBM MQ -Installation bereit.

## Bereitstellung dieser Artefakte

Diese JAR-Dateien und die RAR-Datei für den Ressourcenadapter werden mit den bereits vorhandenen Artefakten auf den üblichen IBM MQ -Installationsmedien gepackt-sowohl mit den plattformspezifischen Installationsmedien, wie z. B. "`.rpm`" -Dateien, als auch mit den weiterverteilbaren Medien, wie z. B. den selbstextrahierenden weiterverteilbaren Client-JAR-Dateien.

## Änderungen zwischen JMS 2.0 und Jakarta Messaging 3.0

Jakarta EE 9 und Jakarta Messaging 3.0 führen keine neue Funktionalität ein. Alles, was sich ändert, sind Namen. Wenn Sie beispielsweise "javax.jms.Connection" in JMS 2.0verwenden, verwenden Sie "jakarta.jms.Connection" in Jakarta Messaging 3.0.

Da die Eclipse Foundation die Jakarta EE -Plattform weiterleitet, wird sie auf dieser Basis aufbauen und diese Namenskonvention wird für neue Funktionen verwendet, die in Zukunft eingeführt werden.

## Änderungen zwischen IBM MQ classes for JMS und IBM MQ classes for Jakarta Messaging

### Zusammenfassung

IBM MQ classes for JMS, die Unterstützung für JMS 2.0bereitstellen, bleiben verfügbar und werden hauptsächlich für die Verwaltung und Erweiterung vorhandener Anwendungen empfohlen. Sie werden vollständig unterstützt.

IBM MQ classes for Jakarta Messaging, die Unterstützung für Jakarta Messaging 3.0bereitstellen, werden für die Neuentwicklung empfohlen.

In IBM MQ 9.3.0sind diese beiden Angebote funktional äquivalent. Nur die Benennung unterscheidet sich. Es ist jedoch wahrscheinlicher, dass neue Messaging-Funktionalität in IBM MQ classes for Jakarta Messaging entsteht als in IBM MQ classes for JMS.

Die beiden Angebote sind interoperabel. Nachrichten, die von IBM MQ classes for JMS erzeugt werden, können von IBM MQ classes for Jakarta Messagingverarbeitet werden und umgekehrt. Die beiden Angebote dürfen jedoch nicht in einer einzigen Anwendung koexistieren.

### Benennungsänderungen

*Tabelle 16. Änderungen an Paketnamen*

| IBM MQ classes for JMS Paketname | IBM MQ classes for Jakarta Messaging Paketname |
|---|---|
| com.ibm.mq.jms[.*] | com.ibm.mq.jakarta.jms[.*] |
| com.ibm.jms | com.ibm.jakarta.jms |
| com.ibm.msg.client.jms.* | com.ibm.msg.client.jakarta.jms.* |
| com.ibm.msg.client.wmq.* | com.ibm.msg.client.jakarta.wmq.* |

Die Pakete, die sich auf allgemeine Services (Trace, Protokollierung, Unterstützung in der Landessprache usw.) und die JMQI-Implementierungen (lokal und fern) beziehen, sind für IBM MQ classes for JMS und IBM MQ classes for Jakarta Messaginggemeinsam, sodass in diesen Bereichen keine Änderungen erforderlich sind.

Beachten Sie, dass auch Eigenschaftsnamen geändert wurden. Die Eigenschaft zum Aktivieren von IBM MQ -Erweiterungen in IBM MQ classes for Jakarta Messaging ist beispielsweise **`com.ibm.mq.jakarta.jms.SupportMQExtensions`**.

Eigenschaftsnamen, die unabhängig von IBM MQ classes for JMS oder IBM MQ classes for Jakarta Messagingsind, wie z. B. die verschiedenen **`com.ibm.msg.client.commonservices.trace.*`** -Eigenschaften, gelten gleichermaßen für beide Angebote.

### Verwaltungsdienstprogramme

Die Dienstprogramme **`crtmqenv`** und **`setmqenv`** akzeptieren jetzt eine Option, mit der angegeben werden kann, ob der Klassenpfad für IBM MQ classes for JMS (`-j 2.0`) oder IBM MQ classes for Jakarta Messaging (`-j 3.0`) konfiguriert werden soll, und es gibt IBM MQ classes for Jakarta Messaging Varianten der **`runjms`** -Dienstprogramme namens **`runjms30`** und ähnliche Namen.

Wenn das Dienstprogramm **dspmqver** aufgefordert wird, Berichte zu Java -Komponenten zu erstellen, wird IBM MQ classes for Jakarta Messaging in die Ausgabe eingeschlossen.

Zum Konfigurieren von IBM MQ classes for Jakarta Messaging -Objekten, die über JNDI abgerufen werden sollen, entspricht das neue Dienstprogramm **JMS30Admin** dem Dienstprogramm **JMSAdmin** für IBM MQ classes for JMS.

Beachten Sie, dass die zugrunde liegenden Objekte aus verschiedenen Paketen stammen. JNDI definitions created by **JMSAdmin** cannot be used by IBM MQ classes for Jakarta Messaging, nor can those created by **JMS30Admin** be used by IBM MQ classes for JMS.

**Anmerkung:** Es gibt keine Unterstützung für IBM MQ classes for Jakarta Messaging -Objekte, die von IBM MQ Explorerbereitgestellt werden; ihre JNDI-Integration gilt nur für IBM MQ classes for JMS .

**Zugehörige Konzepte**

Warum sollte ich IBM MQ -Klassen für Jakarta Messaging verwenden?

## Modell JMS und Jakarta Messaging

Das Modell JMS und Jakarta Messaging definiert eine Gruppe von Schnittstellen, die Java -Anwendungen für Messaging-Operationen verwenden können. IBM MQ classes for JMS und IBM MQ classes for Jakarta Messaging sind Messaging-Provider, die definieren, wie Java -Messaging-Objekte zu IBM MQ -Konzepten gehören. Die Spezifikationen JMS und Jakarta Messaging erwarten, dass bestimmte Messaging-Objekte verwaltete Objekte sind.

Ab IBM MQ 8.0unterstützt das Produkt die JMS 2.0 -Version des JMS-Standards, die eine vereinfachte API eingeführt hat und gleichzeitig die klassische API aus JMS 1.1beibehält.

JM 3.0 V 9.3.0 V 9.3.0 IBM MQ 9.3.0 bietet jetzt Unterstützung für Jakarta Messaging 3.0. JMS 2.0 wird weiterhin vollständig unterstützt. Da JMS und Jakarta Messaging viel gemeinsam genutzt werden, können weitere Verweise auf JMS in diesem Abschnitt als Verweise auf beides betrachtet werden. Alle Unterschiede werden nach Bedarf hervorgehoben.

### Vereinfachte API

JMS 2.0 hat die vereinfachte API eingeführt und gleichzeitig die domänenspezifischen und domänenunabhängigen Schnittstellen von JMS 1.1beibehalten. Die vereinfachte API reduziert die Anzahl der zum Senden und Empfangen von Nachrichten erforderlichen Objekte und besteht aus den folgenden Schnittstellen:

**ConnectionFactory**
Verbindungsfactory - Ein verwaltetes Objekt, das von einem JMS-Client zum Erstellen einer Verbindung verwendet wird. Diese Schnittstelle wird auch in der klassischen API verwendet.

**JMSContext**
JMS-Kontext - Dieses Objekt vereint die Objekte "Connection" und "Session" der klassischen API. JMSContext-Objekte können aus anderen JMSContext-Objekten erstellt werden, wobei die zugrunde liegende Verbindung dupliziert wird.

**JMSProduzent**
JMS-Produzent - Ein JMSProducer wird aus einem JMSContext erstellt und zum Senden von Nachrichten an eine Warteschlange oder ein Thema verwendet. Das JMSProducer-Objekt veranlasst die Erstellung der zum Senden einer Nachricht erforderlichen Objekte.

**JMSKonsument**
JMS-Konsument - Ein JMSConsumer wird aus einem JMSContext erstellt und zum Empfangen der Nachrichten aus einer Warteschlange oder einem Thema verwendet.

Die vereinfachte API hat verschiedene Auswirkungen:

- Das Objekt JMSContext startet die zugrunde liegende Verbindung immer sofort automatisch.
- JMSProducer und JMSConsumer können nun mit der Methode getBody der Nachricht direkt mit dem Nachrichtenhauptteil interagieren, ohne das vollständige Nachrichtenobjekt abrufen zu müssen.

- Nachrichteneigenschaften können nun vor dem Senden des Nachrichten-Bodys (Nachrichtenhauptteils bzw. Inhalt der Nachricht) mittels Methodenverkettung im JMSProducer-Objekt festgelegt werden. Der JMSProducer veranlasst die Erstellung aller Objekte, die zum Senden einer Nachricht erforderlich sind. Mit JMS 2.0 können Eigenschaften wie folgt festgelegt und eine Nachricht gesendet werden:

```
context.createProducer().
setProperty("foo", "bar").
setTimeToLive(10000).
setDeliveryMode(NON_PERSISTENT).
setDisableMessageTimestamp(true).
send(dataQueue, body);
```

JMS 2.0 hat auch gemeinsam genutzte Subskriptionen eingeführt, bei denen Nachrichten von mehreren Konsumenten gemeinsam genutzt werden können. Alle Subskriptionen aus JMS 1.1 werden wie nicht gemeinsam genutzte Subskriptionen behandelt.

## Klassische API

Die folgende Aufstellung geht kurz auf die wichtigsten JMS-Schnittstellen der klassischen API ein:

**Destination**
: Ziel - Ein Destination-Objekt ist der Ort, an den eine Anwendung Nachrichten sendet, und/oder eine Quelle, aus der eine Anwendung Nachrichten empfängt.

**ConnectionFactory**
: Verbindungsfactory - Ein ConnectionFactory-Objekt bindet eine Gruppe von Konfigurationseigenschaften für eine Verbindung ein. Eine Anwendung verwendet eine Verbindungsfactory, um eine Verbindung zu erstellen.

**Verbindung**
: Verbindung - Ein Connection-Objekt bindet die aktive Verbindung einer Anwendung in einen Messaging-Server ein. Eine Anwendung verwendet eine Verbindung, um Sitzungen zu erstellen.

**Sitzung**
: Sitzung - Ein Session-Objekt ist ein Einzelthreadkontext zum Senden und Empfangen von Nachrichten. Eine Anwendung verwendet eine Sitzung, um Nachrichten, Nachrichtenproduzenten und Nachrichtenkonsumenten zu erstellen. Eine Sitzung ist entweder transaktionsbasiert oder nicht transaktionsbasiert.

**Nachricht**
: Nachricht - Ein Message-Objekt bindet eine Nachricht ein, die von einer Anwendung gesendet oder empfangen wird.

**MessageProducer**
: Nachrichtenproduzent - Eine Anwendung verwendet ein MessageProducer-Objekt zum Senden von Nachrichten an ein Ziel.

**MessageConsumer**
: Nachrichtenkonsument - Eine Anwendung verwendet ein MessageConsumer-Objekt zum Empfangen von Nachrichten, die an ein Ziel gesendet wurden.

In Abbildung 53 auf Seite 168 sind diese Objekte und ihre Beziehungen dargestellt.

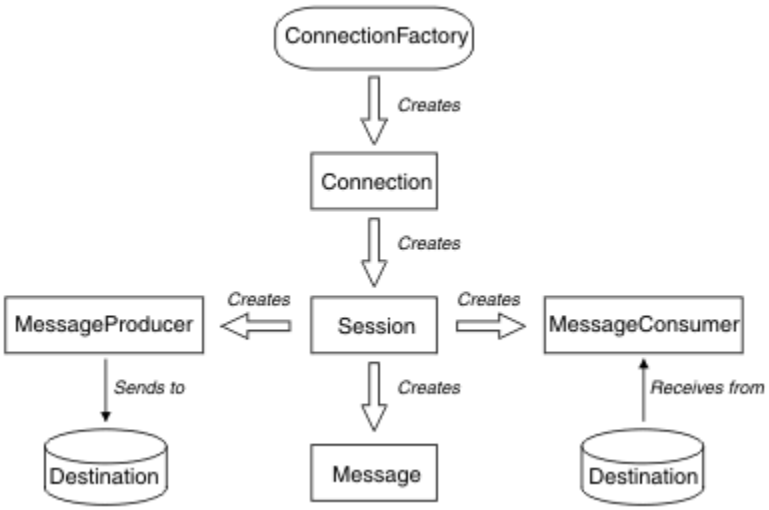



*Abbildung 53. JMS-Objekte und ihre Beziehungen*

Das Diagramm stellt die wichtigsten Schnittstellen dar: ConnectionFactory, Connection, Session, MessageProducer, MessageConsumer, Message und Destination. Eine Anwendung verwendet eine Verbindungsfactory, um eine Verbindung zu erstellen. Für die Erstellung von Sitzungen verwendet sie eine Verbindung. Die Anwendung kann dann eine Sitzung verwenden, um Nachrichten, Nachrichtenproduzenten und Nachrichtenkonsumenten zu erstellen. Die Anwendung verwendet zum Senden von Nachrichten an ein Ziel einen Nachrichtenproduzenten, während sie zum Empfangen von Nachrichten, die an ein Ziel gesendet wurden, einen Nachrichtenkonsumenten verwendet.

Ein Destination-, ConnectionFactory- oder Connection-Objekt kann gleichzeitig von verschiedenen Threads einer Multithread-Anwendung verwendet werden. Ein Session-, MessageProducer- oder MessageConsumer-Objekt kann hingegen nicht gleichzeitig von verschiedenen Threads verwendet werden. Die einfachste Methode, um sicherzustellen, dass ein Session-, MessageProducer- oder MessageConsumer-Objekt nicht gleichzeitig verwendet wird, ist die Erstellung eines separaten Session-Objekts für jeden Thread.

JMS unterstützt zwei Arten von Messaging:

- Punkt-zu-Punkt-Messaging
- Publish/Subscribe-Messaging

Diese Arten des Messaging werden auch als *Messaging-Domänen* bezeichnet. In einer Anwendung können beide Arten kombiniert werden. In einer Punkt-zu-Punkt-Domäne ist das Ziel eine Warteschlange, in einer Publish/Subscribe-Domäne ist es ein Thema.

Bei JMS-Versionen vor JMS 1.1 wird bei der Programmierung für die Punkt-zu-Punkt-Domäne eine Gruppe von Schnittstellen und Methoden verwendet, während bei der Programmierung für die Publish/Subscribe-Domäne eine andere Gruppe verwendet wird. Auch wenn sich beide ähneln, sind sie völlig getrennt voneinander. Ab JMS 1.1 können Sie eine gemeinsame Gruppe von Schnittstellen und Methoden verwenden, die beide Messaging-Domänen unterstützen. Diese gemeinsamen Schnittstellen bieten eine domänenunabhängige Darstellung der Messaging-Domänen. In Tabelle 17 auf Seite 168 sehen Sie eine Übersicht über die domänenunabhängigen JMS-Schnittstellen und deren domänenspezifischen Entsprechungen.

*Tabelle 17. Domänenunabhängige JMS-Schnittstellen und die entsprechenden domänenspezifischen Schnittstellen*

| **Domänenunabhängige Schnittstellen** | **Domänenspezifische Schnittstellen für Punkt-zu-Punkt-Domäne** | **Domänenspezifische Schnittstellen für Publish/Subscribe-Domäne** |
|---|---|---|
| ConnectionFactory | QueueConnectionFactory | TopicConnectionFactory |

*Tabelle 17. Domänenunabhängige JMS-Schnittstellen und die entsprechenden domänenspezifischen Schnittstellen (Forts.)*

| **Domänenunabhängige Schnittstellen** | **Domänenspezifische Schnittstellen für Punkt-zu-Punkt-Domäne** | **Domänenspezifische Schnittstellen für Publish/Subscribe-Domäne** |
|---|---|---|
| Verbindung | QueueConnection | TopicConnection |
| Destination | Warteschlange | Thema |
| Sitzung | QueueSession | TopicSession |
| MessageProducer | QueueSender | TopicPublisher |
| MessageConsumer | QueueReceiver<br>QueueBrowser | TopicSubscriber |

JMS 2.0 IBM MQ classes for JMS 2.0 unterstützt sowohl die früheren domänenspezifischen JMS 1.1 -Schnittstellen als auch die vereinfachte API von JMS 2.0. IBM MQ classes for JMS 2.0 kann daher zur Verwaltung vorhandener Anwendungen verwendet werden, einschließlich der Entwicklung neuer Funktionen in vorhandenen Anwendungen.

JM 3.0 IBM MQ classes for Jakarta Messaging 3.0 unterstützt die Jakarta Messaging -Versionen derselben Schnittstellen und wird für die Entwicklung neuer Anwendungen empfohlen.

In IBM MQ classes for JMS und IBM MQ classes for Jakarta Messagingsind JMS -Objekte wie folgt mit IBM MQ -Konzepten verknüpft:

- Die Eigenschaften eines Connection-Objekts sind von den Eigenschaften der Verbindungsfactory abgeleitet, die zur Erstellung der Verbindung verwendet wurde. Diese Eigenschaften legen fest, wie eine Anwendung eine Verbindung zu einem Warteschlangenmanager herstellt. Beispiele für diese Eigenschaften sind der Name des Warteschlangenmanagers und bei Anwendungen, die sich im Clientmodus mit dem Warteschlangenmanager verbinden, der Hostname oder die IP-Adresse des Systems, auf dem der Warteschlangenmanager ausgeführt wird.
- Ein Session-Objekt umfasst eine IBM MQ-Verbindungskennung, die wiederum den Transaktionsbereich der Sitzung festlegt.
- Sowohl MessageProducer-Objekt als auch MessageConsumer-Objekt umfassen eine IBM MQ-Objektkennung.

Bei Verwendung von IBM MQ classes for JMS oder IBM MQ classes for Jakarta Messaginggelten alle normalen Regeln von IBM MQ . Beachten Sie insbesondere, dass eine Anwendung zwar Nachrichten an eine ferne Warteschlange senden kann, aber nur Nachrichten aus einer Warteschlange empfangen kann, die zu dem Warteschlangenmanager gehört, mit dem die Anwendung verbunden ist.

Die JMS-Spezifikation erwartet, dass es sich bei den ConnectionFactory- und Destination-Objekten um verwaltete Objekte handelt. Ein Administrator erstellt und pflegt verwaltete Objekte in einem zentralen Repository, aus dem diese Objekte von JMS-Anwendungen über die Java Naming and Directory Interface (JNDI)-Schnittstelle abgerufen werden.

In IBM MQ classes for JMS und IBM MQ classes for Jakarta Messagingist die Implementierung der Destination-Schnittstelle eine abstrakte Superklasse von Queue und Topic, sodass eine Instanz von Destination entweder ein Queue-Objekt oder ein Topic-Objekt ist. Die domänenunabhängigen Schnittstellen behandeln eine Warteschlange (Queue) oder ein Thema (Topic) hingegen als Ziel (Destination). Die Messaging-Domäne eines MessageProducer- oder MessageConsumer-Objekts bestimmt sich daraus, ob das Ziel eine Warteschlange oder ein Thema ist.

In IBM MQ classes for JMS und IBM MQ classes for Jakarta Messaging können daher Objekte der folgenden Typen verwaltete Objekte sein:

- ConnectionFactory

- QueueConnectionFactory
- TopicConnectionFactory
- Warteschlange
- Thema
- XAConnectionFactory
- XAQueueConnectionFactory
- XATopicConnectionFactory

## IBM MQ classes for JMS/Jakarta Messaging -Architektur

IBM MQ classes for JMS und IBM MQ classes for Jakarta Messaging verfügen über eine Schichtarchitektur. Die oberste Codeebene ist eine allgemeine Ebene, die jeder IBM Java -Messaging-Provider verwenden kann.

JM 3.0 V 9.3.0 V 9.3.0 IBM MQ 9.3.0 bietet jetzt Unterstützung für Jakarta Messaging 3.0. JMS 2.0 wird weiterhin vollständig unterstützt.

IBM MQ classes for JMS und IBM MQ classes for Jakarta Messaging verfügen über eine Schichtarchitektur, wie im Diagramm Abbildung 54 auf Seite 170dargestellt. Die oberste Codeebene ist eine allgemeine Ebene, die von jedem IBM JMS -oder Jakarta Messaging-Provider verwendet werden kann. Wenn eine Anwendung eine Methode JMS oder Jakarta Messagingaufruft, wird jede Verarbeitung des Aufrufs, die nicht für ein Messaging-System spezifisch ist, von der allgemeinen Schicht ausgeführt, die auch eine konsistente Antwort auf den Aufruf bereitstellt. Die gesamte Verarbeitung des Aufrufs, die sich speziell auf ein Messaging-System bezieht, wird an eine untergeordnete Schicht delegiert. Im folgenden Diagramm ist der IBM MQ-Messaging-Provider in der unteren Schicht gemeinsam mit zwei weiteren Messaging-Providern (Messaging-Provider A und Messaging-Provider B) dargestellt.

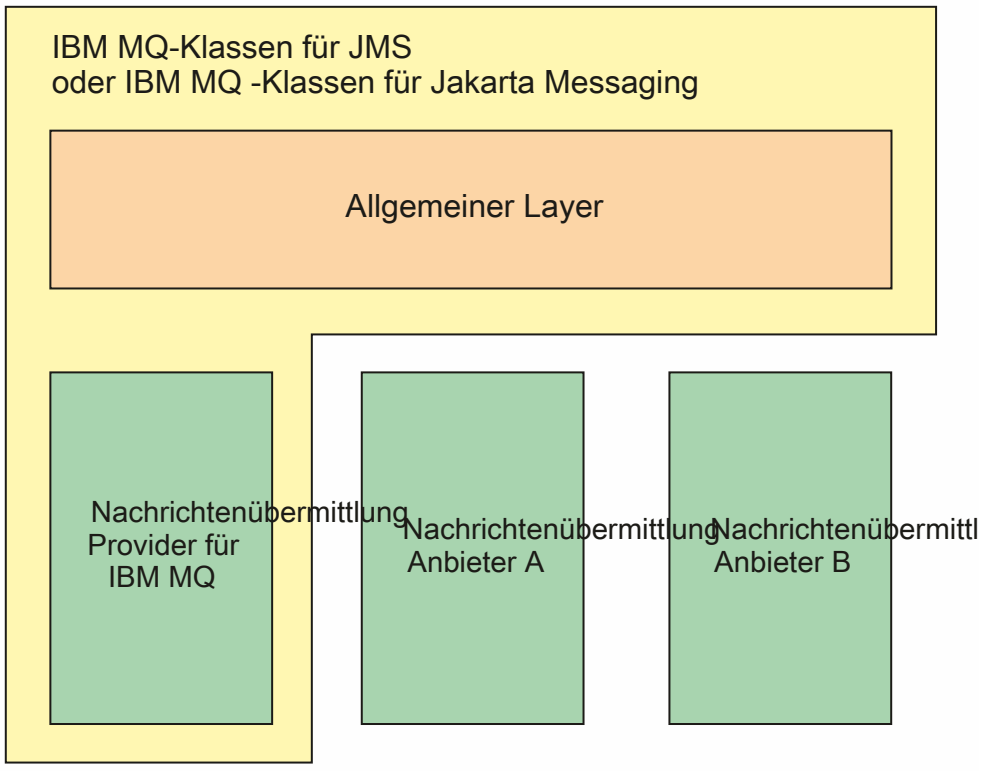



*Abbildung 54. Schichtarchitektur für IBM JMS -und Jakarta Messaging -Provider*

Eine Schichtarchitektur hat folgende Vorteile:

- Zur Verbesserung der Konsistenz des Verhaltens der verschiedenen IBM JMS -und Jakarta Messaging -Provider
- Vereinfachung des Schreibens einer Bridge-Anwendung zwischen zwei IBM Messaging-Systemen
- Um das Portieren einer Anwendung von einem IBM JMS -oder Jakarta Messaging -Provider auf einen anderen zu vereinfachen

**Zugehörige Tasks**

IBM MQ -Klassen für JMS/Jakarta Messaging verwenden

## Unterstützung für verwaltete Objekte

IBM MQ classes for JMS und IBM MQ classes for Jakarta Messaging unterstützen die Verwendung verwalteter Objekte.

JM 3.0 V 9.3.0 V 9.3.0 Ab IBM MQ 9.3.0wird Jakarta Messaging 3.0 für die Entwicklung neuer Anwendungen unterstützt. IBM MQ 9.3.0 unterstützt weiterhin JMS 2.0 für vorhandene Anwendungen. Die Verwendung der Jakarta Messaging 3.0 -API und der JMS 2.0 -API in derselben Anwendung wird nicht unterstützt. Weitere Informationen finden Sie unter Using IBM MQ classes for JMS/Jakarta Messaging.

Der Logikablauf innerhalb einer JMS -oder IBM MQ classes for Jakarta Messaging -Anwendung beginnt mit ConnectionFactory -und Destination-Objekten. Mit dem Objekt ConnectionFactory erstellt die Anwendung ein Connection-Objekt, das die aktive Verbindung von der Anwendung zum Messaging-Server darstellt. Mit dem Objekt Connection erstellt die Anwendung ein Session-Objekt, das ein einzelner Threaded-Kontext für die Produzierung und Konsumierung von Nachrichten ist. Aus diesem Session-Objekt erstellt die Anwendung dann mit einem Destination-Objekt ein MessageProducer-Objekt, mit dessen Hilfe sie Nachrichten an das angegebene Ziel senden kann. Das Ziel ist entweder eine Warteschlange oder ein Thema im Messaging-System, das im Destination-Objekt gekapselt ist. Aus dem Session- und dem Destination-Objekt kann die Anwendung auch ein MessageConsumer-Objekt erstellen, über das sie die an das angegebene Ziel gesendeten Nachrichten empfängt.

Die Spezifikationen JMS und Jakarta Messaging erwarten, dass ConnectionFactory -und Destination-Objekte verwaltete Objekte sind. Ein Administrator erstellt und verwaltet verwaltete Objekte in einem zentralen Repository und eine JMS -oder Jakarta Messaging -Anwendung ruft diese Objekte mithilfe der Java Naming Directory Interface (JNDI) ab. Das Repository der verwalteten Objekte kann von einer einfachen Datei bis zu einem LDAP-Verzeichnis (Lightweight Directory Access Protocol) reichen.

IBM MQ classes for JMS und IBM MQ classes for Jakarta Messaging unterstützen die Verwendung verwalteter Objekte. Eine Anwendung kann alle Funktionen von IBM MQ classes for JMS oder IBM MQ classes for Jakarta Messaging verwenden, die über IBM MQ bereitgestellt werden, ohne dass IBM MQspezifische Informationen fest in der Anwendung selbst codiert sind. Dadurch bleibt die Anwendung bis zu einem gewissen Grad von der zugrundeliegenden IBM MQ-Konfiguration unabhängig.

Um diese Unabhängigkeit zu erreichen, kann die Anwendung JNDI verwenden, um Verbindungsfactorys und Ziele abzurufen, die als verwaltete Objekte gespeichert sind, und nur die im Paket `javax.jms` (JMS 2.0) oder `jakarta.jms` (Jakarta Messaging 3.0) definierten Schnittstellen verwenden, um Messaging-Operationen auszuführen.

JMS 2.0 Für JMS 2.0kann ein Administrator das IBM MQ JMS -Verwaltungstool **JMSAdmin**oder IBM MQ Explorerverwenden, um verwaltete Objekte in einem zentralen Repository zu erstellen und zu verwalten.

JM 3.0 Für Jakarta Messaging 3.0können Sie JNDI nicht mit IBM MQ Explorerverwalten. Die JNDI-Verwaltung wird von der Variante Jakarta Messaging 3.0 von **JMSAdmin**( **JMS30Admin**) unterstützt.

Ein Anwendungsserver stellt normalerweise ein eigenes Repository für verwaltete Objekte und eigene Tools für die Erstellung und Verwaltung der Objekte bereit. Eine Java EE JM 3.0 -oder Jakarta EE -Anwendung kann daher JNDI verwenden, um verwaltete Objekte entweder aus dem Repository des Anwendungsservers oder aus einem zentralen Repository abzurufen.

**Zugehörige Tasks**

JMS-und Jakarta Messaging-Ressourcen konfigurieren

## Unterstützte Kommunikationstypen auf Java EE -und Jakarta EE -Plattformen

Auf den Plattformen Java EE und Jakarta EE unterstützen IBM MQ classes for JMS und IBM MQ classes for Jakarta Messaging zwei Arten der Kommunikation zwischen einer Komponente einer Anwendung und einem IBM MQ -Warteschlangenmanager.

JM 3.0 V 9.3.0 V 9.3.0 IBM MQ 9.3.0 bietet jetzt Unterstützung für Jakarta Messaging 3.0. JMS 2.0 wird weiterhin vollständig unterstützt. Da JMS und Jakarta Messaging viel gemeinsam genutzt werden, können weitere Verweise auf JMS in diesem Abschnitt als Verweise auf beides betrachtet werden. Alle Unterschiede werden nach Bedarf hervorgehoben.

Die folgenden beiden Arten der Kommunikation zwischen einer Anwendungskomponente und einem IBM MQ-Warteschlangenmanager werden unterstützt:

- Abgehende Kommunikation
- Eingehende Kommunikation

### Abgehende Kommunikation

Mithilfe der JMS -oder Jakarta Messaging -API erstellt eine Anwendungskomponente eine Verbindung zu einem Warteschlangenmanager und sendet und empfängt anschließend Nachrichten.

Bei der Anwendungskomponente kann es sich zum Beispiel um einen Anwendungsclient, ein Servlet, eine Java Server Page (JSP), ein Enterprise Java Bean (EJB) oder ein Message-driven Bean (MDB) handeln. Bei dieser Art der Kommunikation stellt der Anwendungsserver-Container nur Low-Level-Funktionen für Messaging-Operationen bereit, beispielsweise Verbindungspooling und Thread-Management.

### Eingehende Kommunikation

Im Falle der eingehenden Kommunikation wird eine auf einem Ziel eingehende Nachricht einem MDB zugestellt, der diese Nachricht dann verarbeitet.

Java EE JM 3.0 -und Jakarta EE -Anwendungen verwenden MDBs, um Nachrichten asynchron zu verarbeiten. Ein MDB fungiert als JMS-Nachrichtenlistener und wird durch eine onMessage()-Methode implementiert, die festlegt, wie eine Nachricht verarbeitet wird. Ein MDB wird im EJB-Container des Anwendungsservers implementiert. Die Konfiguration eines MDB ist vom verwendeten Anwendungsserver abhängig. Jedoch muss in der Konfiguration angegeben sein, mit welchem Warteschlangenmanager die Verbindung hergestellt werden soll, wie diese Verbindung erfolgen soll, welches Ziel auf Nachrichten überwacht werden soll und wie sich das MDB hinsichtlich Transaktionen verhalten soll. Diese Informationen werden vom EJB-Container verwendet. Wenn eine Nachricht, die die Auswahlkriterien der MDB erfüllt, am angegebenen Ziel ankommt, verwendet der EJB-Container IBM MQ classes for JMS oder IBM MQ classes for Jakarta Messaging , um die Nachricht vom Warteschlangenmanager abzurufen, und stellt die Nachricht dann der MDB zu, indem er ihre Methode onMessage() aufruft.

## Beziehung zu IBM MQ classes for Java

IBM MQ classes for Java, IBM MQ classes for Jakarta Messaging und IBM MQ classes for JMS sind Peers, die eine gemeinsame Java -Schnittstelle zur MQI verwenden.

Abbildung 55 auf Seite 173 zeigt die Beziehung zwischen IBM MQ classes for JMS, IBM MQ classes for Jakarta Messaging und IBM MQ classes for Java.

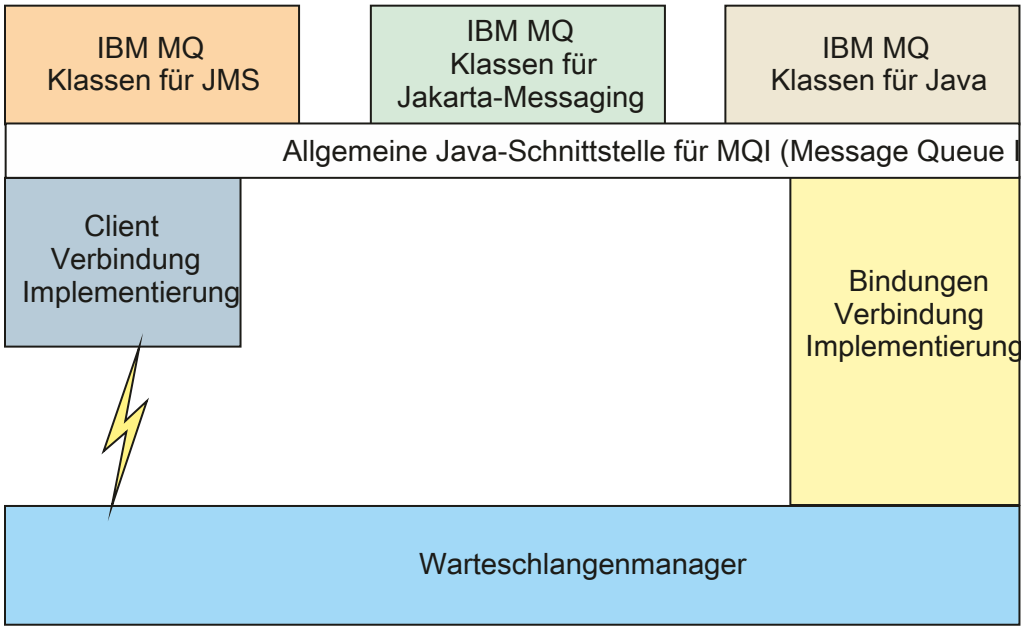



*Abbildung 55. Beziehung zwischen IBM MQ classes for JMS, IBM MQ classes for Jakarta Messaging und IBM MQ classes for Java*

Im Allgemeinen sollten Java -Programme nur eine Schnittstelle für die Schnittstelle mit IBM MQ - IBM MQ classes for Java, IBM MQ classes for Jakarta Messaging oder IBM MQ classes for JMSverwenden. Das Mischen von Schnittstellen wird nicht unterstützt, mit einer Ausnahme. Zur Aufrechterhaltung der Kompatibilität mit Versionen vor IBM WebSphere MQ 7.0 können in Java geschriebene Kanalexit-Klassen nach wie vor die IBM MQ classes for Java-Schnittstellen verwenden, selbst wenn die Kanalexit-Klassen aus IBM MQ classes for JMS aufgerufen werden. Die Verwendung der IBM MQ classes for Java -Schnittstellen bedeutet jedoch, dass Ihre Anwendungen weiterhin von folgenden Komponenten abhängig sind:

- JMS 2.0 Die IBM MQ classes for Java -JAR-Datei `com.ibm.mq.jar` Wenn Sie `com.ibm.mq.jar` nicht in Ihren Klassenpfad aufnehmen möchten, können Sie stattdessen die Schnittstellengruppe im Paket `com.ibm.mq.exits` verwenden.
- JM 3.0 V 9.3.0 V 9.3.0 Verwendung von `com.ibm.mq.jakarta.client.jar`bei der Interaktion mit IBM MQ classes for Jakarta Messaging.

**Zugehörige Konzepte**

V 9.3.0 V 9.3.0 Warum sollte ich IBM MQ -Klassen für Jakarta Messaging verwenden?

Vorteile von IBM MQ-Klassen für JMS

Warum sollte ich IBM MQ -Klassen für Javaverwenden?

## IBM MQ-Messaging-Provider

Der IBM MQ-Messaging-Provider hat drei Betriebsmodi, den Normalmodus, den Normalmodus mit Einschränkungen und den Migrationsmodus.

Der IBM MQ-Messaging-Provider hat drei Betriebsmodi:

- Normalmodus des IBM MQ-Messaging-Providers
- Normalmodus des IBM MQ-Messaging-Providers mit Einschränkungen
- Migrationsmodus des IBM MQ-Messaging-Providers

Der IBM MQ-Normalmodus für Messaging-Provider nutzt alle Funktionen eines IBM MQ-Warteschlangenmanagers, um JMS zu implementieren. Dieser Modus ist für die Verwendung der API und Funktionalität von JMS 2.0 JM 3.0 V 9.3.0 V 9.3.0 oder Jakarta Messaging 3.0 optimiert.

Wenn:

- Der Client gibt die Providerversion 6 auf einem **ConnectionFactory**an. Der Client verhält sich in einer Weise, die mit dem mit IBM WebSphere MQ 6.0bereitgestellten Client kompatibel ist. Es werden nur JMS 1.1 -und JMS 2-Schnittstellen unterstützt, aber einige JMS 2-Funktionen, wie z. B. gemeinsam genutzte Subskriptionen, Zustellungsverzögerung und asynchrones Senden, sind inaktiviert. Es gibt keine gemeinsame Nutzung von Verbindungen.
- Der Client gibt die Providerversion 7 in einem **ConnectionFactory**an. Die JMS-Schnittstellen 1.1 und JMS 2 werden vollständig unterstützt.
- Es wurde keine Providerversion angegeben. Es wird versucht, die Verbindung zu Provider Version 7 herzustellen. Wenn dies fehlschlägt, wird ein weiterer Versuch mit Provider Version 6 unternommen.

Wenn Sie eine Verbindung mit IBM Integration Bus über IBM MQ Enterprise Transport herstellen möchten, verwenden Sie den Migrationsmodus. Bei Verwendung von IBM MQ Real-Time Transport ist der Migrationsmodus automatisch ausgewählt, da Sie im Verbindungsfactory-Objekt explizit Eigenschaften ausgewählt haben. Die Verbindung mit IBM Integration Bus über IBM MQ Enterprise Transport folgt den allgemeinen Regeln der Modusauswahl, die im Abschnitt JMS-Eigenschaft **PROVIDERVERSION** konfigurieren beschrieben ist.

**Zugehörige Tasks**

JMS-Ressourcen konfigurieren

## z/OS IBM MQ for z/OS - Konzepte

Einige der von IBM MQ for z/OS verwendeten Konzepte gelten nur für die z/OS-Plattform. Beispielsweise werden der Protokollierungsmechanismus, die Speicherverwaltungsverfahren, die Disposition der Arbeitseinheit mit Wiederherstellung und die Gruppen mit gemeinsamer Warteschlange nur in IBM MQ for z/OS bereitgestellt. Verwenden Sie dieses Thema als Einführung in weitere Informationen zu diesen Konzepten.

Die Konzepte enthalten eine Übersicht über die Objekte, die von IBM MQ for z/OSverwendet werden, einschließlich der folgenden:

- Der Warteschlangenmanager
- Kanalinitiator
- Gemeinsam genutzte Warteschlangen und Gruppen mit gemeinsamer Warteschlange
- Einreihung in Warteschlange innerhalb von Gruppen

Die folgenden Themen decken auch verschiedene Prozeduren ab, die Sie benötigen, einschließlich:

- Systemdefinitionen unter z/OS
- Speicherverwaltung
- Neustart und Wiederherstellung
- Sicherheitskonzepte in IBM MQ for z/OS

**Zugehörige Konzepte**

„Warteschlangenmanager unter z/OS” auf Seite 175
Damit Ihre Anwendungsprogramme IBM MQ auf Ihrem z/OS-System verwenden können, müssen Sie zunächst das Produkt IBM MQ for z/OS installieren und einen Warteschlangenmanager starten. Der Warteschlangenmanager ist der Besitzer und Verwalter der Ressourcen, die von IBM MQ verwendet werden.

„Kanalinitiator unter z/OS” auf Seite 176
Vom Kanalinitiator werden Ressourcen zur Verfügung gestellt und verwaltet, die eine verteilte Steuerung von Warteschlangen in IBM MQ ermöglichen. IBM MQ verwendet *Nachrichtenkanalagenten* (MCAs), um Nachrichten von einem Warteschlangenmanager an einen anderen zu senden.

„Begriffe und Tasks für die Verwaltung von IBM MQ for z/OS” auf Seite 178
Dieser Abschnitt dient als Einleitung zu der Terminologie und den Tasks, die speziell für IBM MQ for z/OS gelten.

„Gemeinsam genutzte Warteschlangen und Gruppen mit gemeinsamer Warteschlange” auf Seite 181
Mithilfe von gemeinsam genutzten Warteschlangen und Gruppen mit gemeinsamer Warteschlange können Sie die Hochverfügbarkeit von IBM MQ-Ressourcen implementieren. Bei gemeinsam genutzten Warteschlangen und Gruppen mit gemeinsamer Warteschlange handelt es sich um Funktionen, die nur in IBM MQ for z/OS auf der Plattform z/OS verfügbar sind.

„Einreihung in Warteschlange innerhalb von Gruppen” auf Seite 232
In diesem Abschnitt wird die gruppeninterne Warteschlangensteuerung beschrieben, eine Funktion von IBM MQ for z/OS, die für die z/OS-Plattform einzigartig ist. Diese Funktion steht nur Warteschlangenmanagern zur Verfügung, die in einer Gruppe mit gemeinsamer Warteschlange definiert sind.

„Speicherverwaltung unter z/OS” auf Seite 248
IBM MQ for z/OS erfordert permanente und temporäre Datenstrukturen und speichert diese Dateien in Seitengruppen und Hauptspeicherpuffern. In den folgenden Abschnitten wird ausführlich erläutert, wie IBM MQ diese Seitengruppen und Puffer verwendet.

„Protokollierung in IBM MQ for z/OS” auf Seite 253
IBM MQ verwaltet *Protokolle*, in denen Datenänderungen und signifikante Ereignisse aufgezeichnet werden, sobald sie auftreten. Anhand dieser Protokolle können die Daten bei Bedarf auf einen vorherigen Stand zurückgesetzt werden.

„Wiederherstellung und Neustart unter z/OS” auf Seite 278
Über die Links in diesem Abschnitt finden Sie weitere Informationen zu den Funktionen in IBM MQ for z/OS für Neustart und Wiederherstellung.

„Sicherheitskonzepte in IBM MQ for z/OS” auf Seite 297
Mithilfe dieses Abschnitts können Sie den Stellenwert der Sicherheit für IBM MQ und auch die Auswirkungen verstehen, wenn das System über keine adäquaten Sicherheitseinstellungen verfügt.

„Verfügbarkeit unter z/OS” auf Seite 304
IBM MQ for z/OS verfügt über viele Funktionen für Hochverfügbarkeit. In diesem Abschnitt werden einige Überlegungen zur Verfügbarkeit erläutert.

„Disposition einer Arbeitseinheit mit Wiederherstellung unter z/OS” auf Seite 309
Bestimmte Transaktionsanwendungen können, wenn sie mit einem Warteschlangenmanager in einer Gruppe mit gemeinsamer Warteschlange (Queue Sharing Group; QSG) verbunden sind, eine Disposition 'GROUP' statt einer Disposition 'QMGR' der Arbeitseinheit mit Wiederherstellung verwenden, indem bei der Verbindungsherstellung der QSG-Name statt des Warteschlangenmanagernamens angegeben wird. Damit wird die Wiederherstellung von Transaktionen flexibler und leistungsfähiger, da die Anforderung wegfällt, die Verbindung mit demselben Warteschlangenmanager in der QSG wiederherzustellen.

**Zugehörige Verweise**

„Systemdefinition unter z/OS” auf Seite 265
In IBM MQ for z/OS werden viele Standardobjektdefinitionen verwendet und eine Beispiel-Jobsteuersprache zum Erstellen dieser Standardobjekte bereitgestellt. In diesem Abschnitt werden diese Standardobjekte sowie die Beispiel-Jobsteuersprache erläutert.

„Überwachung und Statistik in IBM MQ for z/OS” auf Seite 308
IBM MQ for z/OS verfügt über eine Reihe von Funktionen zur Überwachung des Warteschlangenmanagers und zur Erfassung von statistischen Daten.

# z/OS Warteschlangenmanager unter z/OS

Damit Ihre Anwendungsprogramme IBM MQ auf Ihrem z/OS-System verwenden können, müssen Sie zunächst das Produkt IBM MQ for z/OS installieren und einen Warteschlangenmanager starten. Der Warteschlangenmanager ist der Besitzer und Verwalter der Ressourcen, die von IBM MQ verwendet werden.

## Der Warteschlangenmanager

Ein *Warteschlangenmanager* ist ein Programm, das für Anwendungen einen Nachrichtenservice bereitstellt. Anwendungen, die die Schnittstelle für Nachrichtenwarteschlangen (MQI; Message Queue Interface) verwenden, können Nachrichten in Warteschlangen einreihen und von dort abrufen. Der Warteschlangenmanager stellt sicher, dass Nachrichten an die richtige Warteschlange gesendet oder an einen

anderen Warteschlangenmanager weitergeleitet werden. Er verarbeitet sowohl die MQI-Aufrufe, die an ihn ausgegeben werden, als auch die an ihn übergebenen Befehle (beliebigen Ursprungs). Er generiert für jeden Aufruf oder Befehl die entsprechenden Beendigungscodes.

Der Warteschlangenmanager verwaltet unter anderem folgende Ressourcen:

- Seitengruppen, in denen die IBM MQ-Objektdefinitionen und Nachrichtendaten gespeichert sind
- Protokolle, die bei einem Fehlschlag des Warteschlangenmanagers zur Wiederherstellung von Nachrichten und Objekten verwendet werden
- Hauptspeicher
- Verbindungen, über die unterschiedliche Anwendungsumgebungen (CICS, IMS und Batch) auf die IBM MQ-API zugreifen können
- Der IBM MQ-Kanalinitiator, der die Kommunikation zwischen IBM MQ auf Ihrem z/OS-System und anderen Systemen ermöglicht

Der Warteschlangenmanager hat einen Namen, anhand dessen Anwendungen eine Verbindung zu ihm herstellen können.

Abbildung 56 auf Seite 176 zeigt einen Warteschlangenmanager mit Verbindungen zu unterschiedlichen Anwendungsumgebungen und den Kanalinitiator.

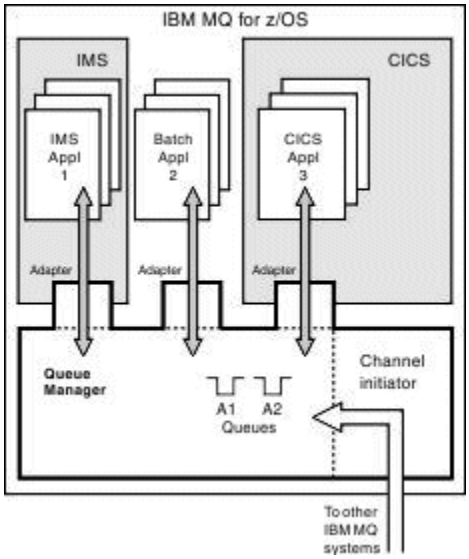



*Abbildung 56. Überblick über IBM MQ for z/OS*

## Warteschlangenmanagersubsystem unter z/OS

Unter z/OS wird IBM MQ als z/OS-Subsystem ausgeführt, das zum Zeitpunkt des einleitenden Programmladens (IPL) gestartet wird. Innerhalb des Subsystems wird der Warteschlangenmanager durch Ausführen einer Prozedur in der Jobsteuersprache gestartet, in der die z/OS-Dateien angegeben sind, die Informationen zu Protokollen sowie die Objektdefinitionen und Nachrichtendaten (die Seitengruppen) enthalten. Das Subsystem und der Warteschlangenmanager haben denselben Namen, der aus maximal vier Zeichen bestehen kann. Alle Warteschlangenmanager in einem Netz müssen eindeutige Namen haben, selbst wenn sie sich in unterschiedlichen Systemen, Sysplex-Installationen oder auf unterschiedlichen Plattformen befinden.

## z/OS Kanalinitiator unter z/OS

Vom Kanalinitiator werden Ressourcen zur Verfügung gestellt und verwaltet, die eine verteilte Steuerung von Warteschlangen in IBM MQ ermöglichen. IBM MQ verwendet *Nachrichtenkanalagenten* (MCAs), um Nachrichten von einem Warteschlangenmanager an einen anderen zu senden.

Um Nachrichten von Warteschlangenmanager A an Warteschlangenmanager B zu senden, muss der *sendende* Nachrichtenkanalagent (MCA) in Warteschlangenmanager A eine Kommunikationsverbindung zu Warteschlangenmanager B einrichten. Der *empfangende* Nachrichtenkanalagent muss in Warteschlangenmanager B gestartet werden, damit Nachrichten über die Kommunikationsverbindung empfangen werden können. Diese unidirektionale Route, die aus dem sendenden Nachrichtenkanalagenten, der Kommunikationsverbindung und dem empfangenden Nachrichtenkanalagenten besteht, wird als *Kanal* bezeichnet. Der sendende Nachrichtenkanalagent ruft Nachrichten aus einer Übertragungswarteschlange ab und sendet sie über den Kanal an den empfangenden Nachrichtenkanalagenten. Der empfangende Nachrichtenkanalagent empfängt die Nachrichten und reiht sie in die Zielwarteschlangen ein.

In IBM MQ for z/OS werden sowohl die sendenden als auch die empfangenden Nachrichtenkanalagenten innerhalb des Kanalinitiators (der auch als *Verlagerungsfunktion* bezeichnet wird) ausgeführt. Der Kanalinitiator wird als ein z/OS-Adressraum ausgeführt und durch den Warteschlangenmanager gesteuert. Mit einem Warteschlangenmanager kann immer nur ein Kanalinitiator verbunden sein, der in demselben z/OS-Image ausgeführt werden muss wie der Warteschlangenmanager. Innerhalb eines Kanalinitiators können Tausende von Nachrichtenkanalagentenprozessen gleichzeitig ausgeführt werden.

Abbildung 57 auf Seite 177 zeigt zwei Warteschlangenmanager innerhalb eines Sysplex. Beide Warteschlangenmanager haben je einen Kanalinitiator und eine lokale Warteschlange. Von Warteschlangenmanagern unter AIX und Windows gesendete Nachrichten werden in die lokale Warteschlange eingereiht und von dort durch eine Anwendung abgerufen. Antwortnachrichten werden über eine ähnliche Route zurückgegeben.

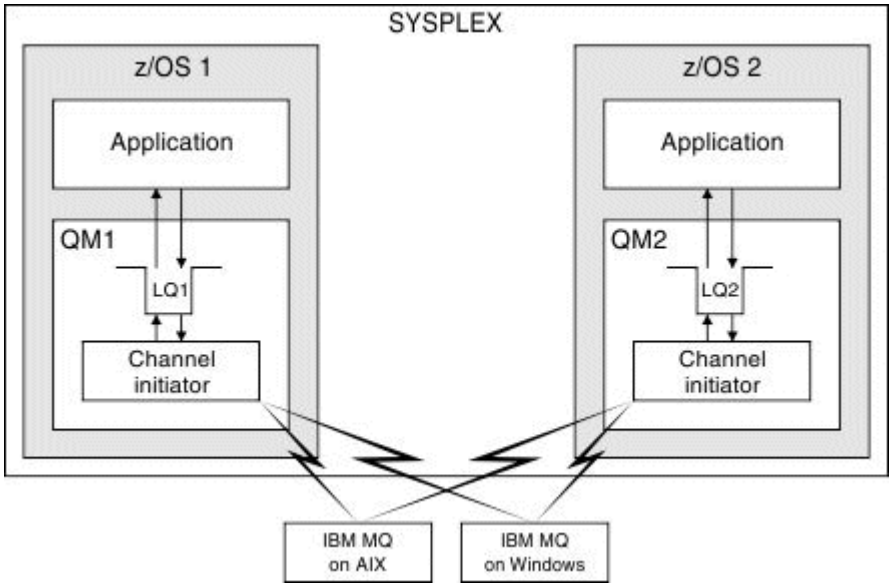



*Abbildung 57. Kommunikation zwischen Warteschlangenmanagern*

Der Kanalinitiator führt auch andere Prozesse aus, die mit der Verwaltung von Kanälen zu tun haben. Dazu gehören folgende Prozesse:

**Empfangsprogramme**
Diese Prozesse sind in einem Kommunikationssubsystem, z. B. TCP, empfangsbereit für eingehende Kanalanforderungen und starten einen benannten Nachrichtenkanalagenten, wenn eingehende Anforderungen empfangen werden.

**Supervisor-Funktion**
Diese Funktion verwaltet den Adressraum des Kanalinitiators und ist beispielsweise für den Neustart des Kanals nach einem Fehler zuständig.

**Namensserver**
Wird zum Auflösen von TCP-Namen in Adressen verwendet.

**TLS-Tasks**
Werden für die Verschlüsselung und Entschlüsselung sowie zur Überprüfung von Zertifikatswiderrufslisten verwendet.

## z/OS SMF-Datensätze für den Kanalinitiator

Der Kanalinitiator (CHINIT) kann SMF-Statistikdatensätze und Abrechnungsdatensätze mit Informationen zu Tasks und Kanälen erstellen.

Der Kanalinitiator (CHINIT) kann SMF-Statistikdatensätze und Abrechnungsdatensätze mit den folgenden Arten von Informationen erstellen:

- Tasks: Dispatcher, Adapter, Domain Name Server (DNS) und SSL. Diese Tasks bilden gemeinsam die sogenannten CHINIT-Statistikdaten.
- Kanäle: Es werden Abrechnungsdaten zur Verfügung gestellt, die den Daten ähneln, die über den Befehl DIS CHSTATUS verfügbar sind. Dies wird als "Kanalabrechnung" bezeichnet.

IBM MQ for Multiplatforms stellt ähnliche Informationen bereit, indem PCF-Nachrichten in das SYSTEM.ADMIN.STATISTICS.QUEUE. Unter Kanalstatistiknachrichtendaten finden Sie weitere Informationen darüber, wie Statistikdaten in IBM MQ for Multiplatforms erfasst werden.

### Statistikdaten

Mithilfe dieser Informationen erfahren Sie Folgendes:

- Ob Sie mehr CHINIT-Tasks, beispielsweise mehrere SSL-TCBs, benötigen. Außerdem erhalten Sie Angaben zur CPU-Belegung durch diese Tasks.
- Die durchschnittliche Anforderungsdauer bei diesen Tasks.
- Die Anforderung mit der längsten Dauer im Intervall und die Tageszeit dieser Situation bei DNS- und SSL-Tasks. Sie können diese Tageszeit mit Problemen korrelieren, die möglicherweise im Kanal auftreten.

### Abrechnungsdaten

Mithilfe dieser Informationen können Sie die Kanalnutzung überwachen und erfahren Folgendes:

- Die Kanäle mit dem höchsten Durchsatz.
- Die Geschwindigkeit, mit der Nachrichten gesendet wurden, sowie die Geschwindigkeit für das Senden von Daten in MB/Sekunde.
- Die erreichte Stapelgröße. Wenn sich die erreichte Stapelgröße der für den Kanal angegebenen Stapelgröße nähert, hat der Kanal seine Grenze für das Senden von Nachrichten möglicherweise nahezu erreicht.

Mit den Befehlen START TRACE und STOP TRACE steuern Sie die Erfassung des Abrechnungstrace und des Statistiktrace. Sie können die Optionen STATCHL und STATACLS auf dem Kanal und Warteschlangenmanager verwenden, um zu steuern, ob Kanäle SMF-Daten erstellen.

## z/OS Begriffe und Tasks für die Verwaltung von IBM MQ for z/OS

Dieser Abschnitt dient als Einleitung zu der Terminologie und den Tasks, die speziell für IBM MQ for z/OS gelten.

Einige der für die Verwaltung von IBM MQ for z/OS erforderlichen Konzepte und Tasks gelten nur für die z/OS-Plattform. In der folgenden Liste sind einige dieser Konzepte und Tasks aufgeführt.

- Gemeinsam genutzte Warteschlangen
- Seitengruppen und Pufferpools
- Protokollierung
- Anpassung der Warteschlangenmanager-Umgebung

- Neustart und Wiederherstellung
- Sicherheit
- Verfügbarkeit
- Bearbeitung von Objekten
- Überwachung und Statistik
- Anwendungsumgebungen

## Gemeinsam genutzte Warteschlangen

Warteschlangen können entweder *nicht gemeinsam genutzt* sein, d. h., Besitzer der Warteschlange ist ein einzelner Warteschlangenmanager, der die alleinige Zugriffsberechtigung hat, oder *gemeinsam genutzt*, d. h., Besitzer der Warteschlange ist eine *Gruppe mit gemeinsamer Warteschlange* (Queue-Sharing Group, QSG). Eine Gruppe mit gemeinsamer Warteschlange umfasst mehrere Warteschlangenmanager, die in einem gemeinsamen z/OS-Sysplex ausgeführt werden und gleichzeitig auf dieselben IBM MQ-Objektdefinitionen und Nachrichtendaten zugreifen können. Innerhalb einer Gruppe mit gemeinsamer Warteschlange sind die gemeinsam nutzbaren Objektdefinitionen in einer gemeinsam genutzten Db2-Datenbank gespeichert. Die Nachrichten der gemeinsam genutzten Warteschlange sind in einer oder mehreren Coupling-Facility-Strukturen (CF-Strukturen) gespeichert. Wenn die Nachrichtendaten zu umfangreich sind (ab einer Größe von 63 KB), um direkt in der Struktur gespeichert zu werden, oder wenn die Nachricht aufgrund ihrer Größe gemäß den bei der Installation definierten Regeln für eine Auslagerung ausgewählt wird, werden zwar die Nachrichtensteuerungsinformationen im Coupling-Facility-Eintrag gespeichert, doch die Nachrichtendaten werden in eine gemeinsam genutzte Nachrichtendatei (Shared Message Data Set, SMDS) oder eine gemeinsam genutzte Db2-Datenbank ausgelagert. Die gemeinsam genutzten Nachrichtendateien, die gemeinsam genutzte Db2-Datenbank und die Coupling-Facility-Strukturen sind Ressourcen, die von allen Warteschlangenmanagern in der Gruppe gemeinschaftlich verwaltet werden.

## Seitengruppen und Pufferpools

Wenn eine Nachricht in eine nicht gemeinsam genutzte Warteschlange eingereiht wird, speichert der Warteschlangenmanager die Daten so in einer Seitengruppe, dass sie wieder abgerufen werden können, wenn eine nachfolgende Operation eine Nachricht aus derselben Warteschlange empfängt. Wenn die Nachricht aus der Warteschlange entfernt wurde, wird der Speicherplatz in der Seitengruppe, der die zugehörigen Daten enthält, später für die Wiederverwendung freigegeben. Je mehr Nachrichten sich in einer Warteschlange befinden, desto mehr Speicherplatz ist in der Seitengruppe belegt, und je weniger Nachrichten sich in der Warteschlange befinden, desto weniger Speicherplatz ist in der Seitengruppe belegt.

Um die Leistungseinbußen durch Schreib- und Lesevorgänge in den Seitengruppen zu verringern, puffert der Warteschlangenmanager die Datenaktualisierungen im Hauptspeicher. Die Größe des Speicherplatzes, der zum Puffern der Schreib-/Lesezugriffe auf die Seitengruppen erforderlich ist, wird durch IBM MQ-Objekte gesteuert, die als *Pufferpools* bezeichnet werden.

Weitere Informationen zu Seitengruppen und Pufferpools finden Sie unter Speicherverwaltung.

## Protokollierung

Alle Änderungen an Objekten, die in Seitengruppen gespeichert sind, und alle an permanenten Nachrichten ausgeführten Operationen, werden als Protokolleinträge aufgezeichnet. Diese Protokolleinträge werden in eine Protokolldatei geschrieben, die als *aktives Protokoll* bezeichnet wird. Der Name und die Größe der aktiven Protokolldatei werden in einer Datei gespeichert, die als *Bootstrap-Data-Set* (BSDS) bezeichnet wird.

Wenn die aktive Protokolldatei ihre maximale Größe erreicht hat, wechselt der Warteschlangenmanager zu einer anderen Protokolldatei, damit die Protokollierung fortgesetzt werden kann, und kopiert den

Inhalt der vollen aktiven Protokolldatei in eine *Archivprotokolldatei*. Informationen zu diesen Aktionen, einschließlich des Namens der Archivprotokolldatei, werden im Bootstrap-Data-Set gespeichert. Man kann es sich so vorstellen, dass es einen Ring von aktiven Protokolldateien gibt, die der Warteschlangenmanager nacheinander verwendet. Wenn ein aktives Protokoll voll ist, werden die Protokolldaten in ein Archivprotokoll ausgelagert und die aktive Protokolldatei steht anschließend zur Wiederverwendung zur Verfügung.

Weitere Informationen zu den Protokolldateien und Bootstrap-Data-Sets finden Sie unter „Protokollierung in IBM MQ for z/OS" auf Seite 253.

## Anpassung der Warteschlangenmanager-Umgebung

Beim Start des Warteschlangenmanagers wird eine Gruppe von Initialisierungsparametern gelesen, mit denen der Betrieb des Warteschlangenmanagers gesteuert wird. Außerdem werden auch Dateien, die IBM MQ-Befehle enthalten, gelesen und die Befehle anschließend ausgeführt. Normalerweise enthalten diese Dateien Definitionen von Systemobjekten, die zum Ausführen von IBM MQ erforderlich sind. Sie können diese Definitionen anpassen, um damit die für Ihre Betriebsumgebung benötigten IBM MQ-Objekte zu definieren und zu initialisieren. Nachdem diese Dateien gelesen wurden, werden alle darin definierten Objekte entweder in einer Seitengruppe oder in Db2 gespeichert.

Weitere Informationen zu Initialisierungsparametern und Systemobjekten finden Sie unter „Systemdefinition unter z/OS" auf Seite 265.

## Neustart und Wiederherstellung

Während des Betriebs von IBM MQ kann es dazu kommen, dass Änderungen im Hauptspeicher vorhanden sind, die noch nicht in der Seitengruppe gespeichert wurden. Diese Änderungen werden in der Seitengruppe gespeichert, deren Verwendung durch die Hintergrundtask des Warteschlangenmanagers am längsten zurückliegt.

Wenn ein Warteschlangenmanager abnormal beendet wird, können die verlorenen Seitengruppenänderungen im Verlauf des Warteschlangenmanager-Neustarts wiederhergestellt werden, weil permanente Nachrichtendaten in Protokolleinträgen gespeichert werden. Dies bedeutet, dass IBM MQ alle permanenten Nachrichtendaten und Objektänderungen bis zum Fehlerpunkt wiederherstellen kann.

Wenn für einen Warteschlangenmanager in einer Gruppe mit gemeinsamer Warteschlange ein Coupling-Facility-Fehler auftritt, können die permanenten Nachrichten in dieser Warteschlange nur dann wiederhergestellt werden, wenn Sie die Coupling-Facility-Struktur zuvor gesichert haben.

Weitere Informationen zu Wiederherstellung und Neustart finden Sie unter „Wiederherstellung und Neustart unter z/OS" auf Seite 278.

## Sicherheit

Sie können eine externe Sicherheitsmanager-Software, z. B. Security Server (zuvor unter dem Namen RACF bekannt) verwenden, um die Ressourcen, deren Besitzer und Verwalter IBM MQ ist, vor dem Zugriff nicht berechtigter Benutzer zu schützen. Außerdem können Sie Transport Layer Security (TLS) für die Kanalsicherheit verwenden. TLS gehört zum Lieferumfang von IBM MQ.

Weitere Informationen zur IBM MQ-Sicherheit finden Sie im Abschnitt „Sicherheitskonzepte in IBM MQ for z/OS" auf Seite 297.

## Verfügbarkeit

IBM MQ verfügt über verschiedene Funktionen zur Erhöhung der Systemverfügbarkeit, falls einmal ein Fehler in Zusammenhang mit einem Warteschlangenmanager oder einem Kommunikationssubsystem auftritt. Weitere Informationen zu diesen Funktionen finden Sie unter „Verfügbarkeit unter z/OS" auf Seite 304.

### Bearbeitung von Objekten

Während der Warteschlangenmanager aktiv ist, können Sie IBM MQ-Objekte entweder über eine z/OS-Konsolenschnittstelle oder über ein Verwaltungsdienstprogramm bearbeiten, das ISPF-Services (Interactive System Productivity Facility) unter TSO (Time Sharing Option) verwendet. Mit beiden Mechanismen können Sie IBM MQ-Objekte definieren, ändern oder löschen. Außerdem können Sie den Status verschiedener IBM MQ- und Warteschlangenmanagerfunktionen anzeigen und steuern.

Weitere Informationen zu diesen Funktionen finden Sie unter Quellen, aus denen Sie MQSC-und PCF-Befehle unter IBM MQ for z/OSausgeben können.

Die Bearbeitung von IBM MQ-Objekten ist darüber hinaus auch mit IBM MQ Explorer möglich. Dabei handelt es sich um eine grafische Benutzerschnittstelle, die visuelle Möglichkeiten zur Bearbeitung von Warteschlangen, Warteschlangenmanagern und sonstigen Objekten bereitstellt.

### Überwachung und Statistik

Es stehen verschiedene Funktionen für die Überwachung der Warteschlangenmanager und Kanalinitiatoren zur Verfügung. Außerdem können Statistikdaten für die Leistungsbewertung und Abrechnung erfasst werden.

Weitere Informationen zu diesen Funktionen finden Sie unter „Überwachung und Statistik in IBM MQ for z/OS” auf Seite 308.

### Anwendungsumgebungen

Nachdem der Warteschlangenmanager gestartet wurde, können Anwendungen eine Verbindung zu ihm herstellen und die IBM MQ-API verwenden. Dabei kann es sich um CICS-, IMS-, Stapel- oder WebSphere Application Server-Anwendungen handeln. IBM MQ -Anwendungen können auch über die CICS -und IMS -Bridges auf Anwendungen auf CICS -und IMS -Systemen zugreifen, die IBM MQnicht kennen.

Weitere Informationen zu diesen Funktionen finden Sie unter „IBM MQ und andere z/OS-Produkte” auf Seite 313.

Informationen zum Schreiben von IBM MQ-Anwendungen finden Sie in folgenden Dokumentationen:

- Anwendungsentwicklung
- Verwendung von C++
- IBM MQ classes for Java verwenden

## z/OS Gemeinsam genutzte Warteschlangen und Gruppen mit gemeinsamer Warteschlange

Mithilfe von gemeinsam genutzten Warteschlangen und Gruppen mit gemeinsamer Warteschlange können Sie die Hochverfügbarkeit von IBM MQ-Ressourcen implementieren. Bei gemeinsam genutzten Warteschlangen und Gruppen mit gemeinsamer Warteschlange handelt es sich um Funktionen, die nur in IBM MQ for z/OS auf der Plattform z/OS verfügbar sind.

In diesem Abschnitt werden die Attribute und Vorteile beschrieben und es wird erläutert, wie mehrere Warteschlangenmanager dieselben Warteschlangen und die darin befindlichen Nachrichten gemeinsam nutzen können.

### Was ist eine gemeinsam genutzte Warteschlange?

Eine gemeinsam genutzte Warteschlange ist ein spezieller Typ von lokaler Warteschlange. Auf die Nachrichten in dieser Warteschlange können einer oder mehrere Warteschlangenmanager, die sich in einem Sysplex befinden, zugreifen.

#### Gruppe mit gemeinsamer Warteschlange

Die Warteschlangenmanager, die auf denselben Satz an gemeinsam genutzten Warteschlangen zugreifen können, bilden eine Gruppe, die als *Gruppe mit gemeinsamer Warteschlange* (Queue-Sharing Group, QSG) bezeichnet wird.

**Zugriff auf Nachrichten durch jeden Warteschlangenmanager**

Jeder Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange kann auf eine gemeinsam genutzte Warteschlange zugreifen. Dies bedeutet, dass es möglich ist, eine Nachricht in eine gemeinsame Warteschlange auf einem Warteschlangenmanager einzureihen und dieselbe Nachricht aus dieser Warteschlange von einem anderen Warteschlangenmanager zu empfangen. Dies ist ein Mechanismus für die schnelle Kommunikation innerhalb einer Gruppe mit gemeinsamer Warteschlange, für den keine Kanäle zwischen Warteschlangenmanagern aktiviert zu werden brauchen.

IBM MQ unterstützt die Auslagerung von Nachrichten in Db2 oder eine gemeinsam genutzte Nachrichtendatei (SMDS). Die Auslagerung von Nachrichten beliebiger Größe ist konfigurierbar.

Abbildung 58 auf Seite 183 zeigt drei Warteschlangenmanager und eine Coupling-Facility, die eine Gruppe mit gemeinsamer Warteschlange bilden. Alle drei Warteschlangenmanager haben Zugriff auf die gemeinsame Warteschlange in der Coupling-Facility.

Eine Anwendung kann zu jedem der drei Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange eine Verbindung herstellen. Da alle Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange auf alle gemeinsam genutzten Warteschlangen Zugriff haben, ist die Anwendung nicht auf die Verfügbarkeit eines bestimmten Warteschlangenmanagers angewiesen, denn jeder dieser Warteschlangenmanager kann die Warteschlange bedienen.

Dies führt zu einer höheren Verfügbarkeit, weil alle anderen Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange die Verarbeitung der Warteschlange fortsetzen können, wenn für einen der Warteschlangenmanager ein Fehler auftritt.

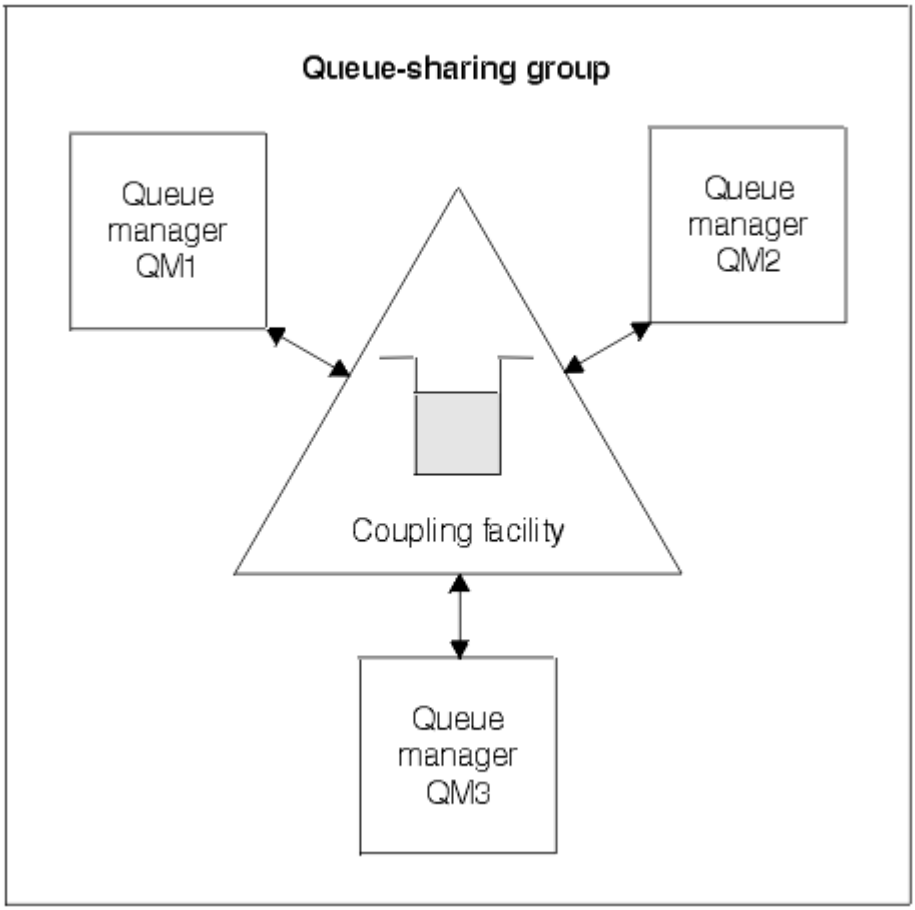



*Abbildung 58. Gruppe mit gemeinsamer Warteschlange*

**Von allen Warteschlangenmanagern gemeinsam genutzte Warteschlangendefinition**

Gemeinsam genutzte Warteschlangendefinitionen werden in der Db2-Datenbanktabelle OBJ_B_QUEUE gespeichert. Das hat den Vorteil, dass die Warteschlange nur einmal definiert werden muss und anschließend alle Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange Zugriff darauf haben. Das heißt, es müssen weniger Definitionen erstellt werden.

Im Gegensatz dazu wird die Definition einer nicht gemeinsam genutzten Warteschlange in der Seitengruppe 0 des Warteschlangenmanagers gespeichert, der der Besitzer der Warteschlange ist (wie unter Seitengruppen beschrieben).

Eine gemeinsam genutzte Warteschlange kann nicht unter demselben Namen definiert werden, der bereits für eine andere Warteschlange in den Seitengruppen desselben Warteschlangenmanagers verwendet wurde. Ebenso ist es nicht möglich, eine lokale Version einer Warteschlange in den Seitengruppen des Warteschlangenmanagers zu definieren, wenn bereits eine gemeinsam genutzte Warteschlange mit diesem Namen vorhanden ist.

## Was ist eine Gruppe mit gemeinsamer Warteschlange?

Eine Gruppe von Warteschlangenmanagern, die auf dieselben gemeinsam genutzten Warteschlangen zugreifen können, wird als Gruppe mit gemeinsamer Warteschlange bezeichnet. Jedes Mitglied der Gruppe mit gemeinsamer Warteschlange kann auf dieselbe Gruppe der gemeinsam genutzten Warteschlangen zugreifen.

Die Namen von Gruppen mit gemeinsamer Warteschlange können bis zu vier Zeichen lang sein. Der Name muss in Ihrem Netz eindeutig sein und muss sich von allen WS-Managernamen unterscheiden.

Abbildung 59 auf Seite 184 zeigt eine Gruppe mit gemeinsamer Warteschlange, die zwei Warteschlangenmanager enthält. Beide Warteschlangenmanager haben je einen Kanalinitiator und ihre eigenen lokalen Seitengruppen und Protokolldateien.

Jedes Mitglied der Gruppe mit gemeinsamer Warteschlange muss darüber hinaus mit einem Db2-System verbunden sein. Die Db2-Systeme müssen sich alle in derselben Db2-Gruppe mit gemeinsamer Datennutzung befinden, damit die Warteschlangenmanager auf das gemeinsam genutzte Db2-Repository zugreifen können, in dem die gemeinsam genutzten Objektdefinitionen gespeichert sind. Für jeden IBM MQ-Objekttyp (z. B. Warteschlangen und Kanäle) wird eine solche Definition nur einmal erstellt und anschließend kann jeder Warteschlangenmanager in der Gruppe darauf zugreifen und sie verwenden. Diese werden als *globale* Definitionen bezeichnet und im Abschnitt Private und globale Definitionen beschrieben.

Mehr als eine Gruppe mit gemeinsamer Warteschlange kann auf eine bestimmte Gruppe mit gemeinsamer Datennutzung verweisen. Sie geben den Namen des Db2 -Subsystems und die Gruppe mit gemeinsamer Datennutzung, die ein Warteschlangenmanager beim Start verwendet, in den IBM MQ -Systemparametern an.

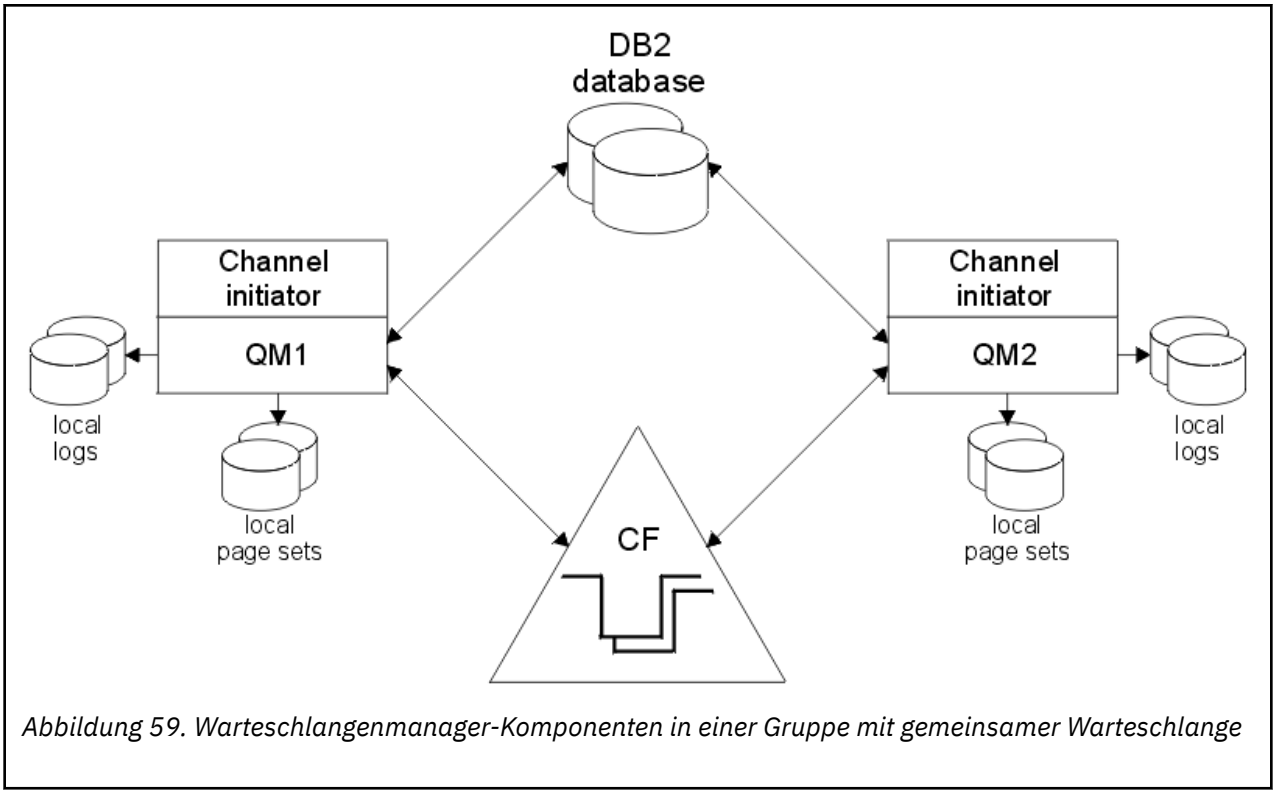



*Abbildung 59. Warteschlangenmanager-Komponenten in einer Gruppe mit gemeinsamer Warteschlange*

Nachdem ein Warteschlangenmanager in eine Gruppe mit gemeinsamer Warteschlange aufgenommen wurde, hat er Zugriff auf die für diese Gruppe definierten gemeinsam genutzten Objekte, und Sie können mit diesem Warteschlangenmanager auch neue gemeinsam genutzte Objekte in der Gruppe definieren. Wenn gemeinsam genutzte Warteschlangen in der Gruppe definiert sind, können Sie mit diesem Warteschlangenmanager Nachrichten in diese Warteschlangen einreihen und daraus abrufen. Jeder Warteschlangenmanager in der Gruppe kann die in einer gemeinsam genutzten Warteschlange gespeicherten Nachrichten abrufen.

Wenn Sie einen WebSphere MQ-Scriptbefehl auf einem Warteschlangenmanager eingeben, kann er auf allen Warteschlangenmanagern innerhalb der Gruppe mit gemeinsamer Warteschlange so ausgeführt werden, als wäre er auf jedem Warteschlangenmanager einzeln eingegeben worden. Dafür wird das Attribut für den *Befehlsbereich* verwendet. Dieses Attribut wird im Abschnitt Befehle an verschiedene Warteschlangenmanager leiten beschrieben.

Wenn ein Warteschlangenmanager als Mitglied einer Gruppe mit gemeinsamer Warteschlange ausgeführt wird, muss eine Unterscheidung möglich sein zwischen IBM MQ-Objekten, die für diesen Warteschlangenmanager privat definiert wurden, und IBM MQ-Objekten, die global definiert wurden und allen Warteschlangenmanagern in der Gruppe mit gemeinsamer Warteschlange zur Verfügung stehen. Dafür wird das

Attribut für die *Disposition der Gruppe mit gemeinsamer Warteschlange* verwendet. Dieses Attribut wird im Abschnitt Private und globale Definitionen beschrieben.

Sie können einen einzelnen Satz von Sicherheitsprofilen definieren, die den Zugriff auf IBM MQ-Objekte in der gesamten Gruppe steuern. Diese bedeutet, dass Sie eine weitaus geringere Anzahl von Profilen definieren müssen.

Ein Warteschlangenmanager kann nur zu einer Gruppe mit gemeinsamer Warteschlange gehören und alle Warteschlangenmanager in der Gruppe müssen sich in demselben Sysplex befinden. Zu welcher Gruppe mit gemeinsamer Warteschlange ein Warteschlangenmanager gehört, können Sie beim Systemstart in den Systemparametern angeben.

**Zugehörige Konzepte**

„Wo werden gemeinsam genutzte Warteschlangen gespeichert?" auf Seite 185
Jede Nachricht in einer gemeinsam genutzten Warteschlange wird durch einen Eintrag in einer z/OS-Coupling-Facility-Listenstruktur dargestellt. Wenn die Nachrichtendaten zu umfangreich sind, um in denselben Eintrag zu passen, werden sie entweder in eine gemeinsam genutzte Nachrichtendatei (Shared Message Data Set, SMDS) oder in Db2 ausgelagert.

„Vorteile von gemeinsam genutzten Warteschlangen" auf Seite 204
Mithilfe gemeinsam genutzter Warteschlangen werden IBM MQ-Anwendungen skalierbar und hoch verfügbar. Darüber hinaus wird die Implementierung eines Lastausgleichs ermöglicht.

„Verteilte Steuerung von Warteschlangen und Gruppen mit gemeinsamer Warteschlange" auf Seite 226
Die verteilte Steuerung von Warteschlangen und Gruppen mit gemeinsamer Warteschlange sind zwei Verfahren, mit denen Sie die Verfügbarkeit von Anwendungssystemen erhöhen können. In diesem Abschnitt werden diese beiden Verfahren ausführlicher erläutert.

„Lastverteilung mit gemeinsam genutzten Warteschlangen beeinflussen" auf Seite 229
In diesem Abschnitt werden die Faktoren beschrieben, die die Lastverteilung mit gemeinsam genutzten Warteschlangen in einer Gruppe mit gemeinsamer Warteschlange beeinflussen.

**Zugehörige Verweise**

„Weitere Informationen zu gemeinsam genutzten Warteschlangen und Gruppen mit gemeinsamer Warteschlange" auf Seite 231
Mit Hilfe der in diesem Abschnitt beschriebenen Tabelle können Sie weitere Informationen darüber erhalten, wie IBM MQ for z/OS gemeinsam genutzte Warteschlangen und Gruppen mit gemeinsamer Warteschlange verwendet.

## z/OS Wo werden gemeinsam genutzte Warteschlangen gespeichert?

Jede Nachricht in einer gemeinsam genutzten Warteschlange wird durch einen Eintrag in einer z/OS-Coupling-Facility-Listenstruktur dargestellt. Wenn die Nachrichtendaten zu umfangreich sind, um in denselben Eintrag zu passen, werden sie entweder in eine gemeinsam genutzte Nachrichtendatei (Shared Message Data Set, SMDS) oder in Db2 ausgelagert.

Wenn die CF-Struktur für die Verwendung von System Class Memory (SCM) konfiguriert wurde, kann diese Struktur in IBM MQ verwendet werden, ohne dass eine zusätzliche Konfiguration erforderlich ist.

**Wichtig:** IBM z16 ist als letzte Generation von IBM Z ® geplant, um die Verwendung von virtuellem Flashspeicher (auch bekannt als Storage Class Memory oder SCM) für Coupling-Facility-Images zu unterstützen. Weitere Informationen finden Sie unter IBM Z und IBM LinuxONE 4Q 2023 Statements of Direction.

Alternativ sollten Sie entweder größere Strukturen verwenden oder Nachrichten in SMDS auslagern.

### Speicherung von Nachrichten in gemeinsam genutzten Warteschlangen

Nachrichten, die in gemeinsam genutzte Warteschlangen eingereiht werden, werden nicht in Seitengruppen gespeichert und verwenden keine Pufferpools.

Für Nachrichten in gemeinsam genutzten Warteschlangen werden Einträge in Listenstrukturen in der z/OS-Coupling-Facility (CF) erstellt. Viele Warteschlangenmanager in demselben Sysplex können über die CF-Listenstruktur auf diese Nachrichten zugreifen.

Die Nachrichtendaten für kleine Nachrichten in einer gemeinsam genutzten Warteschlange sind normalerweise im Coupling-Facility-Eintrag enthalten. Für größere Nachrichten können die Nachrichtendaten entweder in einer gemeinsam genutzten Nachrichtendatei (Shared Message Data Set, SMDS) oder in Form eines oder mehrerer großer Binärobjekte (Binary Large Objects, BLOBs) in einer Db2-Tabelle, die von Db2-Gruppe mit gemeinsamer Datennutzung verwendet wird, gespeichert werden. Nachrichtendaten von mehr als 63 KB werden immer in eine SMDS oder in Db2 ausgelagert. Kleinere Nachrichten können optional auf dieselbe Weise ausgelagert werden, um Speicherplatz in der Coupling-Facility-Struktur zu sparen. Weitere Informationen finden Sie unter „Auslagerungsoptionen für gemeinsam genutzte Nachrichten angeben” auf Seite 188.

Nachrichten, die in eine gemeinsam genutzte Warteschlange eingereiht wurden, werden so lange in einer Coupling-Facility-Struktur referenziert, bis sie von einer MQGET-Operation abgerufen wurden. Coupling-Facility-Operationen ermöglichen folgende Aktionen:

- Suche nach der nächsten abrufbaren Nachricht
- Sperre nicht festgeschriebener Nachrichten in gemeinsam genutzten Warteschlangen
- Benachrichtigung interessierter Warteschlangenmanager über den Eingang festgeschriebener Nachrichten

Die für permanente Nachrichten ausgeführten MQPUT- und MQGET-Operationen werden im Protokoll des ausführenden Warteschlangenmanagers erfasst. Dadurch wird die Gefahr eines Datenverlusts im Falle eines Fehlers in der Coupling-Facility minimiert.

## Coupling-Facility

Die Nachrichten, die in einer gemeinsam genutzten Warteschlange gespeichert sind, werden in einer Coupling-Facility referenziert. Die Coupling-Facility befindet sich außerhalb aller z/OS-Images im Sysplex und wird normalerweise für den Betrieb mit einer separaten Stromversorgung konfiguriert. Die Coupling-Facility ist deshalb ausfallsicher gegenüber Softwarefehlern und kann darüber hinaus so konfiguriert werden, dass sie auch ausfallsicher gegenüber Hardwarefehlern und Stromausfällen ist. Dies bedeutet, dass in der Coupling-Facility gespeicherte Nachrichten hoch verfügbar sind.

Jede von IBM MQ verwendete Coupling-Facility-Listenstruktur ist einer bestimmten Gruppe mit gemeinsamer Warteschlange zugeordnet, kann aber Strukturen für mehrere Gruppen mit gemeinsamer Warteschlange enthalten. Die gemeinsame Nutzung von Daten ist für Warteschlangenmanager, die zu unterschiedlichen Gruppen mit gemeinsamer Warteschlange gehören, nicht möglich. Bis zu 32 Warteschlangenmanager in einer Gruppe mit gemeinsamer Warteschlange können gleichzeitig mit einer Coupling-Facility-Listenstruktur verbunden sein.

Eine einzelne Coupling-Facility-Listenstruktur kann bis zu 512 gemeinsam genutzte Warteschlangen enthalten. Das insgesamt in der Struktur speicherbare Nachrichtendatenvolumen wird durch die Kapazität der Struktur begrenzt. Mit **CFLEVEL(5)** können Sie jedoch die Auslagerungsparameter verwenden, um Daten für Nachrichten unter 63 KB auszulagern, wodurch die Anzahl der Nachrichten, die in der Struktur gespeichert werden können, erhöht wird, obwohl jede Nachricht immer noch mindestens einen Coupling-Facility-Eintrag sowie mindestens 768 Byte an Daten erfordert, die aus 256 Byte für den Eintrag und 512 Byte für die beiden Header-und Deskriptorelemente bestehen.

Die Größe der Listenstruktur wird durch folgende Faktoren begrenzt:

- Sie muss sich innerhalb einer einzelnen Coupling-Facility befinden.
- Sie kann den verfügbaren Coupling-Facility-Speicher gemeinsam mit anderen Strukturen für IBM MQ und andere Produkte nutzen.

Coupling-Facility-Listenstrukturen kann Storage Class Memory (SCM) zugeordnet sein. SCM kann in bestimmten Situationen hilfreich sein, wenn er in Verbindung mit gemeinsam genutzten Warteschlangen

verwendet wird. Weitere Informationen finden Sie unter „Storage Class Memory mit gemeinsam genutzten Warteschlangen verwenden” auf Seite 206.

## CF-Strukturgröße planen

Eine Anleitung zur Dimensionierung der CF-Strukturen finden Sie im SupportPac MP16: IBM MQ for z/OS Capacity planning and tuning. Außerdem können Sie das webbasierte Tool CFSizer verwenden, das von IBM zur Unterstützung der Coupling-Facility-Dimensionierung bereitgestellt wird.

## CF-Strukturobjekte

Die Verwendung einer Coupling-Facility-Struktur durch einen Warteschlangenmanager wird in einem CF-Strukturobjekt (CFSTRUCT) für IBM MQ angegeben.

Diese Strukturobjekte werden in Db2 gespeichert.

Bei der Verwendung von z/OS-Befehlen oder -Definitionen für eine Coupling-Facility-Struktur sind die ersten vier Zeichen des Namens der Gruppe mit gemeinsamer Warteschlange erforderlich. Da ein CFSTRUCT-Objekt für IBM MQ jedoch immer nur zu einer einzigen Gruppe mit gemeinsamer Warteschlange gehört, enthält sein Name nicht die ersten vier Zeichen des Gruppennamens. Beispielsweise verwendet das Objekt CFSTRUCT(MYDATA), das in einer Gruppe mit gemeinsamer Warteschlange definiert ist, deren Name mit SQ03 beginnt, die Coupling-Facility-Listenstruktur SQ03MYDATA.

CF-Strukturen ist das Attribut CFLEVEL zugeordnet, das deren Funktionalität bestimmt:

- 1, 2 - kann für nicht permanente Nachrichten verwendet werden, die kleiner als 63 KB sind
- 3 - kann für permanente und nicht permanente Nachrichten verwendet werden, die kleiner als 63 KB sind
- 4 - kann für permanente und nicht permanente Nachrichten bis zu einer Größe von 100 MB verwendet werden
- 5 - kann für permanente und nicht permanente Nachrichten bis zu einer Größe von 100 MB verwendet werden, die selektiv in gemeinsam genutzte Nachrichtendateien (SMDS) oder in Db2 ausgelagert wurden

**Anmerkung:** Bei Verwendung von IBM MQ können Sie eine Coupling-Facility-Struktur verschlüsseln. Weitere Informationen finden Sie unter Daten der Coupling-Facility-Struktur verschlüsseln .

## Sicherung und Wiederherstellung der Coupling-Facility

Eine Coupling-Facility kann mit dem IBM MQ-Befehl BACKUP CFSTRUCT gesichert werden. Dabei wird eine Kopie der permanenten Nachrichten, die sich aktuell in der CF-Struktur befinden, in der aktiven Protokolldatei des Warteschlangenmanagers gespeichert, der die Sicherung ausführt, und ein Datensatz für die Sicherung in Db2 geschrieben.

Wenn die Coupling-Facility fehlschlägt, können Sie den IBM MQ-Befehl RECOVER CFSTRUCT verwenden. Dabei werden anhand des Sicherungsdatensatzes in Db2 die permanenten Nachrichten in der Sicherung der CF-Struktur gesucht und wiederhergestellt. Alle seit der letzten Sicherung ausgeführten Aktivitäten werden mithilfe der Protokolle aller Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange wiederholt und anschließend wird die CF-Struktur in dem Zustand vor Eintritt des Fehlers wiederhergestellt.

Weitere Informationen finden Sie in den Abschnitten zu den Befehlen BACKUP CFSTRUCT und RECOVER CFSTRUCT.

**Zugehörige Konzepte**

„Auslagerungsoptionen für gemeinsam genutzte Nachrichten angeben” auf Seite 188
Sie können entscheiden, wo die Nachrichtendaten für eine Nachricht in einer gemeinsam genutzten Warteschlange gespeichert werden sollen, entweder in einer Db2-Tabelle oder in einer gemeinsam genutzten Nachrichtendatei (Shared Message Data Set, SMDS). Sie können außerdem auswählen, welche

Nachrichten ausgelagert werden sollen, und zwar basierend auf der Nachrichtengröße und der aktuellen Verwendung der Coupling-Facility-Struktur (CF).

„Umgebung mit gemeinsam genutzten Nachrichtendateien verwalten” auf Seite 190
Wenn Sie gemeinsam genutzte Nachrichtendateien zum Auslagern langer Nachrichten auswählen, sollte Ihnen bekannt sein, welche Informationen IBM MQ zum Verwalten dieser Nachrichtendateien nutzt und welche Befehle zur Verarbeitung dieser Informationen verwendet werden. In diesem Abschnitt wird erläutert, wie gemeinsam genutzte Nachrichtendateien verwaltet werden.

### z/OS *Auslagerungsoptionen für gemeinsam genutzte Nachrichten angeben*

Sie können entscheiden, wo die Nachrichtendaten für eine Nachricht in einer gemeinsam genutzten Warteschlange gespeichert werden sollen, entweder in einer Db2-Tabelle oder in einer gemeinsam genutzten Nachrichtendatei (Shared Message Data Set, SMDS). Sie können außerdem auswählen, welche Nachrichten ausgelagert werden sollen, und zwar basierend auf der Nachrichtengröße und der aktuellen Verwendung der Coupling-Facility-Struktur (CF).

Die Nachrichtendaten für gemeinsam genutzte Warteschlangen können aus der Coupling-Facility ausgelagert und entweder in einer Db2-Tabelle oder in einer von IBM MQ verwalteten Datei, die als *gemeinsam genutzte Nachrichtendatei* (Shared Message Data Set, SMDS) bezeichnet wird, gespeichert werden.

Bei Nachrichten, die größer als die maximale Coupling-Facility-Eintragsgröße von 63 KB sind, kann die Auslagerung von Nachrichtendaten in eine SMDS eine erhebliche Leistungsverbesserung gegenüber der Auslagerung in Db2 bedeuten.

Auch in diesem Fall wird jede Nachricht in einer gemeinsam genutzten Warteschlange noch mithilfe eines Listeneintrags in einer Coupling-Facility-Struktur verwaltet, doch wenn die Nachrichtendaten in eine SMDS ausgelagert werden, enthält der Coupling-Facility-Eintrag nur noch einige Steuerinformationen und eine Liste mit Verweisen auf die relevanten Plattenblöcke, in denen die Nachricht gespeichert ist. Bei Verwendung dieses Mechanismus entspricht der Speicherplatz, der zum Speichern einer jeden Nachricht in einem Coupling-Facility-Element erforderlich ist, nur ein Bruchteil der tatsächlichen Nachrichtengröße.

**Speicherposition für Nachrichten in gemeinsam genutzten Warteschlangen auswählen**

Die Auswahl des gemeinsam genutzten SMDS- oder Db2 -Nachrichtenspeichers wird über den Parameter **OFFLOAD(SMDS|DB2)** in der **CFSTRUCT**-Definition gesteuert. **OFFLOAD(SMDS)** ist der Standardwert.

Dieser Parameter erfordert auch, dass **CFSTRUCT CFLEVEL(5)** oder höher verwendet.

Der Parameter **OFFLOAD** ist nur ab **CFLEVEL(5)** gültig. Weitere Informationen finden Sie im Abschnitt zum Befehl DEFINE CFSTRUCT.

**OFFLOAD(DB2)** wird hauptsächlich für Migrationszwecke unterstützt.

**Auswahl der Nachrichten in gemeinsam genutzten Warteschlangen für die Auslagerung treffen**

Nachrichtendaten werden abhängig von der Nachrichtengröße und der aktuellen Verwendung der Coupling-Facility-Struktur in eine gemeinsam genutzte Nachrichtendatei (SMDS) oder in Db2 ausgelagert. Es gibt drei Regeln, von denen jede ein zusammengehöriges Paar von Parametern angibt. Diese Parameter sind ein entsprechender Prozentsatz des Schwellenwerts für die Nutzung der Coupling-Facility-Struktur ( **OFFLDnTH** ). und eine Nachrichtengrößenbegrenzung ( **OFFLDnSZ** ).

Die aktuelle Implementierung dieser drei Regeln wird mithilfe folgender Schlüsselwortpaare angegeben:

- OFFLD1TH und OFFLD1SZ
- OFFLD2TH und OFFLD2SZ
- OFFLD3TH und OFFLD3SZ

| Regelpaar | Standardwert | Beschreibung |
|---|---|---|
| Regelpaar 1 | OFFLD1TH(70) und OFFLD1SZ(32K) | Wenn die Coupling-Facility-Struktur zu mehr als 70 % voll ist, werden Daten für Nachrichten mit einer Größe über 32 KB ausgelagert. |
| Regelpaar 2 | OFFLD2TH(80) und OFFLD2SZ(4K) | Wenn die Coupling-Facility-Struktur zu mehr als 80 % voll ist, werden Daten für Nachrichten mit einer Größe über 4 KB ausgelagert. |
| Regelpaar 3 | OFFLD3TH(90) und OFFLD3SZ(0K) | Wenn die Coupling-Facility-Struktur zu mehr als 90 % voll ist, werden Daten für Nachrichten mit einer Größe über 0 KB (alle Nachrichten) ausgelagert. |

Wenn für eine Auslagerungsregel für 'OFFLD *x* SZ' der Wert 64K festgelegt ist, ist die Regel nicht wirksam. In diesem Fall werden Nachrichten nur ausgelagert, wenn eine andere Auslagerungsregel wirksam ist oder wenn die Nachricht größer als 63,75 KB ist und damit zu groß, um in der Struktur gespeichert zu werden.

Für jede ausgelagerte Nachricht wird immer noch ein Speicherplatz von 0,75 KB in der Coupling-Facility benötigt.

Die drei Auslagerungsregeln, die für jede Struktur angegeben werden können, sollten wie folgt angewendet werden:

- Leistung
  - Wenn sehr viel Speicherbereich in der Anwendungsstruktur verfügbar ist, sollten Nachrichtendaten nur ausgelagert werden, wenn sie zu groß sind, um in der Struktur gespeichert zu werden, oder wenn sie einen unteren Schwellenwert für die Nachrichtengröße überschreiten, sodass der durch die Speicherung in der Struktur erreichte Leistungsvorteil die Größe des Strukturspeicherbereichs, der von den Daten belegt wird, nicht rechtfertigt.
  - Wenn ein bestimmter Schwellenwert für die Nachrichtengröße erforderlich ist, wird er üblicherweise mit der ersten Auslagerungsregel angegeben.
- Kapazität
  - Wenn in der Anwendungsstruktur sehr wenig Speicherbereich verfügbar ist, sollte der Großteil der Nachrichtendaten ausgelagert werden, damit der verbleibende Speicherbereich möglichst optimal genutzt werden kann.
  - Mit der dritten Auslagerungsregel wird üblicherweise angegeben, dass die meisten Nachrichten ausgelagert werden sollen, wenn die Struktur fast voll ist. Die Einträge in der Anwendungsstruktur haben dann normalerweise die Mindestgröße (etwa 0,75 KB sind erforderlich).
  - Der Belegungsschwellenwertparameter sollte abhängig von der Größe der Anwendungsstruktur und dem erwarteten maximalen Rückstand festgelegt werden. Wenn der erwartete maximale Rückstand beispielsweise 1000 Nachrichten umfasst, beträgt die für diese Anzahl Nachrichten erforderliche Größe des Strukturspeicherbereichs circa 0,75 GB. Bei einer Strukturgröße von etwa 10 GB bedeutet dies zum Beispiel, dass der Belegungsschwellenwert für die Auslagerung aller Nachrichten auf 92% oder weniger eingestellt werden muss.
  - Der Strukturspeicherbereich ist in Elemente und Einträge aufgeteilt, und auch wenn insgesamt genug Speicherbereich vorhanden ist, kann es sein, dass einer der Teilbereiche früher belegt ist als die anderen. Das System stellt eine AUTOALTER-Funktion bereit, um das Verhältnis bei Bedarf anzupassen; da diese aber nicht sehr empfindlich reagiert, kann der tatsächlich verfügbare Speicherbereich etwas klein sein. Deshalb ist es möglicherweise besser, nicht mehr als 90 Prozent

des maximalen Strukturspeicherbereichs zu verwenden. Im oben genannten Beispiel sollte der Belegungsschwellenwert für die Auslagerung aller Nachrichten daher besser auf etwa 80 Prozent eingestellt werden.

- Temporärer dynamischer Reservespeicher:
  - Da die Größe des verbleibenden Speicherbereichs in der Coupling-Facility-Struktur abnimmt, wäre es nicht wünschenswert, wenn es zu einer plötzlichen großen Änderung bei den Leistungsmerkmalen käme. Es wäre auch nicht wünschenswert für die Coupling-Facility-Verwaltung, wenn eine plötzliche Schwellenwertänderung im typischen Verhältnis von Einträgen zu Elementen vorgenommen würde.
  - Mit der zweiten Auslagerungsregel wird üblicherweise ein temporärer dynamischer Reservespeicher zwischen den leistungs- und kapazitätsbezogenen Auslagerungsregeln bereitgestellt. Er kann eingerichtet werden, um eine deutliche Zunahme der Auslagerungsaktivität zu bewirken, wenn der in der Coupling-Facility-Struktur belegte Speicherbereich einen temporären Schwellenwert überschreitet. Dies bedeutet, dass der verbleibende Speicherbereich langsamer belegt wird, was der automatischen Änderungsverarbeitung der Coupling-Facility mehr Zeit zur Anpassung an den höheren Belegungsgrad gibt.

Wenn die Coupling-Facility-Struktur nicht erweitert werden kann, jedoch unbedingt eine vordefinierte Anzahl von Nachrichten gespeichert werden muss, kann die dritte Regel nach Bedarf so angepasst werden, dass die Auslagerung von Daten für alle Nachrichten ab einem geeigneten Schwellenwert beginnt, um sicherzustellen, dass der für die vordefinierte Anzahl von Nachrichten erforderliche Speicherplatz reserviert wird.

Wenn die Coupling-Facility-Struktur beispielsweise eine Größe von 4 GB hat und die vordefinierte Anzahl von Nachrichten 1 Million beträgt, dann wird 1.000.000 * 0,75 KB an Speicherplatz benötigt, das entspricht 768 MB, 18,75 % von 4 GB. In diesem Fall muss der Schwellenwert für die Auslagerung aller Nachrichten eher auf ca. 80 % als auf 90 % festgelegt werden. Als Parameter müssen in diesem Fall also 'OFFLD3TH(80)' und 'OFFLD3SZ(0K)' angegeben werden. Die anderen Auslagerungsparameter müssen ebenfalls angepasst werden.

Wenn festgestellt wird, dass die Auslagerung sehr kleiner Nachrichten erheblichen Einfluss auf die Leistung hat, während der relative Einfluss bei größeren Nachrichten geringer ist, dann können die Verwendungsschwellenwerte für die anderen Regeln gesenkt werden, damit größere Nachrichten früher ausgelagert werden und mehr Speicherplatz in der Struktur für kleine Nachrichten bleibt, bevor diese ausgelagert werden müssen.

Wenn beispielsweise häufig Nachrichten mit einer Größe über 32 KB auftreten, doch der Zeitaufwand für ihre Auslagerung (ermittelt mithilfe der RMF-Statistik (Resource Measurement Facility) oder der Anwendungsleistung) dem Zeitaufwand für ihre Beibehaltung in der Coupling-Facility sehr ähnlich ist, dann kann der Schwellenwert für die erste Regel auf 0 % gesetzt werden, um all diese Nachrichten auszulagern. Als Parameter müssen in diesem Fall 'OFFLD1TH(0)' und 'OFFLD1SZ(32K)' angegeben werden. Die anderen Auslagerungsparameter müssen auch in diesem Fall ebenfalls angepasst werden.

Wenn es viele Nachrichten im Bereich bestimmter mittlerer Größen, z. B. 16 KB und 6 KB, gibt, dann kann es nützlich sein, die Nachrichtengrößenoption für die zweite Regel zu ändern, damit große Nachrichten bei einem relativ niedrigen Verwendungsschwellenwert ausgelagert werden und eine erhebliche Menge an Speicherplatz gespart wird, während die kleinen Nachrichten trotzdem nur in der Coupling-Facility gespeichert werden.

### z/OS *Umgebung mit gemeinsam genutzten Nachrichtendateien verwalten*

Wenn Sie gemeinsam genutzte Nachrichtendateien zum Auslagern langer Nachrichten auswählen, sollte Ihnen bekannt sein, welche Informationen IBM MQ zum Verwalten dieser Nachrichtendateien nutzt und welche Befehle zur Verarbeitung dieser Informationen verwendet werden. In diesem Abschnitt wird erläutert, wie gemeinsam genutzte Nachrichtendateien verwaltet werden.

## SMDS-Objekte

Die Eigenschaften und der Status einer jeden gemeinsam genutzten Nachrichtendatei (Shared Message Data Sets, SMDS) werden in einem gemeinsam genutzten SMDS-Objekt protokolliert, das über jeden Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange aktualisiert werden kann.

Es gibt für jeden Warteschlangenmanager, der auf die jeweilige Coupling-Facility-Anwendungsstruktur zugreifen kann, eine gemeinsam genutzte Nachrichtendatei. Die gemeinsam genutzte Nachrichtendatei wird identifiziert durch den Namen des Warteschlangenmanagers, der Eigner der Datei ist, der mit dem Schlüsselwort SMDS angegeben wird, und durch den Namen der Anwendungsstruktur, der mit dem Schlüsselwort 'CFSTRUCT' angegeben wird.

**Anmerkung:** Wenn Sie SMDS-Dateien für eine Struktur definieren, muss es für jeden Warteschlangenmanager eine Datei geben.

Das SMDS-Objekt wird in einer Feldgruppe (mit einem Eintrag pro Warteschlangenmanager in der Gruppe) gespeichert, die eine Erweiterung des entsprechenden, in Db2 gespeicherten CFSTRUCT-Objekts darstellt.

Es gibt keinen Befehl zum Definieren oder Löschen des SMDS-Objekts, weil es als Teil des CFSTRUCT-Objekts erstellt bzw. gelöscht wird. Es gibt jedoch den Befehl ALTER, mit dem die Einstellungen für einen einzelnen zugehörigen Warteschlangenmanager geändert werden können.

Weitere Informationen zu SMDS-Befehlen finden Sie unter „SMDS-bezogene Befehle” auf Seite 203

## SMDSCONN-Informationen

Selbst wenn eine gemeinsam genutzte Nachrichtendatei einen normalen Status hat, ist es möglich, dass einer oder mehrere Warteschlangenmanager keine Verbindung zu ihr herstellen können, beispielsweise weil es ein Problem mit einer Sicherheitsdefinition oder mit der Einheitenkonnektivität bei Direktzugriff gibt. Deshalb muss jeder Warteschlangenmanager den Verbindungsstatus und die Verfügbarkeitsinformationen für jede gemeinsam genutzte Nachrichtendatei überwachen, denn dort ist beispielsweise angegeben, ob aktuell eine Verbindung zur jeweiligen Nachrichtendatei hergestellt werden kann, und wenn dies nicht der Fall ist, welche Ursache das hat.

Die SMDSCONN-Informationen beziehen sich auf die Verbindung eines Warteschlangenmanagers zu einer gemeinsam genutzten Nachrichtendatei. Zur Identifizierung der gemeinsam genutzten Nachrichtendatei selbst wird der Name des Warteschlangenmanagers, der der Besitzer der gemeinsam genutzten Nachrichtendatei ist (wie im Schlüsselwort SMDS für das gemeinsam genutzte Objekt selbst angegeben), kombiniert mit dem CFSTRUCT-Namen benötigt.

Zur Angabe des Warteschlangenmanagers, der die Verbindung herstellt, ist kein besonderer Parameter erforderlich, weil die an einen bestimmten Warteschlangenmanager gerichteten Befehle nur auf die SMDSCONN-Informationen dieses einen Warteschlangenmanagers zugreifen können.

Die Einträge mit den SMDSCONN-Informationen werden im Hauptspeicher des zugehörigen Warteschlangenmanagers verwaltet und bei einem Neustart des Warteschlangenmanagers erneut erstellt. Falls eine Verbindung eines einzelnen Warteschlangenmanagers jedoch explizit gestoppt wurde, wird diese Information auch als Flag in einer Verbindungsfeldgruppe des entsprechenden CFSTRUCT- oder SMDS-Objekts gespeichert, damit diese Information auch nach dem Neustart des Warteschlangenmanagers bestehen bleibt.

## Status- und Verfügbarkeitsinformationen

Die Statusinformationen geben den Zustand einer Ressource oder Verbindung an (z. B. ob sie noch nicht verwendet wird, ob sie in normaler Verwendung ist oder ob eine Wiederherstellung für sie erforderlich ist). Dies wird normalerweise mit dem Schlüsselwort STATUS beschrieben. Die gültigen Werte hängen vom Objekttyp ab.

Normalerweise werden die Statusinformationen automatisch aktualisiert, beispielsweise wenn beim Verwenden der Ressource oder Verbindung ein Fehler festgestellt wird. In manchen Fällen kann der Status

jedoch auch durch einen Befehl aktualisiert werden. Dies ist beispielsweise nützlich, wenn es für einen Warteschlangenmanager nicht möglich ist, den korrekten Status automatisch zu ermitteln.

Die Verfügbarkeitsinformationen geben an, ob eine Ressource oder Verbindung verwendet werden kann, was normalerweise hauptsächlich anhand der Statusinformationen ermittelt wird. Für die Ressourcen- und Verbindungstypen, die bei der Unterstützung von gemeinsam genutzten Nachrichtendateien verwendet werden, sind drei Verfügbarkeitsstufen implementiert:

**Verfügbar**
Dies bedeutet, dass die Ressource für die normale Verwendung verfügbar ist. Es bedeutet jedoch nicht unbedingt, dass sie derzeit in Verwendung ist (dies lässt sich stattdessen anhand des Werts des Schlüsselworts STATUS ermitteln). Bei einer Nachrichtendatei, die neu gestartet werden muss, ermöglicht dieser Status dem zugehörigen Warteschlangenmanager, der der Besitzer ist, die Nachrichtendatei zu öffnen, während andere Warteschlangenmanager warten müssen, bis die Nachrichtendatei wieder den Status ACTIVE hat.

**Nicht verfügbar wegen eines Fehlers**
Dies bedeutet, dass die Ressource wegen eines Fehlers automatisch gesperrt wird und auch erst wieder verfügbar gemacht wird, nachdem irgendeine Form der Reparatur oder Wiederherstellung ausgeführt wurde. Jedoch werden Versuche, sie ohne Bedienereingriff wieder verfügbar zu machen, zugelassen. Ein solcher Versuch kann auch durch verschiedene Befehle ausgelöst werden, z. B. durch den Befehl, die Ressource als aktiviert zu markieren, oder den Befehl, den Ressourcenstatus so zu ändern, dass er den Abschluss der Wiederherstellung anzeigt.

Der Grund, aus dem die Ressource gesperrt wurde, lässt sich normalerweise aus dem zugehörigen STATUS-Wert ablesen, doch wenn andere Gründe dafür verantwortlich sind, wird im separaten Schlüsselwort REASON die jeweilige Ursache angegeben.

**Nicht verfügbar wegen eines Bedienerbefehls**
Dies bedeutet, dass der Zugriff auf die Ressource durch einen Befehl explizit inaktiviert wurde. Sie kann in diesem Fall nur durch einen Befehl zur erneuten Aktivierung wieder verfügbar gemacht werden.

**SMDS-Verfügbarkeit**

Die Verfügbarkeit des SMDS-Objekts wird durch das Schlüsselwort 'ACCESS' angegeben, wobei die gültigen Werte ENABLED (Aktiviert), SUSPENDED (Ausgesetzt) und DISABLED (Inaktiviert) lauten.

Die Verfügbarkeit kann mit einem **RESET SMDS** -Befehl für das relevante gemeinsam genutzte Objekt von jedem Warteschlangenmanager in der Gruppe aktualisiert werden, um ACCESS (ENABLED) oder ACCESS (DISABLED) festzulegen.

Wenn der Verfügbarkeitswert zuvor ACCESS(SUSPENDED) war, wird durch eine Änderung auf den Wert ACCESS(ENABLED) ein neuer Versuch zum Verwenden der gemeinsam genutzten Nachrichtendatei ausgelöst, doch wenn der vorherige Fehler weiterhin vorliegt, wird die Verfügbarkeit auf den Wert ACCESS(SUSPENDED) zurückgesetzt.

**SMDSCONN-Verfügbarkeit**

Die Verfügbarkeit eines lokalen SMDSCONN-Informationseintrags wird durch das Schlüsselwort AVAIL angegeben, wobei die gültigen Werte NORMAL, ERROR (Fehler) und STOPPED (Gestoppt) lauten. Die Verfügbarkeit kann mit einem **START SMDSCONN** -oder **STOP SMDSCONN** -Befehl aktualisiert werden, der an einen bestimmten Warteschlangenmanager adressiert ist, um seine Verbindung zu aktivieren oder zu inaktivieren.

Wenn der Verfügbarkeitswert zuvor AVAIL(ERROR) war, wird durch eine Änderung auf den Wert AVAIL(NORMAL) ein neuer Versuch zum Verwenden der gemeinsam genutzten Nachrichtendatei ausgelöst, doch wenn der vorherige Fehler weiterhin vorliegt, wird die Verfügbarkeit auf den Wert AVAIL(ERROR) zurückgesetzt.

## Gemeinsam genutzte Status- und Verfügbarkeitsinformationen für gemeinsam genutzte Nachrichtendateien

Die Verfügbarkeit jeder gemeinsam genutzten Nachrichtendatei wird innerhalb der Gruppe mithilfe von gemeinsam genutzten Statusinformationen verwaltet, die mit dem Befehl **DISPLAY CFSTATUS** mit TYPE (SMDS) angezeigt werden können. Mit diesem Befehl werden die Statusinformationen für jeden Warteschlangenmanager angezeigt, der eine Nachrichtendatei für die jeweilige Struktur aktiviert hat. Jede Nachrichtendatei kann einen der folgenden Status aufweisen:

**NOTFOUND**
Dies bedeutet, dass die entsprechende Nachrichtendatei noch nicht aktiviert wurde. Dieser Status wird nur angezeigt, wenn ein bestimmter Warteschlangenmanager angegeben wurde, denn bei Auswahl aller Warteschlangenmanager werden nicht aktivierte Nachrichtendateien einfach übersprungen.

**NEU**
Das Dataset wird gerade zum ersten Mal geöffnet und initialisiert und ist danach bereit, um aktiviert zu werden.

**AKTIV**
Dies bedeutet, dass die Nachrichtendatei vollständig verfügbar ist und von allen aktiven Warteschlangenmanagern für die Struktur zugewiesen und geöffnet werden kann.

**FAILED**
Dies bedeutet, dass die Nachrichtendatei überhaupt nicht verfügbar ist (außer für Wiederherstellungsvorgänge) und von allen Warteschlangenmanagern geschlossen und freigegeben werden muss.

**INRECOVER**
Dies bedeutet, dass die Datenträgerwiederherstellung (ausgelöst durch den Befehl 'RECOVER CFSTRUCT') für diese Nachrichtendatei in Bearbeitung ist.

**RECOVERED**
Dies zeigt an, dass ein Befehl zum Zurücksetzen einer fehlgeschlagenen Nachrichtendatei in den aktiven Status ausgegeben wurde, jedoch weitere Vorgänge für den Neustart erforderlich sind, die noch nicht abgeschlossen sind, sodass die Nachrichtendatei nur von dem zugehörigen Warteschlangenmanager, der der Besitzer ist, für den Neustart geöffnet werden kann.

**EMPTY**
Das Dataset enthält keine Nachrichten. Die Nachrichtendatei erhält diesen Status, wenn sie zu einem Zeitpunkt, zu dem sie keine Nachrichten enthält, vom zugehörigen Warteschlangenmanager, der der Besitzer ist, auf normale Weise geschlossen wird. Der Status 'EMPTY' kann auch zugewiesen werden, wenn der vorherige Inhalt des Datasets gelöscht werden soll, weil der Inhalt der Anwendungsstruktur gelöscht wurde (entweder ausgelöst durch den Befehl **RECOVER CFSTRUCT** mit der Angabe 'TYPE PURGE' oder indem die vorherige Instanz der Struktur gelöscht wurde, jedoch nur, sofern es sich um eine nicht wiederherstellbare Struktur handelt). Beim nächsten Öffnen des Datasets durch den zugehörigen Warteschlangenmanager wird das Speicherabbild auf den Status 'EMPTY' zurückgesetzt und der Status des Datasets in 'ACTIVE' geändert. Da der vorherige Inhalt des Datasets nicht mehr benötigt wird, kann ein Dataset mit diesem Status durch ein neu zugewiesenes Dataset ersetzt werden, beispielsweise um die Bereichszuordnung zu ändern oder das Dataset auf einen anderen Datenträger zu verschieben.

In der Befehlsausgabe sind Datum und Uhrzeit des Zeitpunkts angegeben, zu dem gegebenenfalls die Wiederherstellungsprotokollierung aktiviert wurde, sowie Datum und Uhrzeit des Zeitpunkts, zu dem die Nachrichtendatei fehlschlug, sofern sie aktuell nicht aktiv ist.

Eine gemeinsam genutzte Nachrichtendatei kann mit einem **RESET SMDS** -Befehl oder automatisch in den Status FAILED versetzt werden, wenn einer der folgenden Fehlertypen erkannt wird:

- Die Nachrichtendatei kann durch den zugehörigen Warteschlangenmanager, der der Besitzer ist, nicht zugewiesen oder geöffnet werden.
- Die Gültigkeitsprüfung des Dateideckblatts schlägt fehl, nachdem die Nachrichtendatei erfolgreich durch einen Warteschlangenmanager geöffnet wurde.
- Wenn der zugehörige Warteschlangenmanager, der der Besitzer ist, Daten liest oder schreibt, tritt ein permanenter E/A-Fehler auf.

- Wenn ein anderer Warteschlangenmanager Daten aus der Nachrichtendatei liest, deren Öffnung und Gültigkeitsprüfung erfolgreich abgeschlossen wurden, tritt ein permanenter E/A-Fehler auf.

Wenn eine Nachrichtendatei den Status FAILED oder INRECOVER hat, steht sie für die normale Verwendung nicht zur Verfügung, und wenn sie zu diesem Zeitpunkt den Verfügbarkeitsstatus ACCESS(ENABLED) hat, wird er in ACCESS(SUSPENDED) geändert.

Wenn eine Nachrichtendatei den Status FAILED erhalten hat, obwohl keine Wiederherstellung erforderlich ist, z. B. weil die Daten noch gültig sind, doch die Speichereinheit vorübergehend offline war, dann kann die Änderung des Status in RECOVERED direkt mit dem Befehl **RESET SMDS** angefordert werden.

Wenn die Datei nach Abschluss der Wiederherstellungsverarbeitung oder als Ergebnis des Befehls **RESET SMDS** in den Status RECOVERED versetzt wird, kann sie nach Abschluss der Neustartverarbeitung erneut verwendet werden. Falls sie den Status ACCESS(SUSPENDED) hatte, wird sie automatisch auf den Status ACCESS(ENABLED) zurückgesetzt, der dem zugehörigen Warteschlangenmanager, der der Besitzer ist, das Ausführen des Neustarts ermöglicht. Nach Abschluss des Neustarts wird der Status in ACTIVE geändert und alle Warteschlangenmanager können dann wieder eine Verbindung zu der Nachrichtendatei herstellen.

## Verbindungsstatus- und Verfügbarkeitsinformationen für gemeinsam genutzte Nachrichtendateien

Jeder Warteschlangenmanager verwaltet lokale Status- und Verfügbarkeitsinformationen über seine Verbindungen zu den einzelnen gemeinsam genutzten Nachrichtendateien, deren Besitzer er selbst ist oder deren Besitzer andere Warteschlangenmanager in der Gruppe sind. Diese Informationen können mit dem **DISPLAY SMDSCONN** -Befehl angezeigt werden.

Wenn der Warteschlangenmanager keinen Zugriff auf eine gemeinsam genutzte Nachrichtendatei mit dem Status ACTIVE hat, deren Besitzer ein anderer Warteschlangenmanager ist, markiert er diese Verbindung aus seiner Sicht als nicht verfügbar.

Wenn der Fehler definitiv auf ein Problem mit der Nachrichtendatei selbst hindeutet, ändert der Warteschlangenmanager automatisch auch den gemeinsam genutzten Status in FAILED. Wenn es sich bei dem Problem jedoch um einen umgebungsbedingten Fehler handeln könnte, z. B. eine fehlende Berechtigung zum Öffnen der Nachrichtendatei, dann gibt der Warteschlangenmanager entsprechende Fehlernachrichten aus und behandelt die gemeinsam genutzte Nachrichtendatei als nicht verfügbar, ändert jedoch nicht deren Status. Wenn sich der als umgebungsbedingt eingestufte Fehler doch als Problem mit der Nachrichtendatei selbst herausstellt (z. B. bei Bereitstellung der Nachrichtendatei auf einer Einheit, auf die einige Warteschlangenmanager nicht zugreifen können), kann ein Bediener den Befehl RESET SMDS mit der Angabe STATUS(FAILED) verwenden, damit die Nachrichtendatei nach Bedarf wiederhergestellt oder repariert werden kann.

Wenn zu einer gemeinsam genutzten Nachrichtendatei, keine Verbindung hergestellt werden konnte, obwohl sie gültig zu sein scheint, ist ein neuer Versuch zu deren Verwendung möglich, indem der Befehl **START SMDSCONN** für den zugehörigen Warteschlangenmanager ausgegeben wird.

Wenn die Verbindung zwischen einem bestimmten Warteschlangenmanager und einer Datei vorübergehend beendet werden muss, die Datei selbst jedoch nicht beschädigt ist, kann die Datei mit dem Befehl **STOP SMDSCONN** geschlossen und freigegeben werden. Wenn die Nachrichtendatei in Verwendung ist, schließt der Warteschlangenmanager sie auf normale Weise (obwohl Anforderungen für Daten in der Nachrichtendatei mit einem Rückkehrcode abgelehnt werden). Wenn der Warteschlangenmanager der Besitzer der Nachrichtendatei ist, speichert er das Speicherabbild während der Verarbeitung des Befehls CLOSE, um den ansonsten notwendigen Neustart zu vermeiden.

Wenn eine Datei vorübergehend von allen Warteschlangenmanagern außer Betrieb genommen werden muss (z. B. um sie zu verschieben), aber nicht beschädigt ist, ist es am besten, **STOP SMDSCONN** für die relevante Datei mit der Option CMDSCOPE (*) zu verwenden, um die Warteschlangenmanager, die sie verwenden, zu stoppen, da dadurch die Neustartverarbeitung vermieden wird, wenn die Datei wieder in Betrieb genommen wird. Wenn die Nachrichtendatei hingegen den Status FAILED hat, ist dies ein Hinweis für die Warteschlangenmanager, die Verwendung der Nachrichtendatei sofort zu stoppen und deren Speicherabbild nicht zu speichern, damit es beim Neustart erneut erstellt wird.

Der Zugriff auf alle gemeinsam genutzten Nachrichtendateien, die zuvor den Status ACCESS(SUSPENDED) hatten, wird bei einem Neustart des Warteschlangenmanagers erneut versucht.

## Wiederherstellungsprotokollierung für gemeinsam genutzte Nachrichtendateien

Permanente gemeinsam genutzte Nachrichten werden zum Zweck der Datenträgerwiederherstellung protokolliert. Dies bedeutet, dass die Nachrichten nach einem Fehlschlag von Coupling-Facility-Strukturen oder gemeinsam genutzten Nachrichtendateien wiederhergestellt werden können, sofern die Wiederherstellungsprotokolle noch intakt sind. Nach einem schweren Störfall können permanente Nachrichten mithilfe der Wiederherstellungsprotokolle auch einem anderen Standort wiederhergestellt werden.

Beim Schreiben von Nachrichtendaten in eine gemeinsam genutzte Nachrichtendatei wird jeder Block gefolgt vom Nachrichteneintrag (einschließlich Datenzuordnung) in derselben Schreibweise wie in der Coupling-Facility separat protokolliert. Der Wiederherstellungsprozess stellt immer die Coupling-Facility-Struktur wieder her, aber er muss nicht einzelne gemeinsam genutzte Nachrichtendateien wiederherstellen, außer wenn der Dateistatus FAILED lautet oder oder wenn der Status ACTIVE lautet, aber der Dateiheadersatz nicht mehr gültig ist, was bedeutet, dass die Datei erneut erstellt wurde. Eine Datei wird weder dann zur Wiederherstellung ausgewählt, wenn ihr Status ACTIVE lautet und der Dateiheader weiterhin gültig ist, noch wenn ihr Status EMPTY lautet, was bedeutet, dass zum Zeitpunkt des Ausfalls keine Nachrichten in ihr gespeichert waren.

## Sicherungen von gemeinsam genutzten Nachrichtendateien

Beim Erstellen einer Sicherung der gemeinsam genutzten Nachrichten in einer Anwendungsstruktur mit dem Befehl BACKUP CFSTRUCT werden gleichzeitig alle Daten für permanente Nachrichten, die in gemeinsam genutzten Nachrichtendateien gespeichert sind, sowie für permanente gemeinsam genutzte Nachrichten, die zuvor in einer Datenbank gespeichert wurden, ebenfalls gesichert.

## Wiederherstellung von gemeinsam genutzten Nachrichtendateien

Wenn eine gemeinsam genutzte Nachrichtendatei beschädigt wird oder verloren geht, muss sie in den Status FAILED versetzt werden, damit die Warteschlangenmanager sie nicht mehr verwenden, bis sie repariert wurde. Dieser Vorgang wird normalerweise automatisch ausgeführt, kann jedoch auch durch den Befehl **RESET SMDS** mit der Angabe STATUS(FAILED) ausgelöst werden.

Wenn die gemeinsam genutzte Nachrichtendatei permanente Nachrichten enthielt, können diese mit dem Befehl RECOVER CFSTRUCT wiederhergestellt werden. Mit diesem Befehl werden zunächst alle permanenten Nachrichtendaten für die gemeinsam genutzte Nachrichtendatei aus dem zuletzt ausgeführten Befehl BACKUP CFSTRUCT wiederhergestellt und anschließend alle seitdem protokollierten Änderungen darauf angewendet. Wenn seit der ersten Aktivierung der Datei kein Befehl **BACKUP CFSTRUCT** ausgeführt wurde, wird er auf leer zurückgesetzt, und alle Änderungen seit der Aktivierung werden angewendet.

Wenn der CFSTRUCT-Inhalt und alle gemeinsam genutzten Nachrichtendateien nicht verfügbar sind, beispielsweise in einer Disaster-Recovery-Situation, können sie alle mit einem einzigen **RECOVER CFSTRUCT** -Befehl wiederhergestellt werden.

Wenn eine gemeinsam genutzte Nachrichtendatei beschädigt ist, aber keine Wiederherstellung für das CFSTRUCT-Objekt aktiv war oder das Protokoll mit den Ergebnissen des zuletzt ausgeführten Befehls BACKUP CFSTRUCT nicht verfügbar oder unbrauchbar ist, können die in diese Nachrichtendatei ausgelagerten Nachrichten nicht wiederhergestellt werden. In diesem Fall kann der Befehl **RECOVER CFSTRUCT** mit dem Parameter TYPE(PURGE) ausgeführt werden, um die gemeinsam genutzte Nachrichtendatei als leer zu markieren und alle Nachrichten von der Struktur, die Daten in dieser Nachrichtendatei gespeichert hatte, zu löschen.

Wenn der Befehl **RECOVER CFSTRUCT** ausgegeben wird, ändert sich der Status der gemeinsam genutzten Nachrichtendatei von FAILED in INRECOVER. Bei erfolgreichem Abschluss der Wiederherstellung wird der Status automatisch in RECOVERED geändert, andernfalls wechselt er zu FAILED.

Wenn die Nachrichtendatei zum Status RECOVERED wechselt, ist dies der Hinweis für den zugehörigen Warteschlangenmanager, der der Besitzer ist, dass er nun versuchen kann, die Nachrichtendatei zu öffnen und deren Neustart auszuführen.

## Wiederherstellung und Synchronisationspunkte von gemeinsam genutzten Nachrichtendateien

Bei der Wiederherstellung einer gemeinsam genutzten Nachrichtendatei werden die Änderungen aus allen vollständigen Protokolleinträgen bis zum Ende des Protokolls ohne Berücksichtigung von Synchronisationspunkten erneut angewendet.

Wenn während der Synchronisationspunkt-, Neustart- oder Wiederherstellungsverarbeitung Änderungen vorgenommen wurden, kann dies dazu führen, dass nicht festgeschriebene Anforderungen für das CFSTRUCT-Objekt zurückgesetzt werden, sodass einige der wiederhergestellten Änderungen gar nicht verwendet werden. Es schadet jedoch nicht, sie dennoch wiederherzustellen.

Es kann auch passieren, das eine nicht festgeschriebene MQPUT-Nachricht zwar in die Struktur geschrieben wurde, die zugehörigen Daten jedoch nicht in die Nachrichtendatei oder das Protokoll geschrieben wurden (weil der E/A-Abschluss nur zu Beginn der Synchronisationspunktverarbeitung erzwungen wird). Dies ist jedoch harmlos, weil der Nachrichteneintrag in der Struktur beim Neustart zurückgesetzt wird und die Tatsache, dass er auf nicht wiederhergestellte Daten verweist, dann unerheblich ist.

## Neustart von gemeinsam genutzten Nachrichtendateien

Wenn eine Warteschlangenmanager-Verbindung zu einem CFSTRUCT-Objekt normal beendet wird, lagert der Warteschlangenmanager das Speicherabbild des freien Blocks für jede gemeinsam genutzte Nachrichtendatei unmittelbar vor dem Schließen der jeweiligen Nachrichtendatei in einen Prüfpunktbereich innerhalb der Nachrichtendatei aus. Das Speicherabbild kann dann beim Neustart der Verbindung erneut eingelesen werden, sofern vor dem nächsten Neustart weder für das CFSTRUCT-Objekt noch die gemeinsam genutzte Nachrichtendatei eine Wiederherstellung erforderlich ist.

Wenn ein Warteschlangenmanager jedoch abnormal beendet wird oder die Struktur oder die Nachrichtendatei wiederhergestellt werden müssen, sind zusätzliche Verarbeitungsschritte erforderlich, um das Speicherabbild dynamisch wiederherzustellen, sobald die Warteschlangenmanager-Verbindung zu der Struktur neu gestartet wird.

Sofern die Nachrichtendatei selbst nicht wiederhergestellt werden muss, werden beim Neustart des Warteschlangenmanagers nur der aktuelle Inhalt der Struktur gescannt, um Verweise auf Nachrichtendaten zu suchen, deren Besitzer der aktuelle Warteschlangenmanager ist, und die relevanten Datenblöcke im Speicherabbild als zu einem Besitzer gehörend markiert. Während das Speicherabbild wiederhergestellt wird, können andere Warteschlangenmanager die Struktur weiterhin verwenden und die Daten lesen, deren Besitzer der neu startende Warteschlangenmanager ist.

## Neustart nach der Wiederherstellung von gemeinsam genutzten Nachrichtendateien

Wenn eine gemeinsam genutzte Nachrichtendatei aus einer Sicherung wiederhergestellt werden musste, sind dabei alle in der Nachrichtendatei gespeicherten nicht permanenten Daten verloren gegangen. Und wenn die Nachrichtendatei mit TYPE(PURGE) wiederhergestellt wurde, dann sind dabei alle darin gespeicherten Nachrichten verloren gegangen. Bis zum Abschluss der Wiederherstellung hat die Nachrichtendatei den Status FAILED oder INRECOVER, und wenn ein anderer Warteschlangenmanager währenddessen versucht, eine der betreffenden Nachrichten zu lesen, wird die Fehlermeldung angezeigt, dass die Nachrichtendatei vorübergehend nicht verfügbar ist.

Wenn die Nachrichtendatei wiederhergestellt wurde, erhält sie den Status RECOVERED, damit sie vom zugehörigen Warteschlangenmanager, der der Besitzer ist, für den Neustart geöffnet werden kann, während sie für andere Warteschlangenmanager weiterhin nicht verfügbar bleibt. Beim Neustart des Warteschlangenmanagers wird die Struktur gescannt, um das Speicherabbild für eventuell noch vorhandene Nachrichten wiederherzustellen. Außerdem wird beim Scannen nach Nachrichten gesucht, deren Daten verloren gegangen sind, um sie aus der Struktur zu löschen (oder um sie gegebenenfalls als verloren zu markieren und später zu löschen).

Wenn dieser Neustartscan abgeschlossen ist, wechselt der Status der Nachrichtendatei automatisch von RECOVERED zu ACTIVE, sodass die Nachrichtendatei auch wieder von anderen Warteschlangenmanagern verwendet werden kann.

## Informationen zur Verwendung von gemeinsam genutzten Nachrichtendateien

Mit dem Befehl DISPLAY USAGE werden für alle aktuell geöffneten, gemeinsam genutzten Nachrichtendateien jetzt auch Informationen zu deren Speicherplatz- und Pufferpoolverwendung angezeigt. Dazu muss entweder die neue Option TYPE(SMDS) oder die vorhandene Option TYPE(ALL) angegeben werden.

## Überlegungen zur Leistung und Kapazität von gemeinsam genutzten Nachrichtendateien

### Dateibelegung überwachen

Der aktuelle Prozentsatz der Belegung jeder einzelnen, gemeinsam genutzten Nachrichtendatei kann mit dem Befehl **DISPLAY USAGE** und der Option **TYPE(SMDS)** angezeigt werden.

Der Warteschlangenmanager erweitert normalerweise automatisch eine gemeinsam genutzte Nachrichtendatei, wenn sie zu 90 % belegt ist, vorausgesetzt, die Option **DSEXPAND(YES)** ist für die SMDS-Definition gültig. Dies trifft zu, wenn entweder die SMDS-Option auf **DSEXPAND(YES)** oder die SMDS-Option auf **DSEXPAND(DEFAULT)** und die CFSTRUCT-Standardoption auf **DSEXPAND(YES)** gesetzt sind.

Wenn der Erweiterungsversuch fehlschlägt, weil bei der Erstellung der Datei keine sekundäre Bereichszuordnung angegeben wurde (Ausgabe der Nachricht IEC070I mit dem Ursachencode 203), wiederholt der Warteschlangenmanager die Erweiterungsanforderung, wobei er eine Überschreibung der sekundären Bereichszuordnung von ungefähr 20 % der aktuellen Größe verwendet.

Bei einer Dateierweiterung werden die neuen Dateispeicherbereiche im Rahmen der Erweiterungsverarbeitung formatiert, was einige zehn Sekunden und bei sehr großen Erweiterungen sogar Minuten dauern kann. Der neue Speicherbereich ist verfügbar, sobald die Formatierung abgeschlossen ist und der Katalog aktualisiert wurde, sodass er das neue Steuerintervall für hohe Belegung anzeigt.

Wenn neue Nachrichten sehr schnell erstellt werden, kann es passieren, dass die bestehende Datei voll wird, bevor die Erweiterungsverarbeitung abgeschlossen ist. In diesem Fall wird jede Anforderung, mit der kein Speicherbereich zugeordnet werden kann, so lange ausgesetzt, bis der Erweiterungsversuch erfolgreich ist und der neue Speicherbereich zur Verfügung steht. Sobald die Erweiterung erfolgreich war, wird die Anforderung automatisch wiederholt.

Wenn ein Erweiterungsversuch fehlschlägt, weil nicht genügend Speicherplatz verfügbar ist oder die maximale Anzahl an Speicherbereichen bereits erreicht ist, wird eine Nachricht mit der Ursache für den Fehler ausgegeben. Anschließend wird die Überschreibungsoption für die betroffene SMDS automatisch in **DSEXPAND(NO)** geändert, um weitere Erweiterungsversuche zu verhindern. In diesem Fall besteht das Risiko, dass die Datei voll wird, wodurch weitere Maßnahmen erforderlich werden können, wie sie im Abschnitt Datei wird voll beschrieben werden.

### Belegung der Anwendungsstruktur überwachen

Die Nutzungsstufe einer Anwendungsstruktur kann mit dem MVS-Befehl **DISPLAY XCF,STRUCTURE** unter Angabe des vollständigen Namens der Anwendungsstruktur (einschließlich des Präfix der Gruppe mit gemeinsamer Warteschlange) angezeigt werden. In der Antwortnachricht IXC360I wird die aktuelle Belegung von Elementen und Einträgen angezeigt.

Wenn die Strukturbelegung den in der CFRM-Richtlinie angegebenen **FULLTHRESHOLD**-Wert überschreitet, gibt das System die Nachricht IXC585E aus und führt unter Umständen automatische **ALTER**-Aktionen aus (falls angegeben), durch möglicherweise entweder das Verhältnis von Eintrag zu Element geändert oder die Struktur vergrößert wird.

### Pufferpoolgrößen optimieren

Jeder Puffer in einem gemeinsam genutzten Pufferpool wird zum Lesen oder Schreiben eines zusammenhängenden Seitenbereichs für eine einzige Nachricht bis zur maximalen Größe des logischen Blocks verwendet. Wenn die Nachricht in weitere Blöcke überläuft, ist für jeden Seitenbereich in einem separaten Block ein separater Puffer erforderlich.

Puffer, die nach einer Schreib- oder Leseoperation Nachrichtendaten enthalten, werden im Speicher beibehalten und mithilfe eines LRU-Cacheschemas wiederverwendet, sodass bei einer kurz danach

erfolgenden Anforderung zum Lesen derselben Daten nicht auf den Datenträger zugegriffen werden muss. Dies führt zu einer signifikanten Optimierung, wenn gemeinsam genutzte Nachrichten geschrieben und kurz darauf von Anwendungen auf demselben System gelesen werden. Auch wenn Nachrichten, deren Eigner ein anderer Warteschlangenmanager ist, zu Auswahlzwecken angezeigt und dann abgerufen werden, müssen diese Nachrichten nicht erneut vom Datenträger gelesen werden.

Dies bedeutet, dass für jede Anwendungsstruktur ein einziger Puffer für jede parallele API-Anforderung, mit der lange Nachrichten für die betreffende Anwendungsstruktur gelesen oder geschrieben werden, sowie einige zusätzliche Puffer erforderlich sind, in denen zuletzt gelesene Daten zur Optimierung nachfolgender Lesezugriffe gespeichert werden.

Wenn bei gemeinsam genutzten Pufferpools nicht genug Puffer vorhanden sind, warten API-Anforderungen einfach, falls nicht sofort ein Puffer verfügbar ist. Diese Situation sollte jedoch vermieden werden, da sich die Leistung dadurch signifikant verschlechtern kann.

Die Statistik des Befehls **DISPLAY USAGE** für gemeinsam genutzte Pufferpools zeigt, ob innerhalb des aktuellen Statistikintervalls Pufferwartezeiten aufgetreten sind. Angezeigt werden außerdem die niedrigste Anzahl freier Puffer (bzw. ein negativer Wert, der die maximale Anzahl Threads, die gleichzeitig auf einen Puffer warteten, angibt), die Anzahl der Puffer, die gespeicherte Daten enthalten, und der Prozentsatz der Zeiten, in denen eine Pufferanforderung erfolgreich auf gespeicherte Daten in der LRU-Kette zugreifen konnte ("LRU-Treffer"), statt die Daten lesen zu müssen ("LRU-Fehlversuche")[1].

- Wenn Wartezeiten auftraten, sollte die Anzahl der Puffer erhöht werden.
- Wenn es viele ungenutzte Puffer gibt, kann die Anzahl der Puffer verringert werden, um mehr Speicher in dem betreffenden Bereich für andere Zwecke verfügbar zu machen.
- Wenn es viele Puffer mit gespeicherten Daten gibt, aber der Anteil der erfolgreichen Lesezugriffe auf die gespeicherten Daten sehr niedrig ist, kann die Anzahl der Puffer verringert werden, falls der Speicher für andere Zwecke besser genutzt werden kann. Die Anzahl der Puffer sollte jedoch nicht um mehr als die niedrigste Anzahl freier Puffer verringert werden, da dies zu Wartezeiten führen kann, und sie sollte immer so hoch sein, dass der Zähler für freie Speicher normalerweise deutlich über null liegt.

## Gemeinsam genutzte Nachrichtendateien löschen

Der Befehl DELETE CFSTRUCT (der nur zulässig ist, wenn alle gemeinsam genutzten Warteschlangen in der Struktur leer und geschlossen sind) löscht nicht die gemeinsam genutzten Nachrichtendateien selbst, sondern kann auf die übliche Weise gelöscht werden, nachdem dieser Befehl ausgeführt wurde. Wenn dieselbe Nachrichtendatei erneut als gemeinsam genutzte Nachrichtendatei verwendet werden soll, muss sie zunächst neu formatiert werden, um sie in den leeren Zustand zurückzusetzen.

## Ausnahmesituationen für gemeinsam genutzte Nachrichtendateien

Es gibt eine Reihe von Ausnahmesituationen, die bei normaler Nutzung oder sogar dann, wenn keine Software- oder Hardwarefehler vorliegen, eintreten können.

### Datei wird voll

Wenn eine Datei voll wird, aber nicht erweitert werden kann oder der Erweiterungsversuch fehlschlägt, erhalten Anwendungen, die über den entsprechenden Warteschlangenmanager lange Nachrichten in die entsprechende Anwendungsstruktur schreiben, den Fehler 2192, MQRC_STORAGE_MEDIUM_FULL (auch bekannt als MQRC_PAGESET_FULL).

Eine Datei kann voll werden, weil in der Anwendung zur Verarbeitung der Daten ein Fehler auftritt, was dazu führt, dass sich ein großer Rückstand von Nachrichten bildet. In diesem Fall stellt jede neue Erweiterung nur eine vorläufige Lösung dar und es ist wichtig, dass die verarbeitende Anwendung so schnell wie möglich wieder ihre Arbeit aufnimmt.

Wenn mehr Speicherplatz zur Verfügung gestellt werden kann, ist es möglich, mit dem Befehl **ALTER SMDS** den Parameter **DSEXPAND(YES)** oder **DSEXPAND(DEFAULT)** einzustellen (vorausgesetzt, dass

[1] (Hits / (Hits+Misses))* 100

für die CFSTRUCT-Definition YES als Wert für **DSEXPAND** festgelegt oder als Standardwert vorausgesetzt wurde), um eine Wiederholung auszulösen. Wenn der Fehler jedoch verursacht wurde, weil die maximale Anzahl der Speicherbereiche erreicht wurde, wird der neue Erweiterungsversuch mit einer Nachricht zurückgewiesen und **DSEXPAND(NO)** wird erneut festgelegt. In diesem Fall besteht die einzige Möglichkeit zur Erweiterung darin, die Datei neu zuzuordnen, was dazu führt, dass sie vorübergehend nicht verfügbar ist (siehe unten).

**Datei muss verschoben oder neu zugeordnet werden**

Wenn eine Datei, die im normalen Gebrauch ist, verschoben oder neu zugeordnet werden muss, kann sie vorübergehend "außer Betrieb" genommen werden, um sie zu verschieben oder neu zuzuordnen. Für jede API-Anforderung, mit der versucht wird, die Datei zu verwenden, während sie nicht verfügbar ist, wird der Ursachencode `MQRC_DATA_SET_NOT_AVAILABLE` empfangen.

1. Markieren Sie die Datei mit dem Befehl **RESET SMDS** als **ACCESS(DISABLED)**. Dies führt dazu, dass die Datei normal geschlossen und die Zuordnung zu allen aktuell verbundenen Warteschlangenmanagern aufgehoben wird.
2. Führen Sie je nach Bedarf die Verschiebung oder Neuzuordnung durch, wobei der alte Inhalt in die neu zugeordnete Datei kopiert wird, z. B. mit dem AMS-Befehl **REPRO** (Access Method Services).

   Versuchen Sie nicht, die neue Datei vorzuformatieren, bevor Sie die alten Daten hineinkopiert haben, denn dies würde dazu führen, dass die kopierten Daten an das Ende der formatierten Datei angehängt würden.
3. Markieren Sie die Datei mit dem Befehl **RESET SMDS** als **ACCESS(ENABLED)**, um sie wieder "in Betrieb" zu nehmen.

   Wenn der alte Inhalt kleiner als die neue Datei ist, wird der übrige Speicherbereich automatisch vorformatiert, sobald die neue Datei geöffnet wird.

   Ist der alte Inhalt umfangreicher als die Größe der neuen Datei, muss der Warteschlangenmanager die Nachrichten in der Coupling-Facility-Struktur scannen und die Speicherzuordnung erneut erstellen, um sicherzustellen, dass keine der aktiven Daten verloren gehen. Wenn ein Verweis auf einen Datenblock gefunden wird, der außerhalb der Erweiterungsbereiche liegt, wird die Datei als **STATUS(FAILED)** markiert und muss repariert werden, indem die Datei durch eine mit der richtigen Größe ersetzt wird und entweder die alte Datei erneut hineinkopiert wird oder alle persistenten Nachrichten mit dem Befehl **RECOVER CFSTRUCT** wiederhergestellt werden.

**Coupling-Facility-Struktur verfügt über zu wenig Speicherplatz**

Wenn in der Coupling-Facility-Struktur nicht mehr genug Speicherplatz verfügbar ist, sodass die Nachricht IXC585E ausgegeben wird, sollte überprüft werden, ob die Auslagerungsregeln so festgelegt wurden, dass sichergestellt ist, dass in diesem Fall das maximale Datenvolumen ausgelagert wird. Andernfalls können die Auslagerungsregeln mit dem Befehl **ALTER CFSTRUCT** geändert werde.

## Fehlersituationen für gemeinsam genutzte Nachrichtendateien

Es gilt mehrere Probleme zu beachten, die nur durch Fehler verursacht werden können und in normalen Betriebssituationen nicht auftreten.

**Eigner kann Datei nicht öffnen**

Wenn der Warteschlangenmanager, der Eigner einer gemeinsam genutzten Nachrichtendatei ist, diese nicht zuordnen oder öffnen kann oder die Dateiattribute nicht unterstützt werden, setzt der Warteschlangenmanager einen entsprechenden **SMDSCONN**-Statuswert von **ALLOCFAIL** oder **OPENFAIL** und setzt die **SMDSCONN**-Verfügbarkeit auf **AVAIL(ERROR)**. Außerdem wird die SMDS-Verfügbarkeit auf **ACCESS(SUSPENDED)** gesetzt. Wenn der Fehler behoben wurde, verwenden Sie den **RESET SMDS** -Befehl, um **ACCESS(ENABLED)** festzulegen, um eine Wiederholung auszulösen, oder geben Sie den Befehl **START SMDSCONN** an den Warteschlangenmanager aus, der Eigner ist.

**Schreibgeschützte Datei kann nicht geöffnet werden**

Wenn ein Warteschlangenmanager eine gemeinsam genutzte Nachrichtendatei, deren Eigner ein anderer Warteschlangenmanager ist und die als **STATUS(ACTIVE)** markiert ist, nicht zuordnen oder öffnen kann, geht er davon aus, dass dies wahrscheinlich auf ein bestimmtes Problem mit seiner Verbindung zur Datei (dargestellt durch das **SMDSCONN**-Objekt) und nicht auf ein Problem mit der Datei selbst zurückzuführen ist.

Sie markiert **SMDSCONN** als **STATUS(ALLOCFAIL)** oder **STATUS(OPENFAIL)** als geeignet und markiert die **SMDSCONN**-Verfügbarkeit als AVAIL (ERROR), um weitere Versuche zu ihrer Verwendung zu verhindern.

Wenn das Problem behoben werden kann, ohne dass sich dies auf den Status der Datei selbst auswirkt, lösen Sie mit dem Befehl **START SMDSCONN** eine Wiederholung aus.

Wenn sich das Problem als Problem mit der Datei selbst herausstellt, kann der Befehl **RESET SMDS** verwendet werden, um die Datei als **STATUS(FAILED)** zu markieren, bis sie wiederhergestellt wurde. Wenn die Datei wiederhergestellt ist, führt die Aktion zum Ändern des Status zurück in **STATUS(ACTIVE)** dazu, dass andere Warteschlangenmanager benachrichtigt werden. Wenn **SMDSCONN** als **AVAIL(ERROR)** markiert ist, wird er automatisch wieder in **AVAIL(NORMAL)** geändert, um einen neuen Versuch zum Öffnen der Datei auszulösen.

**Dateiheader ist beschädigt**

Wenn die Datei erfolgreich geöffnet wurde, das Format der Headerinformationen jedoch falsch ist, schließt der Warteschlangenmanager die Datei, hebt die Zuordnung auf und setzt den Status auf **STATUS(FAILED)** und die Verfügbarkeit auf **ACCESS(SUSPENDED)**. Dadurch ist es möglich, den Inhalt mit dem Befehl **RECOVER CFSTRUCT** wiederherzustellen.

Wenn der Fehler darin bestand, dass die Datei vorhandene Daten von einer anderen Nutzung enthielt und danach nicht vorformatiert wurde, müssen Sie die Datei vorformatieren und den Status mit dem Befehl **RESET SMDS** auf **STATUS(RECOVERED)** setzen.

Andernfalls muss die Datei wiederhergestellt werden.

**Datei ist unerwarteterweise leer**

Wenn der Warteschlangenmanager eine Datei öffnet, die als **STATUS(ACTIVE)** markiert ist, aber feststellt, dass sie nicht initialisiert oder neu vorformatiert, aber ansonsten gültig ist, schließt der Warteschlangenmanager die gemeinsam genutzte Nachrichtendatei und hebt die Zuordnung auf. Anschließend wird der Status auf **STATUS(FAILED)** und die Verfügbarkeit auf **ACCESS(SUSPENDED)** gesetzt.

**Datei weist permanente E/A-Fehler auf**

Wenn eine Datei nach erfolgreicher **OPEN**-Verarbeitung permanente E/A-Fehler aufweist, muss sie vermutlich wiederhergestellt werden. Der Warteschlangenmanager markiert die Datei als **STATUS(FAILED)**, sodass sie von allen derzeit verbundenen Warteschlangenmanagern geschlossen und freigegeben wird.

**Datei weist behebbare E/A-Fehler auf**

Wenn es Hardwareprobleme mit der Datei gibt, kann dies behebbare E/A-Fehlern verursachen, die nicht an den Warteschlangenmanager zurückgemeldet werden, aber zu deutlichen Leistungseinbußen führen und das Risiko von permanenten E/A-Fehler in naher Zukunft bergen.

In diesem Fall kann die Datei zur Wiederherstellung offline genommen werden, indem der Befehl **RESET SMDS** verwendet wird, um sie als **STATUS(FAILED)** zu markieren. Dies führt dazu, dass die Datei von allen Warteschlangenmanagern geschlossen und die Zuordnung aufgehoben wird. Sie kann dann beispielsweise auf ein neues Laufwerk verschoben werden, bevor sie wieder zur Verfügung gestellt wird.

Wenn die Verfügbarkeit einer Datei auf diese Weise beendet wird, ist die Speicherbereichszuordnung nicht gespeichert, sodass beim Neustart der Warteschlangenmanager-Verbindung die Coupling-Facility-Struktur gescannt werden muss, um Nachrichten in der Datei zu finden und die Speicherbereichszuordnung erneut herzustellen, bevor die Datei erneut zur Verfügung gestellt werden kann. Falls die

gemeinsam genutzte Nachrichtendatei noch verwendbar ist, kann ihre Verfügbarkeit alternativ besser dadurch beendet werden, dass sie mit dem Befehl **RESET SMDS** als **ACCESS(DISABLED)** markiert wird, bis sie wieder soweit hergestellt ist, dass sie erneut zur Verfügung gestellt werden kann.

**Dateiinhalt ist falsch**

Der Warteschlangenmanager kann nicht direkt erkennen, ob eine Datei falsche Daten enthält oder nicht auf dem neuesten Stand ist, weil beispielsweise ein Datenträger, auf dem sich diese Datei befindet, aus Sicherungen wiederhergestellt werden musste. Er führt jedoch Integritätsprüfungen durch, die es nahezu unmöglich machen, dass solche Fehler dazu führen, dass falsche Nachrichtendaten an Anwendungen übergeben werden.

Zur Integritätsprüfung erhält jeder Nachrichtenblock in der Datei eine Kopie der entsprechenden Coupling-Facility-Eintrags-ID als Präfix, einschließlich einer eindeutigen Zeitmarke, die bei jedem Lesen des Nachrichtenblocks überprüft wird, bevor die Nachrichtendaten an das Benutzerprogramm übergeben werden. Wenn das Nachrichtenblockpräfix nicht mit der Eintrags-ID übereinstimmt (und der Coupling-Facility-Eintrag zwischenzeitlich nicht gelöscht wurde), wird angenommen, dass der Nachrichtenblock beschädigt und unbrauchbar ist.

Wenn die beschädigte Nachricht eine persistente Nachricht war, wird die Datei als **STATUS(FAILED)** markiert und muss der Strukturinhalt mit dem Befehl **RECOVER CFSTRUCT** wiederhergestellt werden. War es eine nicht persistente Nachricht, ist eine Wiederherstellung nicht möglich, sodass eine Diagnosenachricht ausgegeben und der entsprechende Coupling-Facility-Eintrag gelöscht wird.

Wenn beim Öffnen der Datei keine gespeicherte Speicherbereichszuordnung verfügbar ist, wird sie wiederhergestellt, indem die Coupling-Facility-Struktur nach Verweisen auf die Daten in der Datei durchsucht wird. Bei diesem Scan führt der Warteschlangenmanager die folgenden Aktionen durch:

1. Er ermittelt die Position der jüngsten Nachricht (falls vorhanden), die zurzeit in der Datei enthalten ist.
2. Er liest die betreffende Nachricht aus der Datei, um sicherzustellen, dass das Blockpräfix mit der Nachrichteneintrags-ID übereinstimmt.

Durch diese Aktionen ist gewährleistet, dass der Warteschlangenmanager erkennt, wenn die Datei einen älteren Stand hat, und sie als FAILED markiert. Diese Überprüfung toleriert jedoch den Fall, dass die Datei aus einer früheren Kopie wiederhergestellt wurde und seitdem entweder keine neuen Nachrichten hinzugefügt oder alle Nachrichten, die seitdem hinzugefügt wurden, anschließend gelesen und gelöscht wurden.

Um sich in dem Fall, dass die Datei normal geschlossen wurde, vor veralteten Daten zu schützen, führt der Warteschlangenmanager folgende Aktionen durch:

1. Beim normalen Schließen der Datei speichert er eine Kopie der Zeitmarke der Speicherbereichszuordnung im SMDS-Objekt in Db2.
2. Beim erneuten Öffnen der Datei überprüft er, ob die Zeitmarke der Speicherbereichszuordnung übereinstimmt.

Wenn die Zeitmarke nicht übereinstimmt, ist davon auszugehen, dass eine veraltete Kopie der Datei verwendet wurde, sodass der Warteschlangenmanager die bestehende Speicherbereichszuordnung ignoriert und erneut erstellt, was nur erfolgreich ist, wenn tatsächlich keine Nachrichtendaten verloren gegangen sind.

**Anmerkung:** Diese Integritätsprüfungen garantieren nicht, dass in allen theoretisch möglichen Fällen eine veraltete oder beschädigte Datei erkannt wird. Es wird beispielsweise nicht erkannt, wenn der Anfang eines Nachrichtenblocks gültig ist, aber die übrigen Daten teilweise überschrieben wurden.

## Wiederherstellungsszenarios für gemeinsam genutzte Nachrichtendateien

In diesem Abschnitt werden Wiederherstellungsszenarios für gemeinsam genutzte Nachrichtendateien beschrieben.

**Dateiwiederherstellung, wenn keine Daten verloren gingen**

In manchen Fällen kann der richtige Inhalt einer fehlerhaften Datei wiederhergestellt werden, ohne dass eine tatsächliche Wiederherstellung erforderlich ist. Ein Beispiel dafür ist, wenn eine Datei noch Daten von einer vorherigen Nutzung enthält und nicht erneut vorformatiert wurde; zur Behebung des Problems ist dann lediglich eine Vorformatierung erforderlich. Ein anderes Beispiel ist, wenn eine Datei verschoben wurde, aber beim Kopieren der Daten ein Fehler aufgetreten ist; in diesem Fall kann der Fehler durch erneutes Kopieren der Daten behoben werden.

In solchen Fällen kann die korrigierte Datei wieder verfügbar gemacht werden, indem der Befehl **RESET SMDS** verwendet wird, um **STATUS(RECOVERED)** festzulegen. Wenn die Verfügbarkeit derzeit **ACCESS(SUSPENDED)** ist, wird sie automatisch auf **ACCESS(ENABLED)** zurückgesetzt.

Wenn der Eigner-Warteschlangenmanager benachrichtigt wird, dass die Datei wiederhergestellt wurde, durchsucht er den Strukturinhalt, um die Speicherbereichszuordnung wiederherzustellen, und ändert dann den Status in **STATUS(ACTIVE)**. Den anderen Warteschlangenmanagern steht die Datei dann wieder zum Lesen zur Verfügung.

**Dateiwiederherstellung mit TYPE(NORMAL)**

Wenn der Inhalt einer Datei verloren gegangen ist, die Anwendungsstruktur aber mit **RECOVER(YES)** definiert wurde und die geeigneten Wiederherstellungsprotokolle verfügbar sind, können alle in der Struktur gespeicherten persistenten Nachrichten, einschließlich persistenter Nachrichtendaten, die in gemeinsam genutzte Nachrichtendateien ausgelagert wurden, mit dem Befehl **RECOVER CFSTRUCT** wiederhergestellt werden. Dieser Befehl stellt mithilfe der mit dem Befehl **BACKUP CFSTRUCT** protokollierten Informationen sowie aller seit dem Sicherungszeitpunkt protokollierten Änderungen an persistenten Nachrichten den aktuellen Status wieder her.

Der Befehl **RECOVER CFSTRUCT** stellt immer alle persistenten Nachrichten in der Coupling-Facility-Struktur zusammen mit ausgelagerten Nachrichtendaten, die in Db2 gespeichert sind, wieder her. Für ausgelagerte Daten, die in gemeinsam genutzten Nachrichtendateien gespeichert sind, wird jede Datei nur für die Wiederherstellungsverarbeitung ausgewählt, wenn sie bereits als **STATUS(FAILED)** markiert ist oder wenn beim Öffnen durch die Wiederherstellungsverarbeitung festgestellt wird, dass sie unerwartet leer oder anderweitig ungültig ist. Jede gemeinsam genutzte Nachrichtendatei, die als aktiv markiert ist und die Gültigkeitsprüfungen besteht, muss nicht wiederhergestellt werden, da die vorhandenen Nachrichtendaten bereits korrekt sind; allerdings wird der Header aktualisiert, um darauf hinzuweisen, dass eine eventuell gespeicherte Speicherbereichszuordnung nach einer Wiederherstellung erneut aufgebaut werden muss.

Eine Wiederherstellung ist nur möglich, wenn die Struktur als fehlgeschlagen markiert wurde, da bei der Wiederherstellung der gesamte Inhalt der Struktur erneut aufgebaut werden muss. Wenn jedoch mindestens eine gemeinsam genutzte Nachrichtendatei als fehlgeschlagen markiert wurde, markiert der Befehl **RECOVER CFSTRUCT**, falls nötig, die Struktur automatisch als fehlgeschlagen, damit die Wiederherstellung fortgesetzt werden kann.

Eine Wiederherstellung kann von jedem Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange durchgeführt werden, sofern er über Schreibzugriff auf die relevanten Dateien verfügt.

Es werden nur persistente Nachrichten gesichert und protokolliert, sodass bei einer normalen Wiederherstellung zwar alle persistenten Nachrichten wiederhergestellt werden, aber alle nicht persistenten Nachrichten in der Struktur verloren gehen.

Nach Abschluss der Wiederherstellung wird jede Datei, die für die Wiederherstellung ausgewählt wurde, automatisch in **STATUS(RECOVERED)** geändert. Wenn die Verfügbarkeit **ACCESS(SUSPENDED)** lautete, wird sie in **ACCESS(ENABLED)** geändert. Der Warteschlangenmanager erstellt die Speicherzuordnung für jede Datei erneut, indem er die Nachrichten in der Coupling-Facility scannt und dann die Datei als **STATUS(ACTIVE)** markiert, damit sie wieder verwendet werden kann.

**Dateiwiederherstellung mit TYPE(PURGE)**

Wenn bei einer wiederherstellbaren Struktur die Dateiinhalte verloren gegangen sind, aber eine Wiederherstellung aus bestimmten Gründen nicht möglich ist, z. B. weil keine Wiederherstellungsprotokolle verfügbar sind oder die Wiederherstellung zu lange dauern würde, kann der Befehl **RECOVER CFSTRUCT** mit der Option **TYPE(PURGE)** verwendet werden, um die Struktur wieder in einen ver-

wendbaren Zustand zu versetzen. Dadurch wird die Struktur auf den leeren Status zurückgesetzt und alle zugeordneten Dateien als **STATUS(EMPTY)** markiert.

**Anwendungsstruktur löschen**

Wenn eine nicht wiederherstellbare Anwendungsstruktur mit dem MVS-Befehl **SETXCF FORCE** oder als Ergebnis eines Strukturfehlers gelöscht wird, wird bei der nächsten Verbindung der Struktur die Nachricht CSQE028I ausgegeben, um anzugeben, dass die Struktur zurückgesetzt wurde und alle vorhandenen Nachrichten gelöscht wurden. Alle vorhandenen Dateien werden automatisch auf **STATUS(EMPTY)** zurückgesetzt. Auf diese Weise wird eine nicht wiederherstellbare Struktur wieder verwendbar gemacht, nachdem entweder in der Struktur oder in einer der zugeordneten Dateien Daten verloren gegangen sind.

Wird eine wiederherstellbare Struktur gelöscht, wird sie so behandelt, als ob sie fehlgeschlagen wäre.

**Dateiwiederherstellung schlägt fehl**

Wenn **RECOVER CFSTRUCT** aus irgendeinem Grund nicht abgeschlossen werden kann, z. B. weil eine Protokolldatei nicht mehr verfügbar ist oder weil der Warteschlangenmanager während der Wiederherstellung beendet wurde, wird jede Datei, für die die Wiederherstellung mindestens gestartet wurde, im Header markiert, um anzuzeigen, dass eine partielle Wiederherstellung versucht wurde, und die Datei verbleibt im Status **STATUS(FAILED)** .

In diesem Fall bestehen die Optionen darin, die ursprüngliche Wiederherstellungsanforderung zu wiederholen oder stattdessen mit **TYPE(PURGE)** wiederherzustellen, wobei die vorhandenen Daten gelöscht werden.

Wenn versucht wird, die Datei als **STATUS(RECOVERED)** zu markieren, ohne sie tatsächlich wiederherzustellen, erkennt der Warteschlangenmanager beim nächsten Öffnen, dass der Header eine unvollständige Wiederherstellung angibt, und markiert sie erneut als **STATUS(FAILED)** .

**Disaster Recovery an einem anderen Standort**

Bei einem Disaster Recovery an einem anderen Standort können persistente, gemeinsam genutzte Nachrichten nur mithilfe der Protokolle und der gemeinsam genutzten Db2-Objekte, die die CFSTRUCT-Definitionen und zugehörigen SMDS-Statusinformationen enthalten, erneut erstellt werden.

Nachdem die Db2-Tabellen mit den Definitionen eingerichtet wurden, können die Anwendungsstruktur und die gemeinsam genutzten Nachrichtendateien als leere Objekte erstellt werden. Wenn ein Warteschlangenmanager eine Verbindung zu ihnen herstellt und feststellt, dass sie unerwartet leer sind, markiert er sie als fehlgeschlagen. Danach kann ein einzelner **RECOVER CFSTRUCT** -Befehl verwendet werden, um alle persistenten Nachrichten für alle betroffenen Strukturen wiederherzustellen.

## z/OS *SMDS-bezogene Befehle*

In diesem Abschnitt werden die Befehle für gemeinsam genutzte Nachrichtendateien (Shared Message Data Sets, SMDS) beschrieben und Zugriff auf die Befehle bereitgestellt.

**CFSTRUCT** -Optionen für die Auslagerung großer Nachrichten ( **OFFLOAD** und Auslagerungsregeln) und gemeinsam genutzte Nachrichtendateien ( **DSGROUP, DSBLOCK, DSBUFS, DSEXPAND)**) anzeigen und ändern:

- ANZEIGEN CFSTRUCT
- CFSTRUCT DEFINE CFSTRU
- ALTER CFSTRUCT
- DELETE CFSTRUCT

Anzeige des **CFSTRUCT**-Status für die Auslagerung langer Nachrichten (**(OFFLDUSE)**):

- DISPLAY CFSTATUS

Anzeige und Änderung von Dateiüberschreibungsoptionen (**DSEXPAND** und **DSBUFS**) für einzelne Warteschlangenmanager:

- ANZEIGEN SMDS
- ALTER SMDS

Anzeige oder Änderung des Status und der Verfügbarkeit der Dateien innerhalb der Gruppe mit gemeinsamer Warteschlange:

- DISPLAY CFSTATUS TYPE(SMDS)
- RESET SMDS

Anzeige der SMDS-Speicherbelegungs- und Pufferbelegungsinformationen für einen Warteschlangenmanager:

- DISPLAY USAGE TYPE(SMDS)

Anzeige oder Änderung des Status und der Verfügbarkeit der Verbindungen (**SMDSCONN**) zu den Dateien von einem einzelnen Warteschlangenmanager:

- ANZEIGEN SMDSCONN
- START SMDSCONN
- STOP SMDSCONN

Sicherung und Wiederherstellung von gemeinsam genutzten Nachrichten, einschließlich umfangreicher Nachrichtendaten in einer SMDS, wenn nötig:

- BACKUP CFSTRUCT
- RECOVER CFSTRUCT

z/OS ***Vorteile von gemeinsam genutzten Warteschlangen***

Mithilfe gemeinsam genutzter Warteschlangen werden IBM MQ-Anwendungen skalierbar und hoch verfügbar. Darüber hinaus wird die Implementierung eines Lastausgleichs ermöglicht.

## Vorteile von gemeinsam genutzten Warteschlangen

Die auf gemeinsam genutzten Warteschlangen aufbauende Architektur, bei der geklonte Server Arbeit von einer einzigen gemeinsam genutzten Warteschlange beziehen, hat einige nützliche Eigenschaften:

- Sie ist skalierbar, indem neue Instanzen der Serveranwendung oder sogar ein neues z/OS-Image mit einem Warteschlangenmanager (in der Gruppe mit gemeinsamer Warteschlange) und einer Kopie der Anwendung hinzugefügt werden.
- Sie ist hoch verfügbar.
- Sie sorgt auf natürliche Weise für einen Lastausgleich durch *Extraktion mit Pull-Operationen*, der auf der verfügbaren Verarbeitungskapazität des einzelnen Warteschlangenmanagers in der Gruppe mit gemeinsamer Warteschlange basiert.

## Hochverfügbarkeit durch gemeinsam genutzte Warteschlangen

Die folgenden Beispiele zeigen, wie Sie mithilfe einer gemeinsam genutzten Warteschlange die Anwendungsverfügbarkeit erhöhen können.

Dabei wird ein IBM MQ-Szenario angenommen, in dem Clientanwendungen, die in einem Netz ausgeführt werden, Anforderungen an Serveranwendungen stellen sollen, die unter z/OS ausgeführt werden. Die Clientanwendung erstellt eine Anforderungsnachricht und reiht sie in eine Anforderungswarteschlange ein. Danach wartet der Client auf eine Antwort vom Server, die an die Warteschlange für zu beantwortende Nachrichten gesendet wird, die im Nachrichtendeskriptor der Anforderungsnachricht genannt ist.

IBM MQ verwaltet den Transport der Anforderungsnachricht vom Clientsystem zur Eingabewarteschlange des Servers unter z/OS sowie der Antwortnachricht vom Server zurück zum Client. Wenn die Eingabewarteschlange des Servers als gemeinsam genutzte Warteschlange definiert ist, können alle in die Warteschlange eingereihten Nachrichten von jedem Warteschlangenmanager in der Gruppe mit gemeinsamer

Warteschlange abgerufen werden. Dies bedeutet, wenn Sie in jedem z/OS-Image im Sysplex einen Warteschlangenmanager konfigurieren und sie alle mit derselben Gruppe mit gemeinsamer Warteschlange verbinden, dann kann jeder dieser Warteschlangenmanager auf die Eingabewarteschlange des Servers zugreifen.

Die Nachrichten in der Eingabewarteschlange des Servers sind in diesem Fall auch dann noch verfügbar, wenn einer der Warteschlangenmanager abnormal beendet wird oder zu Verwaltungszwecken gestoppt werden muss. Selbst wenn Sie ein gesamtes z/OS-Image offline stellen, bleiben die Nachrichten nach wie vor verfügbar.

Damit Sie den Vorteil der höheren Verfügbarkeit der Nachrichten in einer gemeinsam genutzten Warteschlange nutzen können, führen Sie in jedem z/OS-Image im Sysplex eine Instanz der Serveranwendung aus, um die Kapazität und Verfügbarkeit der Serveranwendung zu erhöhen, wie in Abbildung 60 auf Seite 205 dargestellt.

Eine Instanz der Serveranwendung ruft eine Anforderungsnachricht aus der gemeinsam genutzten Warteschlange ab, verarbeitet sie gemäß ihrem Inhalt und erstellt ein Ergebnis, das in Form einer IBM MQ-Nachricht an den Client zurückgesendet wird. Die Antwortnachricht ist für die Warteschlange und den Warteschlangenmanager für zu beantwortende Nachrichten bestimmt, die im Nachrichtendeskriptor der Anforderungsnachricht genannt sind.

Es gibt eine Reihe von Optionen, mit denen Sie den Rückkehrpfad konfigurieren können. Weitere Informationen zu diesen Optionen finden Sie unter „Verteilte Steuerung von Warteschlangen und Gruppen mit gemeinsamer Warteschlange” auf Seite 226.

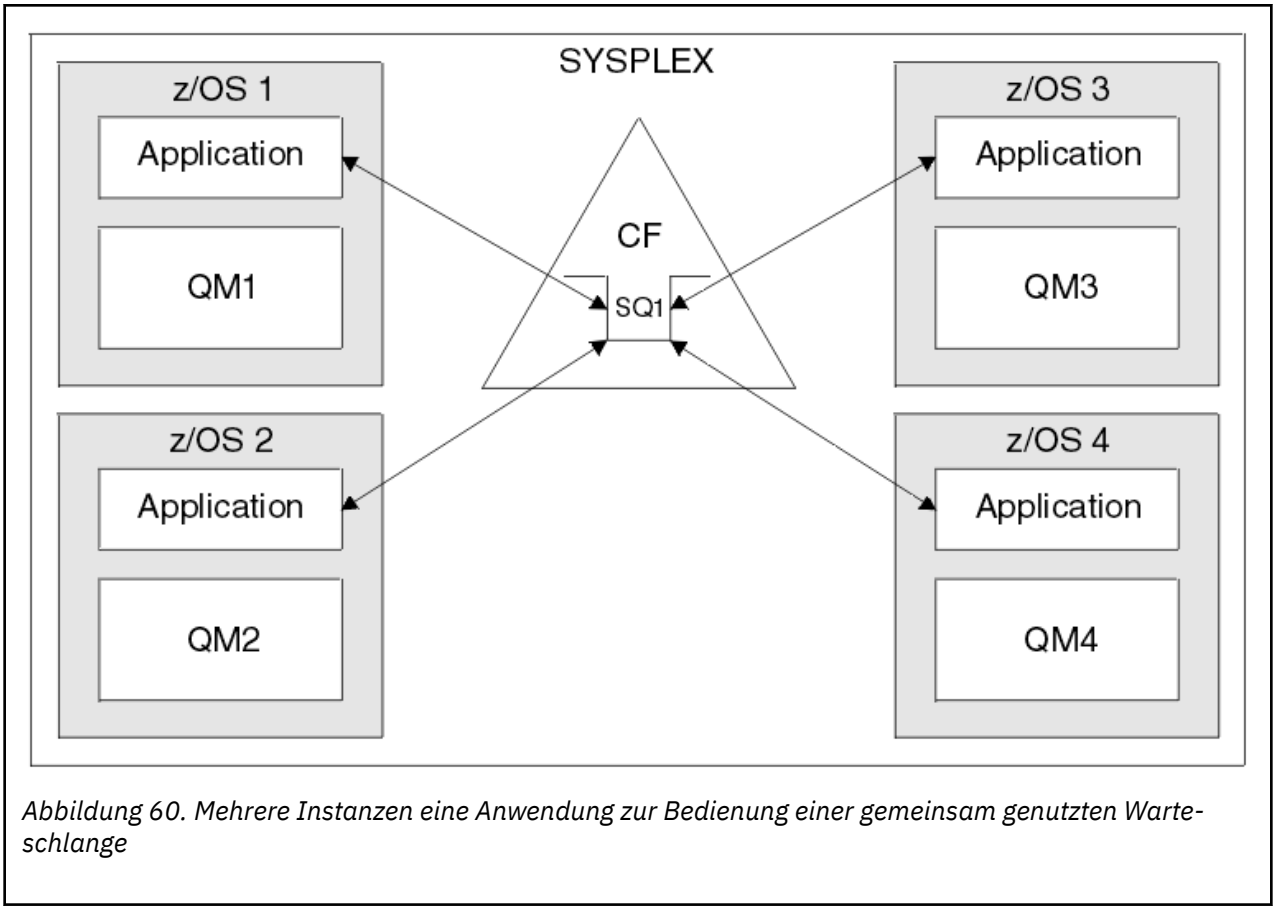



*Abbildung 60. Mehrere Instanzen eine Anwendung zur Bedienung einer gemeinsam genutzten Warteschlange*

## Peerwiederherstellung

Um die Verfügbarkeit von Nachrichten in einer Gruppe mit gemeinsamer Warteschlange weiter zu erhöhen, ermittelt IBM MQ, ob ein anderer Warteschlangenmanager in der Gruppe seine Verbindung zur

Coupling-Facility abnormal beendet hat, und schließt die für diesen Warteschlangenmanager noch anstehenden Arbeitseinheiten ab, falls möglich. Diese Funktion wird als *Peerwiederherstellung* bezeichnet.

Angenommen, ein Warteschlangenmanager wird zu einem Zeitpunkt abnormal beendet, zu dem eine Anwendung eine Anforderungsnachricht aus einer Warteschlange im Synchronisationspunkt abgerufen, aber die Antwortnachricht noch nicht eingereiht oder die Arbeitseinheit noch nicht ausgeführt hat. Wenn ein anderer Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange diesen Fehler erkennt, setzt er die momentan vom fehlerhaften Warteschlangenmanager ausgeführten Arbeitseinheiten zurück. Dies bedeutet, dass die Anforderungsnachricht erneut in die Anforderungswarteschlange eingereiht wird und dort wieder den anderen Serverinstanzen für die Verarbeitung zur Verfügung steht, ohne dass auf den Neustart des fehlgeschlagenen Warteschlangenmanagers gewartet werden muss.

Wenn eine Arbeitseinheit von IBM MQ nicht automatisch aufgelöst werden kann, können Sie den gemeinsam genutzten Teil manuell auflösen, damit andere Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange die Verarbeitung dieser Arbeitseinheit fortsetzen können.

### z/OS *Storage Class Memory mit gemeinsam genutzten Warteschlangen verwenden*

Die Nutzung von Storage Class Memory (SCM) kann in Verbindung mit gemeinsam genutzten Warteschlangen von IBM MQ for z/OS von Vorteil sein.

**Wichtig:** IBM z16 ist als letzte Generation von IBM Z ® geplant, um die Verwendung von virtuellem Flashspeicher (auch bekannt als Storage Class Memory oder SCM) für Coupling-Facility-Images zu unterstützen. Weitere Informationen finden Sie unter IBM Z und IBM LinuxONE 4Q 2023 Statements of Direction.

Alternativ sollten Sie entweder größere Strukturen verwenden oder Nachrichten in SMDS auslagern.

Die Systeme z13, zEC12 und zBC12 ermöglichen die Installation von Flash Express-Karten. Diese Karten enthalten Flash-Solid-State-Laufwerke. Nach der Installation kann Flashspeicher von den Karten einer oder mehreren logischen Partitionen (LPARs) zugeordnet werden, wo er üblicherweise als SCM bekannt ist.

SCM liegt in Bezug auf Ein-/Ausgabelatenzzeit und Ein-/Ausgabeaufwand zwischen Realspeicher und DASD-Einheit (Direct Access Storage Device). Da SCM keine beweglichen Teile hat, weist es viel kürzere Ein-/Ausgabelatenzzeiten auf als DASD.

SCM ist auch viel kostengünstiger als Realspeicher. Deshalb kann zu relativ niedrigen Kosten eine große Speicherkapazität installiert werden; so enthält beispielsweise ein Paar Flash Express-Karten 1424 GB nutzbaren Speicher.

Diese Merkmale bedeuten, dass SCM nützlich ist, wenn einem Realspeicher in einem kurzen Zeitraum ein großes Datenvolumen entnommen werden muss, denn die Daten können viel schneller in SCM als in DASD geschrieben werden. Genau dieser Punkt kann sehr nützlich sein, wenn Coupling-Facility-Listenstrukturen verwendet werden, die gemeinsam genutzte IBM MQ-Warteschlangen enthalten.

## Warum Listenstrukturen voll werden

Bei der Definition einer CF-Struktur wird sie mit dem Attribut SIZE konfiguriert, das die maximale Größe der Struktur angibt. Da CF-Strukturen immer permanent im Realspeicher resident sind, muss die Summe der SIZE-Attribute der für eine CF definierten Strukturen kleiner als die Größe des Realspeichers sein, der der CF zugeordnet ist.

Dies führt dazu, dass es einen ständigen Druck gibt, das SIZE-Attribut für jede vorhandene Struktur auf einen möglichst niedrigen Wert zu setzen, damit die CF mehr Strukturen enthalten kann. Da gleichzeitig sichergestellt werden muss, dass Strukturen groß genug sind, um ihren Zweck zu erfüllen, kommt es zu einem Konflikt, denn eine zu kleine Struktur kann voll werden und dadurch einen Abbruch der Anwendungen und Subsysteme verursachen, die die Struktur nutzen.

Es ist unbedingt erforderlich, dass die Größe einer Struktur ausgehend von ihrer erwarteten Nutzung möglichst genau bestimmt wird. Dies ist jedoch eine schwierige Aufgabe, da sich Arbeitslasten im Laufe der Zeit ändern können und sich deren Schwankungen nur schwer berechnen lassen.

Gemeinsam genutzte IBM MQ-Warteschlangen verwenden CF-Listenstrukturen zum Speichern von Nachrichten. IBM MQ ruft CF-Strukturen auf, die Nachrichten und Anwendungsstrukturen enthalten.

Anwendungen werden über die Informationen referenziert, die in CFSTRUCT-Objekten von IBM MQ gespeichert sind. Wird eine Nachricht, die kleiner als 63 KB ist, in eine gemeinsam genutzte Warteschlange eingereiht, wird die vollständige Nachricht als ein einzelner Listeneintrag in einer Anwendungsstruktur gespeichert, wobei keine oder mehrere Listenelemente vorhanden sein können.

Da gemeinsam genutzte IBM MQ-Warteschlangen Listenstrukturen verwenden, gilt der oben beschriebene Konflikt auch für gemeinsam genutzte Warteschlangen. In diesem Fall ergibt sich die maximale Anzahl Nachrichten, die in einer gemeinsam genutzten Warteschlange gespeichert werden können, aus folgenden Werten:

- Größe der Nachrichten in der Warteschlange
- Maximale Größe der Struktur
- Anzahl der in der Struktur verfügbaren Einträge und Elemente

Dass bis zu 512 gemeinsam genutzte Warteschlangen dieselbe Struktur verwenden können und effektiv um Einträge und Elemente konkurrieren, macht das Ganze noch komplizierter.

Gemeinsam genutzte IBM MQ-Warteschlangen dienen zur Übertragung von Daten zwischen Anwendungen. Eine typische Situation dabei ist, dass eine Anwendung Nachrichten in eine Warteschlange einreiht, während die Partneranwendung, die die Nachrichten erhalten soll, nicht aktiv ist.

Wenn dies der Fall ist, nimmt die Anzahl der Nachrichten in der Warteschlange im Laufe der Zeit zu, bis eine der folgenden Situationen eintritt:

- Die einreihende Anwendung reiht keine weiteren Nachrichten ein.
- Die abrufende Anwendung beginnt damit, Nachrichten abzurufen.
- Vorhandene Nachrichten in der Warteschlange verfallen und werden aus der Warteschlange entfernt.
- Die Warteschlange erreicht ihre maximale Größe, sodass Ursachencode MQRC_Q_FULL an die einreihende Anwendung zurückgegeben wird.
- Die Struktur, die die gemeinsam genutzte Warteschlange enthält, erreicht ihre maximale Größe oder der Coupling-Facility, die die Struktur enthält, steht kein Speicher mehr zur Verfügung. In beiden Fällen wird Ursachencode MQRC_STORAGE_MEDIUM_FULL an die einreihende Anwendung zurückgegeben.

In den drei letztgenannten Situationen ist die Warteschlange voll. An diesem Punkt hat die einreihende Anwendung ein Problem, weil sie ihre Nachrichten nicht mehr übergeben kann. Um dieses Problem zu lösen, ergreift sie normalerweise eine der folgenden Maßnahmen:

- Sie versucht immer wieder, die Nachrichten einzureihen, optional mit einer Verzögerung zwischen den Versuchen.
- Sie legt die Nachrichten woanders ab, z. B. in einer Datenbank oder einer Datei. Auf die Nachrichten kann später zugegriffen werden, um sie ganz normal in die Warteschlange einzureihen.
- Sie löscht Nachrichten, wenn sie nicht persistent sind.

Für einige Anwendungsklassen, z. B. Anwendungen mit sehr vielen eingehenden Nachrichten oder Anwendungen ohne Zugriff auf ein Dateisystem, sind diese Lösungen nicht möglich. Zuallererst sollte deshalb unbedingt sichergestellt werden, dass Warteschlangen nie voll werden oder dies zumindest höchst unwahrscheinlich ist. Dies ist für gemeinsam genutzte Warteschlangen besonders relevant.

## SMDS und Auslagerungsregeln

Die mit IBM WebSphere MQ 7.1 eingeführten Auslagerungsregeln bieten eine Möglichkeit, die Wahrscheinlichkeit zu verringern, dass eine Anwendungsstruktur voll wird.

Jeder Anwendungsstruktur sind drei Regeln zugeordnet, die durch drei Schlüsselwortpaare angegeben werden:

- OFFLD1SZ und OFFLD1TH
- OFFLD2SZ und OFFLD2TH

- OFFLD3SZ und OFFLD3TH

Jede Regel gibt die Bedingungen an, die erfüllt sein müssen, damit Nachrichtendaten in das Speicherverfahren ausgelagert werden, das der Anwendungsstruktur zugeordnet ist. Momentan sind zwei Speicherverfahren verfügbar:

- Db2
- Gruppe linearer VSAM-Dateien (Virtual Storage Access Method), in IBM MQ als Shared Message Data Set (SMDS) bezeichnet

Das folgende Beispiel zeigt den MQSC-Befehl zum Erstellen einer Anwendungsstruktur namens LIST1 unter Verwendung des Befehls DEFINE CFSTRUCT.

Für diese Struktur werden die Standardauslagerungsregeln und SMDS als Auslagerungsverfahren angewendet. Das bedeutet: Wenn die Struktur zu 70% voll ist ( OFFLD1TH ), werden alle Nachrichten ab einer Größe von 32 KB ( OFFLD1SZ ) in SMDS ausgelagert.

Entsprechend gilt: Wenn die Struktur zu 80% voll ist ( OFFLD2TH ), werden alle Nachrichten ab einer Größe von 4 KB ( OFFLD2SZ ) ausgelagert. Wenn die Struktur zu 90% voll ist ( OFFLD3TH ), werden alle Nachrichten ( OFFLD3SZ ) ausgelagert.

```
DEFINE CFSTRUCT(LIST1)
CFLEVEL(5)
OFFLOAD(SMDS)
OFFLD1SZ(32K) OFFLD1TH(70)
OFFLD2SZ(4K) OFFLD2TH(80)
OFFLD3SZ(0K) OFFLD3TH(90)
```

Eine ausgelagerte Nachricht wird auf dem Auslagerungsmedium gespeichert und in der Struktur wird ein Zeiger auf die Nachricht gespeichert. Auch wenn die Auslagerungsregeln die Chance erhöhen, dass die Struktur nicht voll wird, indem sie bei eigenem Speicherengpass weniger Nachrichtendaten in der Struktur ablegen, werden weiterhin für jede Nachricht einige Daten in die Struktur geschrieben. Bei diesen Daten handelt es sich um den Zeiger auf die ausgelagerte Nachricht.

Außerdem sind die Auslagerungsregeln mit Leistungseinbußen verbunden. Das Schreiben einer Nachricht in eine Struktur geht relativ schnell, wobei die meiste Zeit für das Senden der Schreibanforderung an die Coupling-Facility benötigt wird. Der eigentliche Schreibvorgang ist schnell, da er in Realspeichergeschwindigkeit erfolgt.

Das Schreiben einer Nachricht in SMDS dauert wesentlich länger, weil zum einen der Nachrichtenzeiger in die Struktur und zum anderen die Nachrichtendaten in SMDS geschrieben werden müssen. Der zweite Schreibvorgang erfolgt in DASD-Geschwindigkeit und kann deshalb eine zusätzliche Latenzzeit verursachen. Wenn Db2 als Auslagerungsverfahren verwendet wird, sind die Leistungseinbußen noch größer.

z/OS *Storage Class Memory mit IBM MQ for z/OS*

Übersicht über die Verwendung von Storage Class Memory (SCM) mit gemeinsam genutzten IBM MQ for z/OS-Warteschlangen

**Wichtig:** IBM z16 ist als letzte Generation von IBM Z ® geplant, um die Verwendung von virtuellem Flashspeicher (auch bekannt als Storage Class Memory oder SCM) für Coupling-Facility-Images zu unterstützen. Weitere Informationen finden Sie unter IBM Z und IBM LinuxONE 4Q 2023 Statements of Direction.

Alternativ sollten Sie entweder größere Strukturen verwenden oder Nachrichten in SMDS auslagern.

Einer Coupling-Facility (CF) mit CFLEVEL 19 oder höher kann Storage Class Memory (SCM) zugeordnet werden. Die in einer solchen CF definierten Strukturen können dann für die Verwendung von SCM konfiguriert werden, um die Möglichkeit, dass die Strukturen vollständig gefüllt werden, zu verringern. Wenn eine Struktur, die für die Verwendung von SCM konfiguriert ist, bis zu einem vom System festgelegten Punkt gefüllt ist, beginnt die CF, Daten aus der Struktur in SCM zu verschieben, um in der Struktur Speicherplatz für neue Daten freizugeben.

**Anmerkung:** Da SCM selbst auch voll werden kann, verringert die Zuordnung von SCM nur die Wahrscheinlichkeit, dass eine Struktur vollständig gefüllt wird, kann dies also nicht grundsätzlich verhindern.

Eine Struktur wird für die Verwendung von SCM konfiguriert, indem die beiden Schlüsselwörter **SCMALGORITHM** und **SCMMAXSIZE** in der CFRM-Richtlinie (Coupling Facility Resource Manager), die die Definition der betreffenden Struktur enthält, angegeben werden.

Nachdem die Schlüsselwörter angegeben wurden und die CFRM-Richtlinie angewendet wurde, muss die Struktur neu erstellt oder freigegeben werden, damit die Schlüsselwörter wirksam werden können.

## Schlüsselwort SCMALGORITHM

Da die Ein-/Ausgabegeschwindigkeit von SCM langsamer als die eines Realspeichers ist, verwendet die CF einen Algorithmus, der an die erwartete Nutzung der Struktur angepasst wird, um die Auswirkung des Schreibens in bzw. Lesens aus SCM zu verringern.

Der Algorithmus wird mit dem Schlüsselwort **SCMALGORITHM** in der CFRM-Richtlinie für die Struktur konfiguriert, indem dem Schlüsselwort der Wert *KEYPRIORITY1* zugewiesen wird. Der Wert *KEYPRIORITY1* sollte nur für Listenstrukturen angegeben werden, die von gemeinsam genutzten IBM MQ-Warteschlangen verwendet werden.

Der Algorithmus *KEYPRIORITY1* setzt voraus, dass die meisten Anwendungen Nachrichten aus einer gemeinsam genutzten Warteschlange nach Priorität abrufen, eine Anwendung also immer die älteste Nachricht mit der höchsten Priorität abruft.

Sobald eine Struktur den systemdefinierten Füllungsgrad von 90% überschreitet, beginnt die CF, Nachrichten asynchron zu migrieren, bei denen es am unwahrscheinlichsten ist, dass sie als Nächstes abgerufen werden. Das sind Nachrichten mit niedriger Priorität, die zuletzt in die Warteschlange eingereiht wurden.

Diese asynchrone Migration von Nachrichten aus der Struktur in SCM ist als "Vorabspeicherung" bekannt.

Vorabspeicherung verringert die Leistungseinbußen bei der Verwendung von SCM, weil es die Möglichkeit verringert, dass eine Anwendung blockiert wird, während eine synchrone Ein-/Ausgabe in bzw. aus SCM stattfindet.

Zusätzlich zur Vorabspeicherung bringt der Algorithmus *KEYPRIORITY1* Nachrichten auch asynchron aus SCM zurück und in die Struktur, wenn genug freier Speicherplatz verfügbar ist. Für den Algorithmus *KEYPRIORITY1* gilt dies, solange die Struktur noch nicht zu mehr als 70% gefüllt ist.

Das Zurückbringen von Nachrichten aus SCM in die Struktur ist als "Vorabzugriff" bekannt.

Der Vorabzugriff verringert die Wahrscheinlichkeit, dass eine Anwendung versucht, eine Nachricht abzurufen, die vorab in SCM gespeichert wurde, und dass sie warten muss, während die CF die Nachricht synchron in die Struktur zurückbringt.

## Schlüsselwort SCMMAXSIZE

Das Schlüsselwort **SCMMAXSIZE** gibt die maximale SCM-Menge an, die von einer Struktur genutzt werden kann. Da SCM von der CF bei der Anforderung zugeordnet wird, ist es möglich, eine **SCMMAXSIZE** anzugeben, die größer als die Gesamtmenge des verfügbaren freien SCM ist. Dies ist als "Überbelegung" bekannt.

**Wichtig:** SCM darf nicht überbelegt werden. Andernfalls zeigen die Anwendungen, die sich darauf stützen, nicht das erwartete Verhalten. Zum Beispiel empfangen IBM MQ-Anwendungen, die gemeinsam genutzte Warteschlangen verwenden, möglicherweise unerwartete MQRC_STORAGE_MEDIUM_FULL-Ursachencodes.

Die CF überwacht ihre Verwendung von SCM mithilfe verschiedener Datenstrukturen. Diese Datenstrukturen befinden sich im Realspeicher, der der CF zugeordnet ist, und verringern deshalb die Menge des Realspeichern, der von Strukturen genutzt werden kann. Der von den Datenstrukturen belegte Speicher ist als "erweiterter Speicherplatz" bekannt.

Wenn eine Struktur mit SCM konfiguriert ist, wird eine kleine Menge des Realspeichers aus der CF der Struktur zugeordnet (fester erweiterter Speicherplatz). Diese Zuordnung erfolgt auch dann, wenn die Struktur tatsächlich nie SCM nutzt. Werden Daten aus der Struktur in SCM gespeichert, wird zusätzlicher dynamischer erweiterter Speicherplatz aus dem übrigen Realspeicher in der CF zugeordnet.

Wenn die Daten aus SCM entfernt werden, wird der dynamische erweiterte Speicherplatz an die CF zurückgegeben. Erweiterter Speicherplatz, egal ob fest oder dynamisch, wird niemals dem Realspeicher entnommen, der einer Struktur zugeordnet ist.

Wenn eine Struktur für die Verwendung von SCM konfiguriert ist, vergrößert sich neben dem erweiterten Speicher auch die Menge des von der Struktur verwendeten Steuerspeichers. Dies bedeutet, dass eine mit SCM konfigurierte Listenstruktur weniger Einträge und Elemente enthalten kann als eine Struktur derselben Größe, für die kein SCM konfiguriert ist.

Um die Auswirkung von SCM auf neue oder vorhandene Strukturen besser zu verstehen, kann das Tool CFSizer helfen.

Abschließend noch ein wichtiger Hinweis: Nachdem Daten aus der Struktur in SCM verschoben wurden und dynamischer erweiterter Speicherplatz belegt wurde, kann die Struktur nicht mehr geändert werden, weder manuell noch automatisch.

Das heißt, dass die der Struktur zugeordnete Speichermenge nicht erhöht oder verkleinert werden kann, das von der Struktur verwendete Eintrag-Element-Verhältnis nicht geändert werden kann und so weiter. Die Struktur kann erst wieder geändert werden, wenn keine Daten aus der Struktur mehr in SCM gespeichert sind und kein dynamischer erweiterter Speicherplatz mehr belegt wird.

z/OS *Gründe für die Verwendung von SCM*

Notfallspeicher und verbesserte Leistung sind zwei Anwendungsfälle für die Verwendung von SCM mit IBM MQ for z/OS.

**Wichtig:** IBM z16 ist als letzte Generation von IBM Z ® geplant, um die Verwendung von virtuellem Flashspeicher (auch bekannt als Storage Class Memory oder SCM) für Coupling-Facility-Images zu unterstützen. Weitere Informationen finden Sie unter IBM Z und IBM LinuxONE 4Q 2023 Statements of Direction.

Alternativ sollten Sie entweder größere Strukturen verwenden oder Nachrichten in SMDS auslagern.

In diesem Abschnitt wird die Theorie hinter den beiden möglichen Szenarios vorgestellt. Detaillierte Informationen zur Einrichtung der Szenarios finden Sie in folgenden Abschnitten:

- „Notfallspeicher - Basiskonfiguration” auf Seite 213
- „Verbesserte Leistung - Basiskonfiguration” auf Seite 220

**Wichtig:** Die Verwendung von SCM mit CF-Strukturen ist nicht von einer bestimmten Version von IBM MQ abhängig. Das Notfallspeicherszenario funktioniert jedoch nur mit IBM WebSphere MQ 7.1 und höher, weil SMDS und die Auslagerungsregeln erforderlich sind.

## Notfallspeicher

SMDS und Nachrichtenauslagerung können in Verbindung mit SCM verwendet werden, um die Wahrscheinlichkeit zu verringern, dass bei einem längeren Anwendungsausfall der Ursachencode MQRC_STORAGE_MEDIUM_FULL an eine IBM MQ-Anwendung zurückgegeben wird.

### Übersicht

In einer Anwendungsstruktur ist eine einzige gemeinsam genutzte Warteschlange konfiguriert. Die einreihende Anwendung reiht Nachrichten in die gemeinsam genutzte Warteschlange ein; die abrufende Anwendung ruft Nachrichten aus der gemeinsam genutzten Warteschlange ab.

Bei normaler Ausführung wird erwartet, dass die Warteschlangenlänge nahe null liegt, aber eine Geschäftsanforderung verlangt, dass das System in der Lage sein muss, einen zweistündigen Ausfall der abrufenden Anwendung zu tolerieren. Dies bedeutet, dass die gemeinsam genutzte Warteschlange fähig sein muss, in diesen zwei Stunden alle Nachrichten von der einreihenden Anwendung aufzunehmen.

Aktuell wird dies durch die Standardauslagerungsregeln und SMDS erreicht, sodass die Größe der Struktur minimiert und gleichzeitig die mit der Auslagerung verbundenen Leistungseinbußen reduziert werden.

Es wird erwartet, dass sich die Rate der Nachrichten, die an die gemeinsam genutzte Warteschlange gesendet werden, kurz- bis mittelfristig verdoppelt. Obwohl die Anforderung, dass das System in der

Lage sein muss, einen zweistündigen Ausfall zu tolerieren, weiter besteht, ist in der CF nicht genug Realspeicher für eine Verdopplung der Strukturgröße verfügbar.

Da sich die CF, die die Anwendungsstruktur enthält, auf einem zEC12-System befindet, besteht die Möglichkeit, der Struktur ausreichend SCM für die Speicherung so vieler Nachrichten zuzuordnen, dass ein zweistündiger Ausfall toleriert werden kann.

Betrachten Sie folgenden zeitlichen Ablauf:

1. Zu Beginn befindet sich das System in einem stabilen Zustand. Einreihende und abrufende Anwendung arbeiten normal und die Warteschlangenlänge liegt nahe null oder ist null. Das bedeutet, dass die Anwendungsstruktur fast leer ist.
2. Zu einem bestimmten Zeitpunkt wird die abrufende Anwendung aufgrund eines nicht erwarteten Fehlers gestoppt. Die einreihende Anwendung reiht weiter Nachrichten in die Warteschlange ein und die Anwendungsstruktur beginnt sich zu füllen.
3. Sobald die Struktur zu 70% gefüllt ist, sind die Bedingungen der ersten Auslagerungsregel erfüllt und alle Nachrichten ab einer Größe von 32 KB werden in SMDS ausgelagert.

   Im Abschnitt „SMDS und Auslagerungsregeln” auf Seite 207 finden Sie eine Übersicht über die Auslagerungsregeln.
4. Da weiterhin Nachrichten in die gemeinsam genutzte Warteschlange eingereiht werden, wird die Struktur weiter gefüllt (entweder aufgrund der Nachrichtendaten, die in der Struktur gespeichert werden, oder infolge der in der Struktur gespeicherten Zeiger auf die ausgelagerten Nachrichten).

   Sobald die Struktur zu 80% gefüllt ist, greift die zweite Auslagerungsregel und Nachrichten ab einer Größe von 4 KB werden in SMDS ausgelagert.
5. Sobald die Struktur zu mehr als 90% gefüllt ist, werden alle Nachrichten in SMDS ausgelagert und nur die Nachrichtenzeiger in der Struktur gespeichert.

   Etwa zu diesem Zeitpunkt wird der Vorabspeicherungsalgorithmus gestartet und beginnt damit, Daten aus der Struktur in SCM zu verschieben. Vorausgesetzt, dass alle Nachrichten in der Warteschlange die gleiche Priorität haben, erfolgt eine Vorabspeicherung der neuesten Nachrichten.

   Da jetzt alle Nachrichten in SMDS ausgelagert werden, handelt es sich bei den Daten, die in SCM verschoben werden, nicht um tatsächliche Nachrichtendaten, sondern um die Zeiger auf die Nachrichten in SMDS.

   Dies führt dazu, dass die Anzahl der Nachrichten, die in der Struktur und dem ihr zugeordneten SCM und SMDS insgesamt gespeichert werden können, sehr groß ist.

   **Leistung:** In dieser Phase des Anwendungsausfalls verzeichnet die einreihende Anwendung möglicherweise Leistungseinbußen aufgrund der erforderlichen Schreibvorgänge in SMDS. In diesem Fall sollte die Verwendung von SCM kein limitierender Faktor für die einreihende Anwendung hinsichtlich der Leistung sein. SCM stellt zusätzlichen Speicherplatz bereit, um zu verhindern, dass die Struktur voll wird.
6. Die abrufende Anwendung ist wieder verfügbar und der Ausfall ist beendet.

   Die Struktur verwendet jedoch weiterhin SCM. Die abrufende Anwendung beginnt damit, Nachrichten aus der Warteschlange zu lesen, wobei sie zuerst die ältesten Nachrichten mit der höchsten Priorität abruft.

   Da diese Nachrichten geschrieben wurden, bevor sich die Struktur zu füllen begann, kommen sie vollständig aus dem Realspeicheranteil der Struktur.
7. Die Struktur beginnt sich zu leeren und unterschreitet irgendwann die Schwelle für die Aktivierung des Vorabspeicherungsalgorithmus, sodass er gestoppt wird.
8. Die Strukturbelegung sinkt unter den Punkt, an dem die Auslagerungsregeln wirksam werden, sodass Nachrichten nicht mehr in SMDS ausgelagert werden, außer wenn sie größer als 63 KB sind.

   Etwa zu diesem Zeitpunkt beginnt der Vorabzugriffsalgorithmus damit, Daten aus SCM in die Struktur zu verschieben. Da die abrufende Anwendung Nachrichten in der von den SCM-Algorithmen erwar-

teten Reihenfolge aus der Warteschlange abruft, werden die Nachrichten bereitgestellt, bevor die abrufende Anwendung sie benötigt.

Dies bedeutet, dass die abrufende Anwendung nie darauf warten muss, dass Nachrichten synchron aus SCM bereitgestellt werden.

9. Die abrufende Anwendung arbeitet die Warteschlange weiter ab und beginnt damit, auch die in SMDS ausgelagerten Nachrichten abzurufen.
10. Das System befindet sich wieder in einem stabilen Zustand. Es werden keine Nachrichten in SCM oder SMDS gespeichert und die Warteschlangenlänge liegt nahe null.

## Verbesserte Leistung

Dieses Szenario beschreibt die Verwendung von SCM zur Erhöhung der Anzahl Nachrichten, die in einer gemeinsam genutzten Warteschlange gespeichert werden können, ohne dass es zu Leistungseinbußen bei der Verwendung von SMDS kommt.

**Beschreibung**

In diesem Szenario kommunizieren eine einreihende und eine abrufende Anwendung über eine gemeinsam genutzte Warteschlange, die in einer Anwendungsstruktur gespeichert ist.

Die einreihende Anwendung neigt dazu, in den Burst-Modus zu wechseln, wenn sie sehr viele Nachrichten in kurzer Zeit einreiht. Sie produziert dann über einen längeren Zeitraum überhaupt keine Nachrichten.

Die abrufende Anwendung verarbeitet Nachrichten sequenziell, wobei für jede Nachricht eine komplexe Verarbeitung stattfindet. Dies führt dazu, dass die Warteschlangenlänge die meiste Zeit bei null liegt, außer zu dem Zeitpunkt, an dem die einreihende Anwendung ihre Arbeit aufnimmt und die Warteschlangenlänge zunimmt, weil Nachrichten schneller eingereiht als abgerufen werden.

Die Warteschlangenlänge nimmt zu, bis die einreihende Anwendung gestoppt wird und die abrufenden Anwendung genug Zeit hat, die Nachrichten in der Warteschlange zu verarbeiten.

**Anmerkungen:**

1. In diesem Szenario ist die Leistung der Schlüsselfaktor. Die Nachrichten, die an die Warteschlange gesendet werden, sind immer kleiner als 63 KB und müssen deshalb nie in SMDS ausgelagert werden.
2. Die Anwendungsstruktur wurde so dimensioniert, dass sie groß genug ist, um alle Nachrichten aufzunehmen, die von der einreihenden Anwendung in einem einzigen "Block" platziert werden.
3. Alle Auslagerungsregeln müssen inaktiviert sein, sodass die Nachrichten nicht einmal dann in SMDS ausgelagert werden, wenn die Struktur voll ist. Der Grund dafür ist, dass die Leistungseinbußen, die mit dem Schreiben von Nachrichten in und Lesen von Nachrichten aus SMDS verbunden sind, als nicht akzeptabel betrachtet werden.

Im Laufe der Zeit muss die Anzahl Nachrichten, die die einreihende Anwendung in einem Burst sendet, um mehrere Größenordnungen zunehmen. Da die abrufende Anwendung die Nachrichten sequenziell verarbeiten muss, steigt die Anzahl Nachrichten in der Warteschlange bis zu dem Punkt, an dem die Struktur voll ist.

Ist dieser Punkt erreicht, empfängt die einreihende Anwendung einen Ursachencode (MQRC_STORAGE_MEDIUM_FULL), wenn sie eine Nachricht einreiht, und die Einreihungsoperation schlägt fehl. Die einreihende Anwendung kann Zeiträume, in denen sie keine Nachrichten in die Warteschlange einreihen kann, nur kurze Zeit tolerieren. Ist der Zeitraum zu lang, wird die Anwendung beendet.

Sofern Sie nicht die Zeit haben oder nicht über die erforderlichen Kenntnisse verfügen, um entweder die einreihende oder die abrufende Anwendung neu zu programmieren, gibt es drei mögliche Lösungen für dieses Problem:

1. Vergrößern Sie die Anwendungsstruktur.
2. Fügen Sie Auslagerungsregeln zur Anwendungsstruktur hinzu, sodass Nachrichten in SMDS ausgelagert werden, sobald die Warteschlange voll wird.
3. Ordnen Sie SCM der Struktur zu.

Die erste Lösung lässt sich schnell umsetzen, aber es ist nicht genug Realspeicher in der CF verfügbar.

Auch die zweite Lösung lässt sich möglicherweise schnell umsetzen, aber die zu starken Leistungseinbußen durch die Auslagerung in SMDS sprechen gegen diese Option.

Die dritte Lösung, also die Zuordnung von SCM zur Struktur, bietet einen akzeptablen Ausgleich zwischen Aufwand und Leistung.

Die Zuordnung von SCM zu einer Struktur führt aufgrund des erweiterten Speichers, den Abrufoperationen verwenden, zu einer stärkeren Nutzung von Realspeicher in der CF. Die tatsächliche Realspeichergröße ist jedoch geringer als bei der ersten Option.

Ein anderer Aspekt sind die Kosten von SCM. Diese Kosten liegen jedoch unter denen für Realspeicher. Die Kombination dieser Faktoren macht die dritte Option kostengünstiger als die erste Option.

Obwohl die dritte Option möglicherweise nicht die gleiche Leistung wie die erste Option bietet, können die von der CF verwendeten Vorabzugriffs- und Vorabspeicherungsalgorithmen gemeinsam bewirken, dass die Leistungsunterschiede akzeptabel sind, oder in einigen Fällen zumindest vernachlässigbar.

Auf jeden Fall kann die Leistung viel besser sein als bei der Verwendung von SMDS zum Auslagern von Nachrichten.

Betrachten Sie folgenden zeitlichen Ablauf:

1. Zu Beginn ist die abrufende Anwendung aktiv und wartet darauf, dass Nachrichten in die gemeinsam genutzte Warteschlange gestellt werden. Die einreihende Anwendung ist nicht aktiv und die gemeinsam genutzte Warteschlange ist leer.
2. Zu einem bestimmten Zeitpunkt wird die einreihende Anwendung aktiv und beginnt damit, sehr viele Nachrichten in die gemeinsam genutzte Warteschlange einzureihen. Die abrufende Anwendung beginnt damit, Nachrichten abzurufen, aber die Warteschlangenlänge nimmt schnell zu, weil die abrufende Anwendung langsamer als die einreihende Anwendung ist.

   Dies führt dazu, dass die Anwendungsstruktur sich zu füllen beginnt.
3. Die einreihende Anwendung bleibt aktiv. Die Anwendungsstruktur wird zu etwa 90% gefüllt.

   In diesem Moment beginnt der SCM-Vorabspeicherungsalgorithmus damit, Nachrichten aus der Struktur in SCM zu verschieben und so Speicherplatz in der Struktur freizugeben.

   Da die abrufende Anwendung zuerst die ältesten Nachrichten mit der höchsten Priorität aus der Warteschlange abruft, erhält sie immer Nachrichten aus der Struktur und muss nicht darauf warten, dass Nachrichten synchron aus SCM in der Struktur bereitgestellt werden.
4. Die einreihende Anwendung ist weiter aktiv und reiht Nachrichten in die gemeinsam genutzte Warteschlange ein. Die Anwendung empfängt jedoch nie den Ursachencode MQRC_STORAGE_MEDIUM_FULL, weil in SCM genug Speicherplatz vorhanden ist, um alle Nachrichten zu speichern, für die in der Struktur kein Platz mehr verfügbar ist.
5. Die einreihende Anwendung wird gestoppt, weil keine weiteren Nachrichten eingereiht werden müssen.

   Der Vorabspeicherungsalgorithmus wird gestoppt, weil der Füllungsgrad der Struktur unter 90% sinkt, und die abrufende Anwendung setzt die Verarbeitung der Nachrichten in der Warteschlange fort.
6. Da die abrufende Anwendung nach und nach Speicherplatz in der Struktur freigibt, beginnt der Vorabzugriffsalgorithmus damit, Nachrichten aus SCM zurück in die Struktur zu verschieben.

   Weil die abrufende Anwendung Nachrichten in der vom Vorabzugriffsalgorithmus erwarteten Reihenfolge verarbeitet, wird sie nie dadurch blockiert, dass sie auf die synchrone Bereitstellung von Nachrichtendaten aus SCM in der Struktur warten muss.
7. Die abrufende Anwendung verarbeitet alle Nachrichten in der gemeinsam genutzten Warteschlange und wartet dann, bis die nächste Nachricht verfügbar ist. Die Struktur und SCM enthalten keine Nachrichten mehr.

z/OS *Notfallspeicher - Basiskonfiguration*
Einrichten eines Basisszenarios für Notfallspeicher in IBM MQ

## Informationen zu diesem Vorgang

**Wichtig:** IBM z16 ist als letzte Generation von IBM Z ® geplant, um die Verwendung von virtuellem Flashspeicher (auch bekannt als Storage Class Memory oder SCM) für Coupling-Facility-Images zu unterstützen. Weitere Informationen finden Sie unter IBM Z und IBM LinuxONE 4Q 2023 Statements of Direction.

Alternativ sollten Sie entweder größere Strukturen verwenden oder Nachrichten in SMDS auslagern.

SMDS und Nachrichtenauslagerung können in Verbindung mit SCM verwendet werden, um die Wahrscheinlichkeit zu verringern, dass bei einem längeren Anwendungsausfall der Ursachencode MQRC_STORAGE_MEDIUM_FULL an eine IBM MQ-Anwendung zurückgegeben wird.

Beispiel: In Ihrem Unternehmen gibt es eine Anwendung, die Nachrichten in die Warteschlange einreiht, und eine Anwendung, die Nachrichten aus der Warteschlange abruft. Bei normaler Ausführung wird erwartet, dass die Warteschlangenlänge nahe null liegt, aber eine Geschäftsanforderung verlangt, dass das System in der Lage sein muss, einen zweistündigen Ausfall der Anwendung, die die Nachrichten abruft, zu tolerieren.

Dies bedeutet, dass die verwendete gemeinsam genutzte Warteschlange fähig sein muss, in diesen zwei Stunden alle Nachrichten von der einreihenden Anwendung aufzunehmen. Aktuell erreichen Sie dies mithilfe der Standardauslagerungsregeln und von SMDS.

Sie erwarten, dass sich die Rate der Nachrichten, die an die gemeinsam genutzte Warteschlange gesendet werden, kurz- bis mittelfristig verdoppelt. Obwohl Ihre Anforderung, dass das System in der Lage sein muss, einen zweistündigen Ausfall zu tolerieren, weiter besteht, ist in der CF nicht genug Realspeicher für eine Verdopplung der Strukturgröße verfügbar. Da sich die CF, die die Anwendungsstruktur enthält, auf einem zEC12-System befindet, haben Sie die Möglichkeit, der Struktur ausreichend SCM für die Speicherung so vieler Nachrichten zuzuordnen, dass ein zweistündiger Ausfall toleriert werden kann.

In diesem Ausgangsszenario wird Folgendes verwendet:

- Eine Gruppe mit gemeinsamer Warteschlange (`IBM1`), die einen einzelnen Warteschlangenmanager (`CSQ3`) enthält. Neben der Verwaltungsstruktur ist für die Gruppe mit gemeinsamer Warteschlange auch eine einzelne Anwendungsstruktur (`SCEN1`) definiert.
- Eine Coupling-Facility (`CF01`), in der die Anwendungsstruktur `SCEN1` als die Struktur `IBM1SCEN1` gespeichert ist. Diese Struktur hat eine maximale Größe von 1 GB.
- Eine einzelne gemeinsam genutzte Warteschlange (`SCEN1.Q`), die von der Anwendungsstruktur verwendet wird.

Diese Konfiguration wird in Abbildung 61 auf Seite 214 dargestellt.

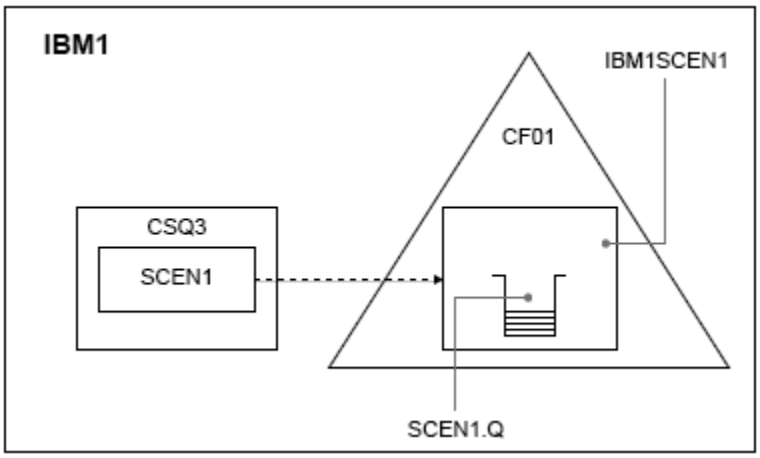



*Abbildung 61. Basiskonfiguration*

Darüber hinaus wird vorausgesetzt, dass Warteschlangenmanager `CSQ3` bereits das einzige Mitglied von `IBM1`, der Gruppe mit gemeinsamer Warteschlange, ist.

Sie müssen die Definition für die Struktur IBM1SCEN1 zur CFRM-Richtlinie (Coupling Facility Resource Manager) hinzufügen. Zur Vereinfachung ist die Struktur so definiert, dass sie nur in einer einzigen Coupling-Facility (CF01) erstellt werden kann, indem PREFLIST(CF01) angegeben wird.

**Achtung:** Um hohe Verfügbarkeit in Ihrem Produktionssystem zu gewährleisten, sollten Sie mindestens zwei CFs in die PREFLIST-Anweisung für alle Strukturen, die von IBM MQ verwendet werden, einschließen.

## Vorgehensweise

1. Aktualisieren Sie die CFRM-Richtlinie mit folgendem Befehl:

```
SETXCF START,POLICY,TYPE=CFRM,POLNAME=IBM1SCEN1
```

   CFRM-Beispielrichtlinie für Struktur IBM1SCEN1:

```
STRUCTURE
NAME(IBM1SCEN1)
SIZE(1024M)
INITSIZE(512M)
ALLOWAUTOALT(YES)
FULLTHRESHOLD(85)
PREFLIST(CF01)
ALLOWREALLOCATE(YES)
DUPLEX(DISABLED)
ENFORCEORDER(NO)
```

2. Überprüfen Sie mit folgendem Befehl, ob die Struktur ordnungsgemäß erstellt wurde:

```
D XCF,STR,STRNAME=IBM1SCEN1
```

   Zu diesem Zeitpunkt ist die Struktur der Gruppe mit gemeinsamer Warteschlange noch nicht zugeordnet (siehe Zeile STATUS).
3. Konfigurieren Sie IBM MQ für die Verwendung der in der CFRM-Richtlinie definierten Struktur.
   a. Erstellen Sie mit dem Befehl DEFINE CFSTRUCT unter Angabe des Strukturnamens SCEN1 ein IBM MQ-CFSTRUCT-Objekt:

```
DEFINE CFSTRUCT(SCEN1)
CFCONLOS(TOLERATE)
CFLEVEL(5)
DESCR('Structure for SCM scenario 1')
RECOVER(NO)
RECAUTO(YES)
OFFLOAD(DB2)
OFFLD1SZ(64K) OFFLD1TH(70)
OFFLD2SZ(64K) OFFLD2TH(80)
OFFLD3SZ(64K) OFFLD3TH(90)
```

      .
   b. Überprüfen Sie die Struktur mit dem Befehl DISPLAY CFSTRUCT.
   c. Weisen Sie mit folgendem MQSC-Befehl die gemeinsam genutzte Warteschlange SCEN1.Q an, die Struktur SCEN1 zu verwenden:

```
DEFINE QLOCAL(SCEN1.Q) QSGDISP(SHARED) CFSTRUCT(SCEN1) MAXDEPTH(999999999)
```

4. Reihen Sie mithilfe von IBM MQ Explorer eine einzelne Nachricht in die Warteschlange SCEN1.Q ein und rufen Sie sie wieder daraus ab.
5. Überprüfen Sie mit folgendem Befehl, dass die Struktur jetzt zugeordnet ist:

```
D XCF,STR,STRNAME=IBM1SCEN1
```

Überprüfen Sie in der Ausgabe des Befehls, ob die STATUS-Linie ALLOCATED anzeigt.

## Ergebnisse

Damit haben Sie die Basiskonfiguration erstellt. Sie können jetzt eine beliebige Methode wählen, um eine Vorstellung von der grundsätzlichen Leistung Ihrer Konfiguration zu erhalten.

## Nächste Schritte

SMDS und SCM der Ausgangsstruktur hinzufügen

**Zugehörige Konzepte**

„Storage Class Memory mit gemeinsam genutzten Warteschlangen verwenden” auf Seite 206
Die Nutzung von Storage Class Memory (SCM) kann in Verbindung mit gemeinsam genutzten Warteschlangen von IBM MQ for z/OS von Vorteil sein.

z/OS *SMDS und SCM zur Ausgangsstruktur hinzufügen*

Hinzufügen von SMDS und SCM für Notfallspeicher in IBM MQ

## Informationen zu diesem Vorgang

**Wichtig:** IBM z16 ist als letzte Generation von IBM Z ® geplant, um die Verwendung von virtuellem Flashspeicher (auch bekannt als Storage Class Memory oder SCM) für Coupling-Facility-Images zu unterstützen. Weitere Informationen finden Sie unter IBM Z und IBM LinuxONE 4Q 2023 Statements of Direction.

Alternativ sollten Sie entweder größere Strukturen verwenden oder Nachrichten in SMDS auslagern.

Für diesen Teil der Aufgabe wird die im Abschnitt „Notfallspeicher - Basiskonfiguration” auf Seite 213 beschriebene Basiskonfiguration verwendet. Das Szenario beschreibt die Hinzufügung von Shared Message Data Sets (SMDS) und anschließend von SCM zur Ausgangsstruktur.

Die Endkonfiguration wird in Abbildung 62 auf Seite 216 dargestellt.

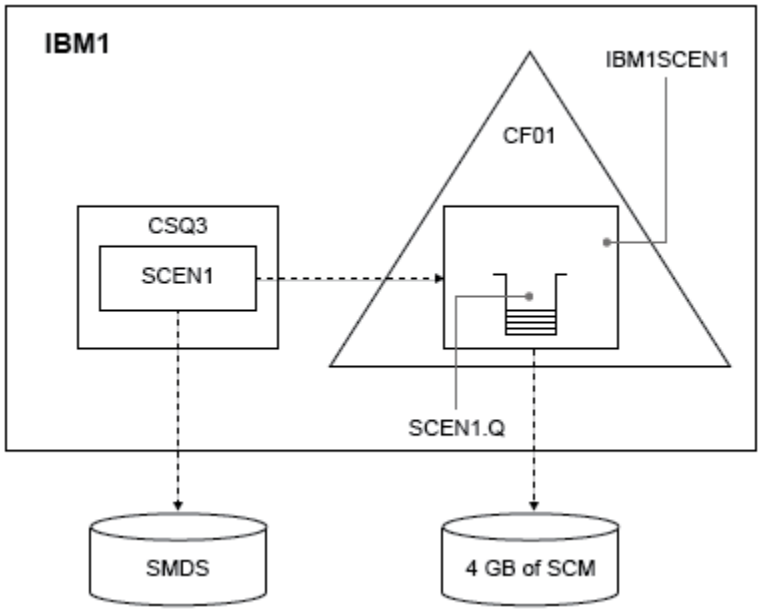



*Abbildung 62. Konfiguration mit Hinzufügung von SMDS und SCM für Notfallspeicher*

## Vorgehensweise

1. Erstellen Sie die von der Anwendungsstruktur SCEN1 verwendeten SMDS-Dateien, indem Sie den JCL-Beispielcode **CSQ4SMDS** wie gezeigt bearbeiten:

```
//CSQ4SMDS JOB NOTIFY=&SYSUID
//*
//* Allocate SMDS
//*
//DEFINE EXEC PGM=IDCAMS,REGION=4M
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
DEFINE CLUSTER           -
(NAME(CSQSMDS.SCEN1.CSQ3.SMDS) -
MEGABYTES(5000 3000)     -
LINEAR             -
SHAREOPTIONS(2 3) )      -
DATA                -
(NAME(CSQSMDS.SCEN1.CSQ3.SMDS.DATA) )
/*
//*
//* Format the SMDS
//*
//FORM   EXEC PGM=CSQJUFMT,COND=(0,NE),REGION=0M
//STEPLIB DD DSN=MQ800.SCSQANLE,DISP=SHR
//     DD DSN=MQ800.SCSQAUTH,DISP=SHR
//SYSUT1  DD DISP=OLD,DSN=CSQSMDS.SCEN1.CSQ3.SMDS
//SYSPRINT DD SYSOUT=*
```

2. Geben Sie den Befehl ALTER CFSTRUCT aus, um die Anwendungsstruktur SCEN1 so zu ändern, dass sie SMDS zur Auslagerung verwendet, indem die Standardauslagerungsregeln implementiert werden:

```
ALTER CFSTRUCT(SCEN1) OFFLOAD(SMDS) OFFLD1SZ(32K) OFFLD2SZ(4K) OFFLD3SZ(0K)
DSGROUP('CSQSMDS.SCEN1.*.SMDS') DSBLOCK(1M)
```

   Beachten Sie Folgendes:

   - Da SCEN1.Q die einzige gemeinsam genutzte Warteschlange in der Anwendungsstruktur SCEN1 ist, wurde **DSBLOCK** auf den Wert '1M' und damit auf den größtmöglichen Wert gesetzt. Dies sollte die effizienteste Einstellung für das Szenario sein.
   - Da die von der einreihenden Anwendung gesendeten Nachrichten 30 KB groß sind, beginnt die Auslagerung in SMDS erst, wenn die zweite Auslagerungsregel greift, die Struktur also zu 80% gefüllt ist.

3. Führen Sie die Testanwendung erneut aus.

   Beachten Sie die erhöhte Speicherung von Nachrichten in der Warteschlange.

4. Fügen Sie SCM mit einer Größe von 4 GB zur Struktur IBM1SCEN1 hinzu, indem Sie wie folgt vorgehen:

   a) Überprüfen Sie mit folgendem Befehl, wie viel SCM installiert und CF01 zugeordnet ist:

   ```
   D CF,CFNAME=CF01
   ```

   b) Überprüfen Sie die STORAGE-CLASS MEMORY -Abbildungen im Abschnitt STORAGE CONFIGURATION der angezeigten Ausgabe, um den verfügbaren Speicher anzuzeigen.

   c) Aktualisieren Sie die CFRM-Richtlinie wie folgt mit den Schlüsselwörtern SCMMAXSIZE und SCMALGORITHM:

   ```
   STRUCTURE
   NAME(IBM1SCEN1)
   SIZE(1024M)
   INITSIZE(512M)
   ALLOWAUTOALT(YES)
   FULLTHRESHOLD(85)
   PREFLIST(CF01)
   ALLOWREALLOCATE(YES)
   DUPLEX(DISABLED)
   ENFORCEORDER(NO)
   SCMMAXSIZE(4G)
   SCMALGORITHM(KEYPRIORITY1)
   ```

5. Aktivieren Sie die CFRM-Richtlinie mit folgendem Befehl:

```
SETXCF START,POLICY,TYPE=CFRM,POLNAME=polname
```

6. Erstellen Sie die Struktur IBM1SCEN1 neu.

   Dies ist erforderlich, weil die Struktur zugeordnet war, als Sie die obigen Änderungen vorgenommen haben.

   Geben Sie zur erneuten Erstellung der Struktur folgenden Befehl aus:

```
SETXCF START,REBUILD,STRNM=IBM1SCEN1
```

## Ergebnisse

Sie haben erfolgreich SCM zur Konfiguration hinzugefügt.

## Nächste Schritte

Optimieren Sie die Leistung des Systems. Weitere Informationen finden Sie unter „Verwendung von Storage Class Memory optimieren” auf Seite 218.

z/OS *Verwendung von Storage Class Memory optimieren*

Verwendung von Storage Class Memory (SCM) verbessern

**Wichtig:** IBM z16 ist als letzte Generation von IBM Z ® geplant, um die Verwendung von virtuellem Flashspeicher (auch bekannt als Storage Class Memory oder SCM) für Coupling-Facility-Images zu unterstützen. Weitere Informationen finden Sie unter IBM Z und IBM LinuxONE 4Q 2023 Statements of Direction.

Alternativ sollten Sie entweder größere Strukturen verwenden oder Nachrichten in SMDS auslagern.

Führen Sie den folgenden Befehl aus:

```
D XCF,STR,STRNAME=IBM1SCEN1
```

Da die Struktur aufgrund der vorhergehenden Tests bereits mit Nachrichtendaten gefüllt war, werden bei der Neuerstellung einige Nachrichten aus der Struktur vorab in SCM gespeichert. Dieser Prozess wurde durch den oben genannten Befehl eingeleitet.

Die Ausgabe dieses Befehls sieht wie folgt aus:

```
ACTIVE STRUCTURE
----------------
ALLOCATION TIME: 06/17/2014 09:28:50
CFNAME : CF01
COUPLING FACILITY: 002827.IBM.02.00000000B8D7
PARTITION: 3B CPCID: 00
STORAGE CONFIGURATION ALLOCATED MAXIMUM %
ACTUAL SIZE: 1024 M 1024 M 100
AUGMENTED SPACE: 3 M 142 M 2
STORAGE-CLASS MEMORY: 88 M 4096 M 2
ENTRIES: 120120 1089536 11
ELEMENTS: 240240 15664556 1
SPACE USAGE IN-USE TOTAL %
ENTRIES: 84921 219439 38
ELEMENTS: 2707678 3149050 85
EMCS: 2 282044 0
LOCKS: 1024
SCMHIGHTHRESHOLD : 90
SCMLOWTHRESHOLD : 70
ACTUAL SUBNOTIFYDELAY: 5000
PHYSICAL VERSION: CD5186A0 2BD8B85C
LOGICAL VERSION: CD515C50 CE2ED258
SYSTEM-MANAGED PROCESS LEVEL: 9
XCF GRPNAME : IXCLO053
DISPOSITION : KEEP
ACCESS TIME : NOLIMIT
MAX CONNECTIONS: 32
# CONNECTIONS : 1
CONNECTION NAME ID VERSION  SYSNAME JOBNAME  ASID STATE
```

```
--------------- -- -------- -------- -------- ---- ----------------
CSQEIBM1CSQ301  01 00010059 SC61    CSQ3MSTR 0091 ACTIVE
```

Die Ausgabe des Befehls liefert folgende Informationen:

- Der Wert von STORAGE_CLASS MEMORY bestätigt, dass SCM mit einer maximalen Größe (**MAXIMUM**) von 4096 MB zur Struktur hinzugefügt wurde.
- Die ALLOCATED -Zahl für die Menge von STORAGE-CLASS MEMORY , die für die Vorabbereitstellung verwendet wird Es gibt jetzt freien Speicherplatz in der Struktur, wo vor der Hinzufügung von SCM keiner war.
- AUGMENTED SPACE gibt den Speicherplatz an, der zur Überwachung der SCM-Nutzung verwendet wird.
- Der Vorabspeicherungsalgorithmus beginnt damit, Daten aus der Struktur in SCM zu verschieben, sobald die Struktur zu 90% gefüllt ist. Dies wird durch die nicht konfigurierbare Eigenschaft **SCMHIGHTHRESHOLD** angegeben.
- Der Vorabzugriffsalgorithmus beginnt damit, Daten aus SCM in die Struktur zu verschieben, sobald die Struktur zu 70% gefüllt ist. Dies wird durch die nicht konfigurierbare Eigenschaft **SCMLOWTHRESHOLD** angegeben.

Sie können jetzt verschiedene Möglichkeiten zur Optimierung der Verwendung von SCM testen. Beachten Sie Folgendes:

- Sobald Nachrichten in SCM gespeichert sind, kann die Struktur nur geändert werden, nachdem alle Daten aus SCM entfernt wurden.

  Das bedeutet in diesem Fall, dass das Eintrag-Element-Verhältnis fest auf den Wert eingestellt wird, der bei der ersten SCM-Verwendung gültig war. Deshalb müssen Sie sicherstellen, dass sich die Struktur in dem von Ihnen gewünschten Zustand befindet, bevor der Vorabspeicherungsalgorithmus beginnt, Daten in SCM zu verschieben.
- Ist die aktuelle Strukturgröße richtig eingestellt, bevor SCM verwendet wird?

  Haben Sie beispielsweise den Wert von **INITSIZE** von 512 MB auf eine Größe (SIZE) von 1 GB erhöht?

  Falls nicht, ist es möglich, dass der Vorabspeicherungsalgorithmus trotz Aktivierung der Struktur für automatische Änderung damit beginnt, Daten in SCM zu verschieben, bevor die Änderung gestartet werden kann. Dies führt dazu, dass die Struktur mit einer Realspeichergröße von 512 MB eingefroren wird.
- Ist das Eintrag-Element-Verhältnis richtig eingestellt, bevor SCM verwendet wird?

  Ziel dieses Szenarios ist es, die Anzahl ausgelagerter Nachrichtenzeiger, die in der Struktur und in SCM als Ganzes gespeichert werden können, zu erhöhen und gleichzeitig so viele Nachrichten wie möglich vollständig im Strukturspeicher zu behalten. Auf diese Nachrichten kann schneller zugegriffen werden als auf Nachrichten in SMDS.

  Deshalb benötigen Sie eine Struktur, die mit einem Eintrag-Element-Verhältnis startet, das gut für die Speicherung von Nachrichten ist, und dann vor dem ersten Start des Vorabspeicherungsalgorithmus zu einem Verhältnis übergeht, das gut für die Speicherung von Nachrichtenzeigern ist. Dieser Übergang kann teilweise mithilfe der IBM MQ-Auslagerungsregeln erreicht werden.

Ändern Sie die Auslagerungsregeln, indem Sie folgenden Befehl ausgeben:

```
ALTER CFSTRUCT(SCEN1) OFFLD1SZ(0K)
```

Sie müssen den Befehl möglicherweise mehrfach ausführen, um die Eintrag-Element-Verhältnisse zu optimieren.

Die folgende Tabelle zeigt mögliche Verbesserungen bezüglich der Anzahl der Nachrichten, die während der verschiedenen Phasen des Notfallspeicherszenarios in die Warteschlange eingereiht werden.

| *Tabelle 18. Vergleich der Ergebnisse für das Notfallspeicherszenario* | | |
|---|---|---|
| **Testbeschreibung** | **Anzahl der Nachrichten** | **Zeit zum Füllen der Warteschlange (Sekunden)** |
| Basiskonfiguration | 27.850 | 3.2 |
| SMDS mit Standardauslagerungsregeln | 205.000 | 158 |
| SCM mit Standardauslagerungsregeln | 828.610 | 469 |
| SCM mit angepassten Auslagerungsregeln | 1.135.775 | 679 |

Die letzte Zeile der Tabelle zeigt, dass die Anpassung der Auslagerungsregeln die gewünschte Wirkung hatte.

Sie müssen Ihr System daraufhin überprüfen, ob Sie es hinsichtlich dieser Zahlen irgendwie verbessern können. Es kann beispielsweise sein, dass der verfügbare SMDS-Speicher zu klein wird. Falls Sie mehr SMDS-Speicher zuordnen können, sollte es möglich sein, die Anzahl Nachrichten in der Warteschlange deutlich zu erhöhen.

z/OS *Verbesserte Leistung - Basiskonfiguration*

Einrichten eines Basisszenarios für verbesserte Leistung in IBM MQ

## Informationen zu diesem Vorgang

**Wichtig:** IBM z16 ist als letzte Generation von IBM Z ® geplant, um die Verwendung von virtuellem Flashspeicher (auch bekannt als Storage Class Memory oder SCM) für Coupling-Facility-Images zu unterstützen. Weitere Informationen finden Sie unter IBM Z und IBM LinuxONE 4Q 2023 Statements of Direction.

Alternativ sollten Sie entweder größere Strukturen verwenden oder Nachrichten in SMDS auslagern.

Dieses Szenario beschreibt die Verwendung von SCM zur Erhöhung der Anzahl Nachrichten, die in einer gemeinsam genutzten Warteschlange gespeichert werden können, ohne dass es zu Leistungseinbußen bei der Verwendung von SMDS kommt.

Dieses Ausgangsszenario ähnelt sehr dem für den Notfallspeicher. Es enthält folgende Elemente:

- Eine Gruppe mit gemeinsamer Warteschlange (`IBM1`), die einen einzelnen Warteschlangenmanager (`CSQ3`) enthält. Neben der Verwaltungsstruktur ist für die Gruppe mit gemeinsamer Warteschlange auch eine einzelne Anwendungsstruktur (`SCEN2`) definiert.
- Eine Coupling-Facility (`CF01`), in der die Anwendungsstruktur `SCEN2` als die Struktur `IBM1SCEN2` gespeichert ist. Diese Struktur hat eine maximale Größe von 2 GB.
- Eine einzelne gemeinsam genutzte Warteschlange (`SCEN2.Q`), die für die Verwendung der Anwendungsstruktur konfiguriert ist.

Diese Konfiguration wird in Abbildung 63 auf Seite 221 dargestellt.

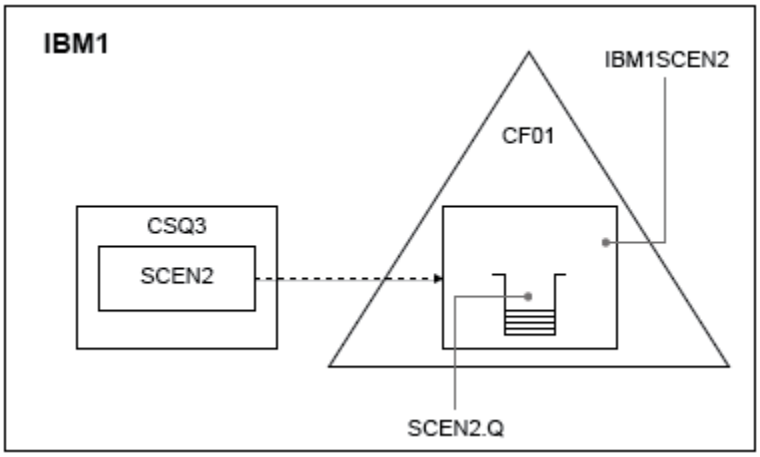



*Abbildung 63. Basiskonfiguration*

Darüber hinaus wird vorausgesetzt, dass Warteschlangenmanager CSQ3 bereits das einzige Mitglied von IBM1, der Gruppe mit gemeinsamer Warteschlange, ist.

Sie müssen die Definition für die Struktur IBM1SCEN2 zur CFRM-Richtlinie (Coupling Facility Resource Manager) hinzufügen. Zur Vereinfachung ist die Struktur so definiert, dass sie nur in einer einzigen Coupling-Facility (CF01) erstellt werden kann, indem PREFLIST(CF01) angegeben wird.

CFRM-Beispielrichtlinie für Struktur IBM1SCEN2:

```
STRUCTURE
NAME(IBM1SCEN2)
SIZE(2048M)
INITSIZE(2048M)
ALLOWAUTOALT(YES)
FULLTHRESHOLD(85)
PREFLIST(CF01)
ALLOWREALLOCATE(YES)
DUPLEX(DISABLED)
ENFORCEORDER(NO)
```

Sowohl das Schlüsselwort **INITSIZE** als auch das Schlüsselwort **SIZE** haben den Wert 2048M, sodass die Größe der Struktur nicht geändert werden kann.

## Vorgehensweise

1. Aktualisieren Sie die CFRM-Richtlinie mit folgendem Befehl:

   ```
   SETXCF START,POLICY,TYPE=CFRM,POLNAME=IBM1SCEN2
   ```

2. Überprüfen Sie mit folgendem Befehl, ob die Struktur ordnungsgemäß erstellt wurde:

   ```
   D XCF,STR,STRNAME=IBM1SCEN2
   ```

   Dieser Befehl führt zu folgender Ausgabe:

   ```
   RESPONSE=SC61
   IXC360I 07.58.51 DISPLAY XCF 581
   STRNAME: IBM1SCEN2
   STATUS: NOT ALLOCATED
   POLICY INFORMATION:
   POLICY SIZE : 2048 M
   POLICY INITSIZE: 2048 M
   POLICY MINSIZE : 1536 M
   FULLTHRESHOLD : 85
   ALLOWAUTOALT : YES
   REBUILD PERCENT: N/A
   ```

```
DUPLEX : DISABLED
ALLOWREALLOCATE: YES
PREFERENCE LIST: CF01
ENFORCEORDER : NO
EXCLUSION LIST IS EMPTY

EVENT MANAGEMENT: MESSAGE-BASED   MANAGER SYSTEM NAME: SC53
MANAGEMENT LEVEL : 01050107
```

Zu diesem Zeitpunkt ist die Struktur der Gruppe mit gemeinsamer Warteschlange noch nicht zugeordnet (siehe Zeile STATUS).

3. Konfigurieren Sie IBM MQ für die Verwendung der in der CFRM-Richtlinie definierten Struktur.
   a. Erstellen Sie mit dem Befehl DEFINE CFSTRUCT unter Angabe des Strukturnamens SCEN2 ein IBM MQ-CFSTRUCT-Objekt.

```
DEFINE CFSTRUCT(SCEN2)
CFCONLOS(TOLERATE)
CFLEVEL(5)
DESCR('Structure for SCM scenario 2')
RECOVER(NO)
RECAUTO(YES)
OFFLOAD(DB2)
OFFLD1SZ(64K) OFFLD1TH(70)
OFFLD2SZ(64K) OFFLD2TH(80)
OFFLD3SZ(64K) OFFLD3TH(90)
```

   b. Überprüfen Sie die Struktur mit dem Befehl DISPLAY CFSTRUCT.
   c. Weisen Sie mit folgendem MQSC-Befehl die gemeinsam genutzte Warteschlange SCEN2.Q an, die Struktur SCEN2 zu verwenden:

```
DEFINE QLOCAL(SCEN2.Q) QSGDISP(SHARED) CFSTRUCT(SCEN2) MAXDEPTH(999999999)
```

4. Reihen Sie mithilfe von IBM MQ Explorer eine einzelne Nachricht in die Warteschlange SCEN2.Q ein und rufen Sie sie wieder daraus ab.
5. Überprüfen Sie mit folgendem Befehl, dass die Struktur jetzt zugeordnet ist:

```
D XCF,STR,STRNAME=IBM1SCEN2
```

Überprüfen Sie die Ausgabe des Befehls (ein Teil davon wird angezeigt) und stellen Sie sicher, dass die STATUS-Linie ALLOCATED anzeigt.

```
RESPONSE=SC61
IXC360I 08.31.27 DISPLAY XCF 703
STRNAME: IBM1SCEN2
STATUS: ALLOCATED
EVENT MANAGEMENT: MESSAGE-BASED
TYPE: SERIALIZED LIST
POLICY INFORMATION:
POLICY SIZE : 2048 M
POLICY INITSIZE: 2048 M
POLICY MINSIZE : 1536 M
FULLTHRESHOLD : 85
ALLOWAUTOALT : YES
REBUILD PERCENT: N/A
DUPLEX : DISABLED
ALLOWREALLOCATE: YES
PREFERENCE LIST: CF01
ENFORCEORDER : NO
EXCLUSION LIST IS EMPTY
```

Beachten Sie außerdem die Werte der Felder im Abschnitt SPACE USAGE:

- ENTRIES
- ELEMENTS
- EMCS

- LOCKS

Es folgt ein Beispiel für die Werte:

```
SPACE USAGE   IN-USE      TOTAL    %
ENTRIES:      344686     345242   99
ELEMENTS:    6548455    6548467   99
EMCS:              2     780318    0
LOCKS:          1024
```

## Ergebnisse

Damit haben Sie die Basiskonfiguration erstellt. Sie können jetzt eine beliebige Methode wählen, um eine Vorstellung von der grundsätzlichen Leistung Ihrer Konfiguration zu erhalten.

## Nächste Schritte

Es wird empfohlen, das Basisszenario zu testen. Sie können zum Beispiel die folgenden drei Anwendungen verwenden, indem Sie die Anwendungen in der genannten Reihenfolge starten und gleichzeitig ausführen.

1. Fordern Sie mit einer PCF-Anwendung alle fünf Sekunden die aktuelle Länge ( **CURDEPTH** ) der Warteschlange SCEN2.Q an. Anhand der Ausgabe kann die Länge der Warteschlange über einen längeren Zeitraum grafisch dargestellt werden.
2. Eine aus Einzelthreads bestehende abrufende Anwendung ruft wiederholt Nachrichten aus SCEN2.Q ab und verwendet dazu eine Abrufanforderung mit unbegrenzter Wartezeit. Um eine Verarbeitung der abgerufenen Nachrichten zu simulieren, wird die abrufende Anwendung nach jeweils zehn abgerufenen Nachrichten vier Millisekunden lang angehalten.
3. Eine aus Einzelthreads bestehende einreihende Anwendung reiht insgesamt eine Million nicht persistente 4-KB-Nachrichten in SCEN2.Q ein. Da diese Anwendung zwischen dem Einreihen der einzelnen Nachrichten nicht angehalten wird, werden Nachrichten schneller in SCEN2.Q eingereiht, als sie von der abrufenden Anwendung abgerufen werden können.

   Dies führt dazu, dass die Länge von SCEN2.Q zunimmt, solange die einreihende Anwendung aktiv ist.

   Sobald die Struktur IBM1SCEN2 voll ist und die einreihende Anwendung den Ursachencode MQRC_STORAGE_MEDIUM_FULL empfängt, wird die einreihende Anwendung fünf Sekunden lang inaktiviert, bevor sie versucht, die nächste Nachricht in die Warteschlange einzureihen.

Sie können die Ergebnisse der CURDEPTH-Anwendung über einen längeren Zeitraum grafisch darstellen. Die Darstellung erfolgt in Form einer Sägezahnwelle, wenn die einreihende Anwendung angehalten wird, damit die Warteschlange teilweise geleert werden kann.

Wechseln Sie zu „SCM zur Ausgangsstruktur hinzufügen” auf Seite 223.

**Zugehörige Konzepte**

„Storage Class Memory mit gemeinsam genutzten Warteschlangen verwenden” auf Seite 206
Die Nutzung von Storage Class Memory (SCM) kann in Verbindung mit gemeinsam genutzten Warteschlangen von IBM MQ for z/OS von Vorteil sein.

z/OS *SCM zur Ausgangsstruktur hinzufügen*

SCM für verbesserte Leistung in IBM MQ hinzufügen

## Informationen zu diesem Vorgang

**Wichtig:** IBM z16 ist als letzte Generation von IBM Z ® geplant, um die Verwendung von virtuellem Flashspeicher (auch bekannt als Storage Class Memory oder SCM) für Coupling-Facility-Images zu unterstützen. Weitere Informationen finden Sie unter IBM Z und IBM LinuxONE 4Q 2023 Statements of Direction.

Alternativ sollten Sie entweder größere Strukturen verwenden oder Nachrichten in SMDS auslagern.

Für diesen Teil der Aufgabe wird die im Abschnitt „Verbesserte Leistung - Basiskonfiguration” auf Seite 220 beschriebene Basiskonfiguration verwendet. Das Szenario beschreibt die Hinzufügung von SCM zur Ausgangsstruktur.

Die Endkonfiguration wird in Abbildung 64 auf Seite 224 dargestellt.

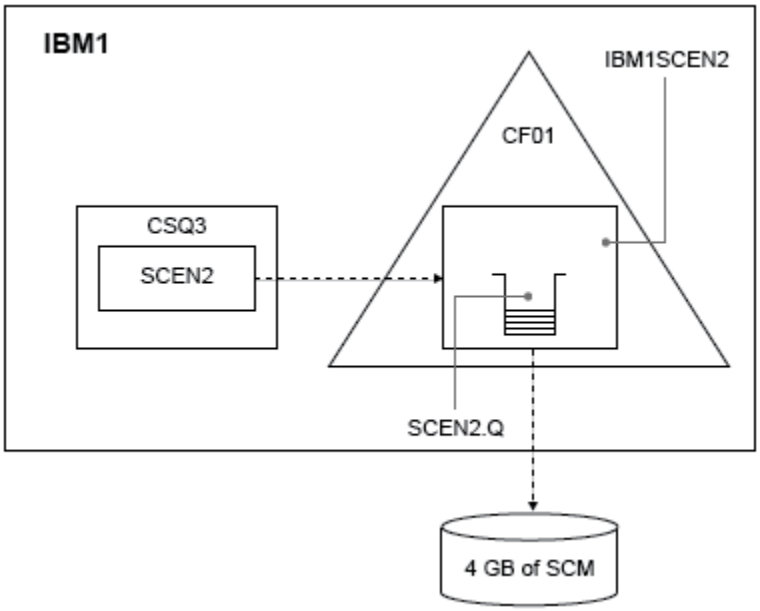



*Abbildung 64. Konfiguration mit Hinzufügung von SCM für verbesserte Leistung*

## Vorgehensweise

1. Fügen Sie SCM mit einer Größe von 4 GB zur Struktur IBM1SCEN2 hinzu, indem Sie wie folgt vorgehen:
   a) Überprüfen Sie mit folgendem Befehl, wie viel SCM installiert und CF01 zugeordnet ist:

```
D CF,CFNAME=CF01
```

   b) Überprüfen Sie die STORAGE-CLASS MEMORY -Abbildungen im Abschnitt STORAGE CONFIGURATION der angezeigten Ausgabe, um den verfügbaren Speicher anzuzeigen.
   c) Aktualisieren Sie die CFRM-Richtlinie wie folgt mit den Schlüsselwörtern SCMMAXSIZE und SCMALGORITHM:

```
STRUCTURE
NAME(IBM1SCEN2)
SIZE(2048M)
INITSIZE(2048M)
ALLOWAUTOALT(YES)
FULLTHRESHOLD(85)
PREFLIST(CF01)
ALLOWREALLOCATE(YES)
DUPLEX(DISABLED)
ENFORCEORDER(NO)
SCMMAXSIZE(4G)
SCMALGORITHM(KEYPRIORITY1)
```

2. Aktivieren Sie die CFRM-Richtlinie mit folgendem Befehl:

```
SETXCF START,POLICY,TYPE=CFRM,POLNAME=IBM1SCEN2
```

3. Erstellen Sie die Struktur IBM1SCEN2 neu.

   Dies ist erforderlich, weil die Struktur zugeordnet war, als Sie die obigen Änderungen vorgenommen haben.

Geben Sie zur erneuten Erstellung der Struktur folgenden Befehl aus:

```
SETXCF START,REBUILD,STRNM=IBM1SCEN2
```

4. Geben Sie folgenden Befehl aus, um die neue Konfiguration der Struktur zu bestätigen:

```
D XCF,STR,STRNAME=IBM1SCEN2
```

Überprüfen Sie die Ausgabe des Befehls, aus der hier ein Ausschnitt angezeigt wird:

```
SPACE USAGE     IN-USE       TOTAL      %
ENTRIES:          33         342684     0
ELEMENTS:         48        6503697     0
EMCS:              2         575600     0
LOCKS:                         1024
```

## Ergebnisse

Berechnen Sie die Änderung bei der Nutzung von Realspeicher durch die Zunahme des für die Verwendung von SCM erforderlichen Steuerspeichers.

- Bevor SCM zur Struktur hinzugefügt wird, weist die Struktur folgende Gesamtzahlen auf (siehe „Verbesserte Leistung - Basiskonfiguration” auf Seite 220):
  - 345.242 Einträge
  - 6.548.467 Elemente
  - 780.318 EMCS
- Nachdem SCM zur Struktur hinzugefügt wurde, weist die Struktur folgende Gesamtzahlen auf:
  - 342.684 Einträge
  - 6.503.697 Elemente
  - 575.600 EMCS

Die Zahlen zeigen, dass sich die Struktur nach der Hinzufügung von SCM verkleinert hat, und zwar um:

- 2558 Einträge
- 44.770 Elemente
- 204.718 EMCS

Für eine 2-GB-Struktur, der SCM mit einer Größe von 4 GB zugeordnet ist, wird folgender Strukturspeicherplatz für die Verwaltung von SCM verwendet:

```
(2558 + 44,770 + 204,718) * 256 = 61.5 MB
```

Beachten Sie, dass eine Hinzufügung von mehr SCM die Größe der Struktur voraussichtlich nur geringfügig weiter reduziert, weil mit der Strukturgröße und der Größe des zugeordneten SCM auch der Steuerspeicherplatz für die SCM-Überwachung zunimmt.

## Nächste Schritte

Wiederholen Sie die am Ende des Abschnitts „Verbesserte Leistung - Basiskonfiguration” auf Seite 220 beschriebenen Tests.

Sie können die Ergebnisse der überarbeiteten Anwendung über einen längeren Zeitraum grafisch darstellen. Im Vergleich zur vorherigen Darstellung erhalten Sie jetzt eine Ausgabe ohne Sägezahnwelle, weil die einreihende Anwendung nicht mehr darauf warten muss, dass die Warteschlange teilweise geleert wird.

Weitere Informationen finden Sie unter MP16: WebSphere MQ for z/OS -Kapazitätsplanung und -optimierung.

## z/OS Verteilte Steuerung von Warteschlangen und Gruppen mit gemeinsamer Warteschlange

Die verteilte Steuerung von Warteschlangen und Gruppen mit gemeinsamer Warteschlange sind zwei Verfahren, mit denen Sie die Verfügbarkeit von Anwendungssystemen erhöhen können. In diesem Abschnitt werden diese beiden Verfahren ausführlicher erläutert.

Um die hohe Verfügbarkeit von Nachrichten in gemeinsam genutzten Warteschlangen zu ergänzen, bietet die IBM MQ-Komponente für die verteilte Steuerung von Warteschlangen zusätzlich folgende Funktionen:

- Höhere Verfügbarkeit des Netzes
- Höher Kapazität der Gruppe mit gemeinsamer Warteschlange für eingehende Netzverbindungen

Abbildung 65 auf Seite 226 zeigt die verteilte Steuerung von Warteschlangen und Gruppen mit gemeinsamer Warteschlange. Es sind zwei Warteschlangenmanager innerhalb eines Sysplex zu sehen, die beide zu derselben Gruppe mit gemeinsamer Warteschlange gehören. Beide können auf die gemeinsam genutzte Warteschlange 'SQ1' zugreifen. Warteschlangenmanager im Netz (z. B. unter AIX und Windows) können über ihren jeweiligen Kanalinitiator Nachrichten in diese Warteschlange einreihen. Geklonte Anwendungen auf beiden Warteschlangenmanagern bedienen die Warteschlange.

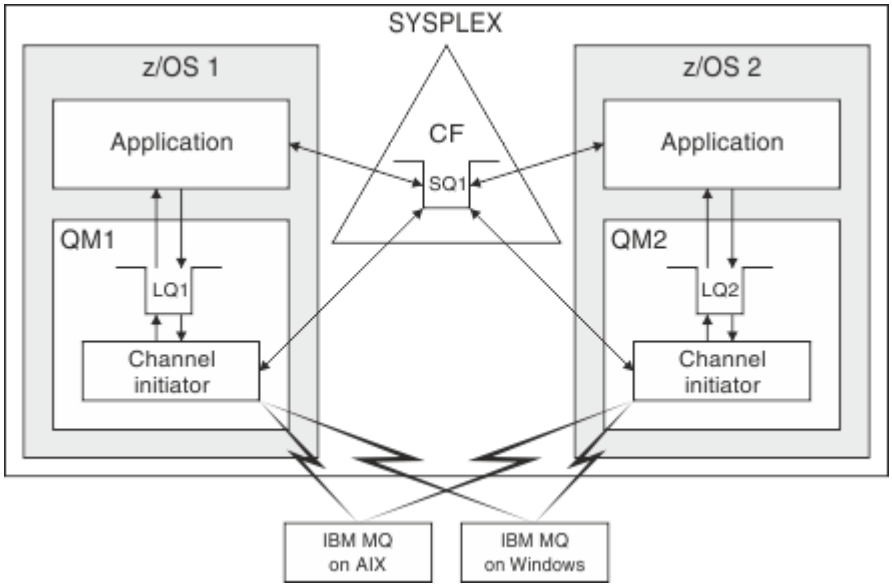



*Abbildung 65. Verteilte Steuerung von Warteschlangen und Gruppen mit gemeinsamer Warteschlange*

**Zugehörige Konzepte**

„Gemeinsam genutzte Kanäle" auf Seite 226
In diesem Abschnitt werden die Konzepte gemeinsam genutzter Kanäle und deren Verwendung in Verbindung mit IBM MQ for z/OS erläutert.

„Einreihung in Warteschlange innerhalb von Gruppen" auf Seite 232
In diesem Abschnitt wird die gruppeninterne Warteschlangensteuerung beschrieben, eine Funktion von IBM MQ for z/OS, die für die z/OS-Plattform einzigartig ist. Diese Funktion steht nur Warteschlangenmanagern zur Verfügung, die in einer Gruppe mit gemeinsamer Warteschlange definiert sind.

„Cluster und Gruppen mit gemeinsamer Warteschlange" auf Seite 229
In diesem Abschnitt wird erläutert, wie Gruppen mit gemeinsamer Warteschlange zusammen mit Clustern verwendet werden.

## z/OS *Gemeinsam genutzte Kanäle*

In diesem Abschnitt werden die Konzepte gemeinsam genutzter Kanäle und deren Verwendung in Verbindung mit IBM MQ for z/OS erläutert.

Einige Netzprodukte stellen einen Mechanismus bereit, mit dem Serverfehlern gegenüber dem Netz verborgen oder ein Lastausgleich für die eingehenden Netzanforderungen zwischen einer Gruppe von berechtigten Servern bewirkt werden kann. Die Netzprodukte machen einen *generischen Port* für eingehende Netzverbindungsanforderungen verfügbar, und die eingehende Anforderung kann durch Herstellung einer Verbindung zu einem der berechtigten Server erfüllt werden.

Zu diesen Netzprodukten gehören unter anderem folgende:

- VTAMVTAM Generic Resources
- SYSPLEX Distributor

Der Kanalinitiator profitiert von diesen Produkten, indem er die Funktionalität von gemeinsam genutzten Warteschlangen einsetzt.

Es gibt zwei Arten von gemeinsam genutzten Kanälen: *gemeinsam genutzte eingehende Kanäle* und *gemeinsam genutzte ausgehende Kanäle*.

- Gemeinsam genutzte eingehende Kanäle
- Gemeinsam genutzte ausgehende Kanäle

Weitere Informationen zu Kanälen finden Sie unter

- Gemeinsam genutzte Kanäle - Zusammenfassung
- Status gemeinsam genutzter Kanäle

## Gemeinsam genutzte eingehende Kanäle

Jeder Kanalinitiator in der Gruppe mit gemeinsamer Warteschlange startet eine zusätzliche Empfangsprogrammtask, um an einem *generischen Port* empfangsbereit zu sein. Dieser generische Port wird von einem der unterstützenden Verfahren (VTAM, TCP/IP) für das Netz bereitgestellt. Am generischen Port eingehende Netzverbindungsanforderungen werden von der Netztechnologie einem der Empfangsprogramme in der Gruppe mit gemeinsamer Warteschlange (Queue Sharing Group, QSG) zugeteilt, die am generischen Port empfangsbereit sind.

Sie können einen Kanal in dem Kanalinitiator starten, an den die eingehende Verbindung gerichtet ist, wenn der Kanalinitiator Zugriff auf eine Kanaldefinition für einen Kanal mit diesem Namen hat. Sie können angeben, ob eine Kanaldefinition nur für einen bestimmten Warteschlangenmanager bestimmt ist oder im gemeinsamen Repository gespeichert und so überall verfügbar ist (globale Definition). Dies bedeutet, dass Sie eine Kanaldefinition in jedem Kanalinitiator innerhalb der Gruppe mit gemeinsamer Warteschlange bereitstellen können, indem Sie festlegen, dass es eine globale Definition ist.

Als weiterer Unterschied beim Starten eines Kanals über den generischen Port erfolgt die Kanalsynchronisation mit der Gruppe mit gemeinsamer Warteschlange und nicht mit einem einzelnen Warteschlangenmanager. Angenommen, ein ferner Warteschlangenmanager startet einen Kanal über einen generischen Port. Beim ersten Start des Kanals wird er beispielsweise auf Warteschlangenmanager 'QM1' gestartet und anschließend beginnt der Nachrichtenfluss. Wenn der Kanal gestoppt und dann auf Warteschlangenmanager 'QM2' neu gestartet wird, sind die Informationen zur Anzahl der Nachrichten, die über diesen Kanal gegangen sind, immer noch korrekt, weil die Synchronisation mit der Gruppe mit gemeinsamer Warteschlange ausgeführt wurde.

Mithilfe des eingehenden Kanals, der über den generischen Port gestartet wurde, können Sie Nachrichten in eine beliebige Warteschlange einreihen. Dem fernen Warteschlangenmanager ist nicht bekannt, ob die Zielwarteschlange gemeinsam genutzt wird oder nicht. Wenn das Ziel eine gemeinsam genutzte Warteschlange ist, stellt der ferne Warteschlangenmanager die Verbindung über einen beliebigen Kanalinitiator mit einem Lastausgleichsverfahren her und die Nachrichten werden in die gemeinsam genutzte Warteschlange eingereiht.

Wenn das Ziel eine private Warteschlange ist, werden die Nachrichten in diejenige private Warteschlange eingereiht, deren Besitzer der Warteschlangenmanager ist, mit dem die aktuelle Instanz des Kanals gerade verbunden ist. In einer solchen Umgebung, die als *replizierte lokale Warteschlangen* bezeichnet wird, muss für alle Warteschlangenmanager derselbe Satz an privaten Warteschlangen definiert sein.

## SVRCONN-Kanäle für eine Gruppe mit gemeinsame Warteschlange konfigurieren

Die optimale Konfiguration für SVRCONN-Kanäle in einer Gruppe mit gemeinsamer Warteschlange ist die Konfiguration privater Empfangsprogramme in jedem Kanalinitiator (CHINIT), die alle eine unterschiedliche Portnummer von den Punkt-zu-Punkt-Kanälen verwenden. Diese Empfangsprogrammports werden dann als "Back-End-Ressourcen" für einen neuen Lastausgleichsmechanismus verwendet, z. B. Sysplex Distributor unter Verwendung virtueller IP-Adressen. Die externe virtuelle IP-Adresse wird dann als Zieladresse für die CLNTCONN-Definitionen im Netz verwendet. Der SVRCONN-Kanal kann mit dem Parameter 'QSGDISP(GROUP)' definiert werden, damit dieselbe Definition für alle Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange verfügbar ist. Diese Konfiguration vermeidet die Verwendung eines gemeinsam genutzten Listeners und verringert daher die Leistungseinwirkungen der Gruppe mit gemeinsamer Warteschlange, die den Status gemeinsam genutzter Kanäle verwaltet, was für Client/Server-Kanäle nicht erforderlich ist.

## Gemeinsam genutzte ausgehende Kanäle

Ein ausgehender Kanal wird als gemeinsam genutzt betrachtet, wenn er Nachrichten aus einer gemeinsam genutzten Übertragungswarteschlange abruft. Wenn er gemeinsam genutzt wird, verfügt er über Synchronisationsinformationen auf der Ebene der Gruppe mit gemeinsamer Warteschlange. Dies bedeutet, dass der Kanal in einer anderen Warteschlangenmanager- und Kanalinitiatorinstanz innerhalb der Gruppe mit gemeinsamer Warteschlange neu gestartet werden kann, falls das Kommunikationssubsystem, der Kanalinitiator oder der Warteschlangenmanager fehlschlägt. Der auf diese Weise ausgeführte Neustart fehlgeschlagener Kanäle ist eine Funktion gemeinsamer Kanäle, die als *Peer-Kanalwiederherstellung* bezeichnet wird.

### Lastausgleich für gemeinsam genutzte ausgehende Kanäle

Ein gemeinsamer ausgehender Kanal kann in einem beliebigen Kanalinitiator innerhalb der Gruppe mit gemeinsamer Warteschlange gestartet werden, sofern Sie nicht angegeben haben, dass er nur in einem bestimmten Kanalinitiator gestartet werden soll. Welcher Kanalinitiator von IBM MQ ausgewählt wird, hängt von folgenden Kriterien ab:

- Ist das erforderliche Kommunikationssubsystem aktuell für den Kanalinitiator verfügbar?
- Ist eine Db2-Verbindung für den Kanalinitiator verfügbar?
- Welcher Kanalinitiator hat aktuell die geringste Arbeitslast? Zur Arbeitslast gehören auch Kanäle, die aktiv sind oder einen Vorgang wiederholen.

## Gemeinsam genutzte Kanäle - Zusammenfassung

Gemeinsam genutzte Kanäle unterscheiden sich in folgenden Punkten von privaten Kanälen:

**Privater Kanal**
An einen einzelnen Kanalinitiator gebunden.

- Ausgehender Kanal verwendet eine lokale Übertragungswarteschlange.
- Eingehender Kanal startet über einen lokalen Port.
- Synchronisationsinformationen werden in der Warteschlange 'SYSTEM.CHANNEL.SYNCQ' gespeichert.

**Gemeinsamer Kanal**
Lastausgleich mit Hochverfügbarkeit.

- Ausgehender Kanal verwendet eine gemeinsam genutzte Übertragungswarteschlange.
- Eingehender Kanal startet über einen generischen Port.
- Synchronisationsinformationen werden in der Warteschlange 'SYSTEM.QSG.CHANNEL.SYNCQ' gespeichert.

Beim Start des Kanals geben Sie mit den CHLDISP-Optionen des Befehls START CHANNEL an, ob es sich um einen privaten oder einen gemeinsam genutzten Kanal handelt. Für einen gemeinsamen Kanal kann der Start auf dieselbe Weise ausgelöst werden wie für einen privaten Kanal. Wenn jedoch ein gemeinsamer Kanal gestartet wird, führt IBM MQ einen Lastausgleich durch und startet den Kanal in dem am besten geeigneten Kanalinitiator innerhalb der Gruppe mit gemeinsamer Warteschlange. (Falls erforderlich, können Sie angeben, dass ein gemeinsamer Kanal nur in einem bestimmten Kanalinitiator gestartet werden soll.)

## Status gemeinsam genutzter Kanäle

Die Kanalinitiatoren in einer Gruppe mit gemeinsamer Warteschlange verwalten in Db2 eine Tabelle mit den Status gemeinsam genutzter Kanäle. Darin wird aufgezeichnet, welche Kanäle in welchen Kanalinitiatoren aktiv sind. Die Tabelle mit den Status gemeinsamer Kanäle kommt zum Einsatz, wenn ein Kanalinitiator oder ein Kommunikationssubsystem fehlschlägt. Darin ist angegeben, welche Kanäle in einem anderen Kanalinitiator innerhalb der Gruppe mit gemeinsamer Warteschlange neu gestartet werden müssen.

### z/OS *Cluster und Gruppen mit gemeinsamer Warteschlange*

In diesem Abschnitt wird erläutert, wie Gruppen mit gemeinsamer Warteschlange zusammen mit Clustern verwendet werden.

Sie können Ihre gemeinsam verwalteten Warteschlangen mithilfe einer einzigen Definition für ein Cluster bereitstellen. Dazu geben Sie beim Definieren der gemeinsam genutzten Warteschlange den Namen des Clusters an.

Für die Benutzer im Netz wird angezeigt, dass die gemeinsam genutzte Warteschlange von jedem der Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange per Hosting bereitgestellt wird (es wird hingegen nicht angegeben, dass die gemeinsam genutzte Warteschlange von der Gruppe mit gemeinsamer Warteschlange per Hosting bereitgestellt wird). Um Nachrichten in die gemeinsam genutzte Warteschlange einzureihen, können Clients eine Sitzung mit einem beliebigen Mitglied der Gruppe mit gemeinsamer Warteschlange starten.

Abbildung 66 auf Seite 229 zeigt, wie Cluster-Mitglieder über ein beliebiges Mitglied einer Gruppe mit gemeinsamer Warteschlange auf eine gemeinsam genutzte Warteschlange zugreifen können.

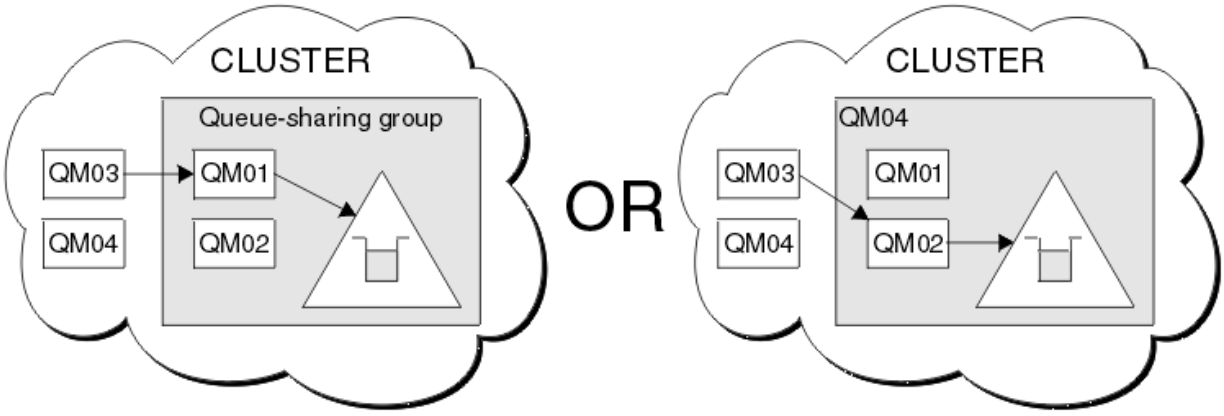



*Abbildung 66. Gruppe mit gemeinsamer Warteschlange als Teil eines Clusters*

## z/OS Lastverteilung mit gemeinsam genutzten Warteschlangen beeinflussen

In diesem Abschnitt werden die Faktoren beschrieben, die die Lastverteilung mit gemeinsam genutzten Warteschlangen in einer Gruppe mit gemeinsamer Warteschlange beeinflussen.

IBM MQ stellt keinen Lastausgleich für gemeinsam genutzte Warteschlangen bereit. Die Lastverteilung in einer Gruppe mit gemeinsamer Warteschlange (Queue-Sharing Group, QSG) kann jedoch auf Basis der *Extraktion mit Pull-Operationen* beeinflusst werden. Die Auswahl, welcher Warteschlangenmanager eine Warteschlange bedient (d. h. eine an eine gemeinsam genutzte Warteschlange gerichtete Nachricht empfängt), wird durch die verfügbare Verarbeitungskapazität jedes einzelnen Warteschlangenmanagers in der Gruppe mit gemeinsamer Warteschlange und durch die für das gesamte Sysplex definierten Auslastungsmanagementziele beeinflusst.

Es ist jedoch wichtig zu verstehen, dass auch der Warteschlangenmanager, der die Operation 'MQPUT' für eine Nachricht ausführt, einen großen Einfluss auf die Entscheidung hat, welcher Nachrichtenmanager die Nachricht empfängt.

## Größere Wahrscheinlichkeit zum Ausführen von 'MQGET' für lokalen Warteschlangenmanager

Für eine Anwendung, die die Operation 'MQPUT' ausführt, wird der lokale Warteschlangenmanager als derjenige angegeben, mit dem die Anwendung verbunden ist.

Welcher Warteschlangenmanager genau die Operation 'MQPUT' für eine Nachricht ausführt, indem er 'MQGET' im Namen einer abrufenden Anwendung ausführt, wird durch folgende Überlegungen beeinflusst.

Wenn eine Nachricht in eine leere gemeinsam genutzte Warteschlange eingereiht wird, wird der lokale Warteschlangenmanager normalerweise vor allen anderen Warteschlangenmanagern in der Gruppe mit gemeinsamer Warteschlange darüber benachrichtigt. Wenn der lokale Warteschlangenmanager in der Lage ist, die Nachricht zu verarbeiten, empfängt er vor jedem anderen Warteschlangenmanager in der Gruppe eine Listenübergangsbenachrichtigung von der Coupling-Facility (CF). (Eine Listenübergangsbenachrichtigung ist eine Benachrichtigung darüber, dass die gemeinsam genutzte Warteschlange vom leeren Zustand zum nicht leeren Zustand gewechselt ist.)

In diesem Fall gibt es die folgenden möglichen Szenarien:

1. 'MQPUT' für nicht permanente Nachricht außerhalb des Synchronisationspunkts und *schnelle Einreihung für wartende Abruffunktion*.

   Wenn es eine Anwendung mit der Anweisung *MQGET mit Wartestatus* auf dem lokalen Warteschlangen für die Warteschlange gibt, dann wird die Operation 'MQPUT' für die Nachricht direkt an den Puffer der abrufenden Anwendung übergeben und nicht in die Warteschlange gestellt. Dies gilt für gemeinsam genutzte und nicht gemeinsam genutzte Warteschlangen. Dieser Mechanismus wird häufig als *schnelle Einreihung für wartende Abruffunktion* bezeichnet. Im Falle von gemeinsam genutzten Warteschlangen wird kein anderer Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange benachrichtigt, weil kein Übergang vom leeren zum nicht leeren Zustand für die Warteschlange stattfindet. Wenn dieser Warteschlangenmanager alle Einreihungsoperationen von dieser Anwendung verarbeiten kann und keine anderen Anwendungen Nachrichten in die Warteschlange einreihen, bedeutet dies beispielsweise, dass kein anderer Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange an der Entleerung dieser Warteschlange beteiligt ist. Wenn jedoch auf dem lokalen Warteschlangenmanager die Anweisung 'MQGET mit Wartestatus' nicht vorhanden ist und eine Nachricht in die gemeinsam genutzte Warteschlange eingereiht wird, dann benachrichtigt die Coupling-Facility die anderen Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange gemäß ihren Benachrichtigungsregeln für Listenübergänge.
2. 'MQPUT' für permanente oder im Synchronisationspunkt befindliche Nachricht

   Wenn es in diesem Fall eine Anwendung mit der Anweisung *MQGET mit Wartestatus* auf dem lokalen Warteschlangenmanager gibt, dann wird die Nachricht in die gemeinsam genutzte Warteschlange eingereiht und die Coupling-Facility benachrichtigt die anderen Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange gemäß ihren Benachrichtigungsregeln für Listenübergänge. Der lokale Warteschlangenmanager wartet jedoch nicht auf eine Übergangsbenachrichtigung von der Coupling-Facility, sondern berücksichtigt zuerst die lokale Anweisung *MQGET mit Wartestatus* und ruft normalerweise diese Nachricht im Namen der Anwendung ab, bevor einer der anderen Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange auf die Benachrichtigung von der Coupling-Facility reagieren kann. Dies hängt jedoch davon ab, wie ausgelastet der lokale Warteschlangenmanager ist.

Ansonsten versucht jeder Warteschlangenmanager, der von der Coupling-Facility über den Eingang der Nachricht in der leeren Warteschlange benachrichtigt wurde, die Nachricht als erster abzurufen. Der Warteschlangenmanager, der am schnellsten reagiert, verarbeitet die neue Nachricht.

3. Wenn die Warteschlange, für die die Coupling-Facility eine Benachrichtigung über die Änderung vom leeren zum nicht leeren Zustand gesendet hat, nicht entleert wird, haben schließlich alle verbundenen Warteschlangenmanager Gelegenheit, die Warteschlange zu verarbeiten. In diesem Fall wird der Mechanismus zur Verarbeitung der Arbeitslast als *Extraktion durch Pull-Operationen* ("Pull-basiert") bezeichnet.

Dieser Ansatz ermöglicht eine Leistungssteigerung gegenüber der allein auf der Extraktion durch Pull-Operationen basierenden Lastverteilung. Das Ziel dieses Ansatzes ist es, den Vorteil der Hochverfügbarkeit von Warteschlangen, die einer Coupling-Facility zugeordnet sind, zu nutzen, und dem Warteschlangenmanager gegebenenfalls gleichzeitig die Möglichkeit zu geben, die Operation 'MQGET' ohne Verweis auf die Coupling-Facility auszuführen, um die Nachrichtenarbeitslast insgesamt so effizient wie möglich zu verarbeiten.

Alternative Ansätze können umgesetzt werden, wenn die gleichmäßige Verteilung der Arbeitslast wichtiger als die zuvor beschriebenen Leistungsverbesserungen ist. Beispielsweise wenn ausgeschlossen werden soll, dass eine abrufende und eine einreihende Anwendungen mit demselben Warteschlangenmanager verbunden sind. Bei diesem Ansatz werden alle Nachrichten in die Warteschlange eingereiht, und wenn die Warteschlange vom leeren zum nicht leeren Zustand wechselt, werden alle Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange gemäß dem Coupling-Facility-Algorithmus für die Handhabung solcher Übergänge benachrichtigt. Darüber hinaus ist der Mechanismus *schnelle Einreihung für wartende Abruffunktion* in diesem Fall nicht zulässig.

## z/OS Weitere Informationen zu gemeinsam genutzten Warteschlangen und Gruppen mit gemeinsamer Warteschlange

Mit Hilfe der in diesem Abschnitt beschriebenen Tabelle können Sie weitere Informationen darüber erhalten, wie IBM MQ for z/OS gemeinsam genutzte Warteschlangen und Gruppen mit gemeinsamer Warteschlange verwendet.

*Tabelle 19. Weitere Informationen zu gemeinsam genutzten Warteschlangen und Gruppen mit gemeinsamer Warteschlange*

| **Thema** | **Quelle** |
|---|---|
| Wiederherstellung von Gruppen mit gemeinsamer Warteschlange | „Wiederherstellung und Neustart unter z/OS” auf Seite 278 |
| Sicherheit von Gruppen mit gemeinsamer Warteschlange | „Sicherheitskonzepte in IBM MQ for z/OS” auf Seite 297 |
| Private und globale Objektdefinitionen<br>Befehle an verschiedene Warteschlangenmanager<br>leiten | Quellen, aus denen Sie Befehle unter z/OS |
| Planen der Coupling-Facility-Umwelt | Coupling-Facility-Ressourcen definieren |
| SMDS-Umgebung planen | Umgebung mit gemeinsam genutzten Nachrichtendateien planen |
| :NONE.<br>Db2-Umgebung | Db2-Umgebung planen |

*Tabelle 19. Weitere Informationen zu gemeinsam genutzten Warteschlangen und Gruppen mit gemeinsamer Warteschlange (Forts.)*

| Thema | Quelle |
|---|---|
| Gemeinsam genutzte Warteschlangen einrichten Systemparameter | „Gemeinsam genutzte Warteschlangen und Gruppen mit gemeinsamer Warteschlange” auf Seite 181 |
| Dienstprogramme für die Migration von Warteschlangen | Referenzinformationen zuIBM MQ -Dienstprogrammen unter z/OS |
| Konsolennachrichten | Nachrichten für IBM MQ for z/OS |
| MQSC-Befehle | MQSC-Befehle |
| IBM MQ-Cluster | Warteschlangenmanager-Cluster konfigurieren |
| Verteilte Steuerung von Warteschlangen in IBM MQ Kanalnamen | Einführung in verteiltes Warteschlangenmanagement |
| Anwendungen schreiben | Übersicht über den Anwendungsentwurf |
| MQCONNX, Aufruf | MQCONNX |

## z/OS Einreihung in Warteschlange innerhalb von Gruppen

In diesem Abschnitt wird die gruppeninterne Warteschlangensteuerung beschrieben, eine Funktion von IBM MQ for z/OS, die für die z/OS-Plattform einzigartig ist. Diese Funktion steht nur Warteschlangenmanagern zur Verfügung, die in einer Gruppe mit gemeinsamer Warteschlange definiert sind.

Informationen zu Gruppen mit gemeinsamer Warteschlange finden Sie unter „Gemeinsam genutzte Warteschlangen und Gruppen mit gemeinsamer Warteschlange” auf Seite 181.

### Konzept der gruppeninternen Warteschlangensteuerung

Zwischen Warteschlangenmanagern innerhalb einer Gruppe mit gemeinsamer Warteschlange ist eine schnelle Nachrichtenübertragung möglich, ohne dass Kanäle definiert werden müssen. Dabei wird die Systemwarteschlange 'SYSTEM.QSG.TRANSMIT.QUEUE' verwendet, bei der es sich um eine gemeinsam genutzte Übertragungswarteschlange handelt. Jeder Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange startet eine Task, die als Agent zur gruppeninternen Steuerung von Warteschlangen bezeichnet wird und auf eingehende Nachrichten in dieser Warteschlange wartet, die für ihren zugehörigen Warteschlangenmanager bestimmt sind. Wenn die Task eine solche Nachricht erkennt, wird die Nachricht aus der Warteschlange abgerufen und in die richtige Zielwarteschlange eingereiht.

Dabei werden Standardregeln für die Namensauflösung verwendet, doch wenn die gruppeninterne Steuerung von Warteschlangen aktiviert ist und sich der Ziel-Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange befindet, dann wird die Nachricht über die Warteschlange 'SYSTEM.QSG.TRANSMIT.QUEUE' direkt an den richtigen Ziel-Warteschlangenmanager übertragen, ohne eine Übertragungswarteschlange und einen Kanal zu verwenden.

Die gruppeninterne Steuerung von Warteschlangen wird mithilfe eines Warteschlangenmanager-Attributs aktiviert. Die gruppeninterne Steuerung von Warteschlangen verschiebt nicht permanente Nachrichten außerhalb des Synchronisationspunkts und permanente Nachrichten innerhalb des Synchronisationspunkts. Wenn bei der Zustellung von Nachrichten an eine Zielwarteschlange Probleme auftreten, versucht die gruppeninterne Steuerung von Warteschlangen, sie in die Warteschlange für nicht zustellbare Nach-

richten einzureihen. Wenn die Warteschlange für nicht zustellbare Nachrichten voll oder nicht definiert ist, werden nicht permanente Nachrichten gelöscht, während permanente Nachrichten zurückgesetzt und an die Warteschlange 'SYSTEM.QSG.TRANSMIT.QUEUE' zurückgegeben werden. Der Agent für die gruppeninterne Steuerung von Warteschlangen wiederholt seine Versuche für die Zustellung der Nachrichten, bis sie erfolgreich waren.

Ein gemeinsamer eingehender Kanal, der eine Nachricht empfängt, die für einen anderen Warteschlangenmanager innerhalb der Gruppe mit gemeinsamer Warteschlange bestimmt ist, kann die Nachricht mithilfe der gruppeninternen Steuerung von Warteschlangen über einen *Hop* an das richtige Ziel übertragen.

Wenn Sie in bestimmten Situationen möchten, dass der lokale Warteschlangenmanager eine Nachricht nicht zuerst an den Ziel-Warteschlangenmanager überträgt, sondern sie direkt in die Zielwarteschlange einreiht, sofern es sich dabei um eine gemeinsam genutzte Warteschlange handelt. Dieses Verhalten lässt sich mit dem Warteschlangenmanager-Attribut 'SQQMNAME' steuern. Wenn Sie den Wert von SQQMNAME auf USE setzen, wird der Befehl MQOPEN auf dem Warteschlangenmanager ausgeführt, der von **ObjectQMgrName**angegeben wird.

Wenn es sich bei der Zielwarteschlange jedoch um eine gemeinsam genutzte Warteschlange handelt und Sie den Wert von SQQMNAME auf IGNORE setzen und **ObjectQMgrName** der eines anderen Warteschlangenmanagers in der Gruppe mit gemeinsamer Warteschlange ist, wird die gemeinsam genutzte Warteschlange auf dem lokalen WS-Manager geöffnet. Wenn der lokale Warteschlangenmanager die Zielwarteschlange nicht öffnen oder eine Nachricht in die Warteschlange einreihen kann, wird die Nachricht entweder über IGQ oder einen IBM MQ -Kanal an den angegebenen **ObjectQMgrName** übertragen.

Mit der gruppeninternen Warteschlangensteuerung können kurze Nachrichten effektiver an Warteschlangen übermittelt werden, die sich auf fernen Warteschlangenmanagern innerhalb einer Gruppe mit gemeinsamer Warteschlange befinden. Die gruppeninterne Warteschlangensteuerung unterstützt auch große Nachrichten, wobei der größte Wert 100 MB *minus* der Länge des Headers der Übertragungswarteschlange beträgt.

**Anmerkung:** Wenn Sie diese Funktion verwenden, müssen die Benutzer für die Warteschlangen aller Warteschlangenmanager innerhalb der Gruppe mit gemeinsamer Warteschlange dieselbe Zugriffsberechtigung haben.

Das folgende Diagramm enthält ein typisches Beispiel einer gruppeninternen Warteschlangensteuerung.

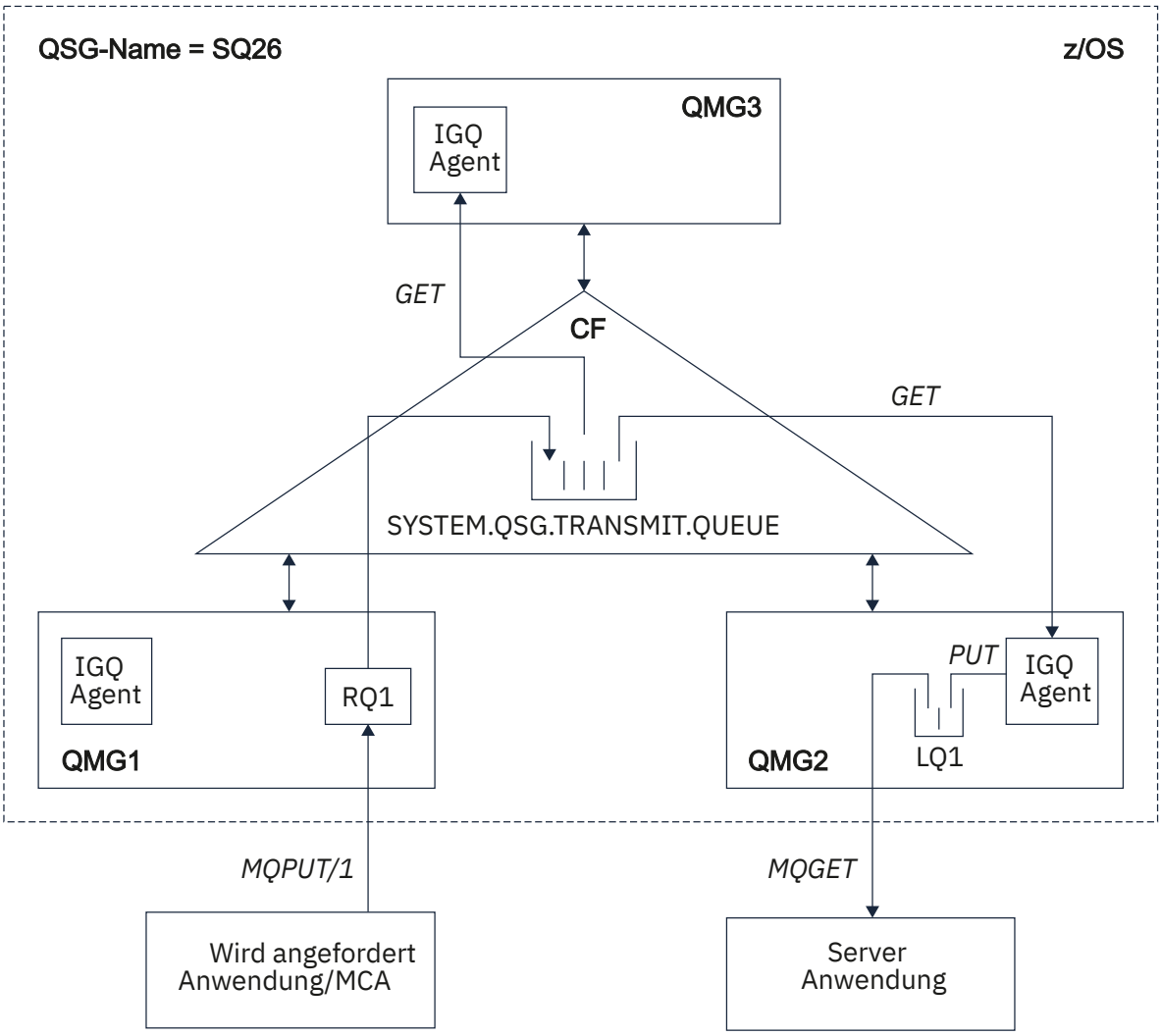



*Abbildung 67. Beispiel einer gruppeninternen Warteschlangensteuerung*

Das Diagramm enthält folgende Punkte:

- Agenten der gruppeninternen Warteschlangensteuerung, die auf drei Warteschlangenmanagern (QMG1, QMG2 und QMG3) ausgeführt werden, die in einer Gruppe mit gemeinsamer Warteschlange namens SQ26 definiert sind.
- Die in der Coupling-Facility (CF) definierte gemeinsame Übertragungswarteschlange SYSTEM.QSG.TRANSMIT.QUEUE.
- Eine Definition einer fernen Warteschlange, die in Warteschlangenmanager QMG1 definiert ist.
- Eine lokale Warteschlange, die in Warteschlangenmanager QMG2 definiert ist.
- Eine anfordernde Anwendung (dabei könnte es sich um einen Nachrichtenkanalagenten handeln), die mit Warteschlangenmanager QMG1 verbunden ist.
- Eine Serveranwendung, die mit Warteschlangenmanager QMG2 verbunden ist.
- Eine Anforderungsnachricht, die in die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE gestellt wird.

**Gruppeninterne Warteschlangensteuerung und der Agent der gruppeninternen Warteschlangensteuerung**

Während der Initialisierung des Warteschlangenmanagers wird ein Agent der gruppeninternen Warteschlangensteuerung gestartet. Wenn Anwendungen ferne Warteschlangen öffnen und dort Nachrichten

einreihen, bestimmt der lokale Warteschlangenmanager, ob für die Nachrichtenübertragung eine gruppeninterne Warteschlangensteuerung verwendet wird. Soll eine gruppeninterne Warteschlangensteuerung verwendet werden, stellt der lokale Warteschlangenmanager die Nachricht in die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE. Der Agent der gruppeninternen Warteschlangensteuerung auf dem fernen Zielwarteschlangenmanager ruft die Nachricht ab und stellt sie in die Zielwarteschlange.

## Terminologie der gruppeninternen Warteschlangensteuerung

Erläuterungen der Terminologie: gruppeninterne Warteschlangensteuerung, gemeinsame Übertragungswarteschlange zur Verwendung durch die gruppeninterne Warteschlangensteuerung und Agent der gruppeninternen Warteschlangensteuerung.

### Einreihung in Warteschlange innerhalb von Gruppen

Mit der gruppeninternen Warteschlangensteuerung kann eine meist schnellere und weniger aufwendige Nachrichtenübertragung zwischen Warteschlangenmanagern innerhalb einer Gruppe mit gemeinsamer Warteschlange durchgeführt werden, ohne Kanäle definieren zu müssen.

### Gemeinsame Übertragungswarteschlange zur Verwendung durch die gruppeninterne Warteschlangensteuerung

Jede Gruppe mit gemeinsamer Warteschlange verfügt über eine gemeinsame Übertragungswarteschlange namens SYSTEM.QSG.TRANSMIT.QUEUE, die zur Verwendung durch die gruppeninterne Warteschlangensteuerung bestimmt ist. Wenn die gruppeninterne Warteschlangensteuerung aktiviert ist, wird SYSTEM.QSG.TRANSMIT.QUEUE beim Öffnen ferner Warteschlangen im Namensauflösungspfad angezeigt. Wenn Anwendungen (einschließlich Nachrichtenkanalagenten) Nachrichten in eine ferne Warteschlange einreihen, ermittelt der lokale Warteschlangenmanager die Eignung der Nachrichten für eine schnelle Übertragung und stellt diese in die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE.

### Agent der gruppeninternen Warteschlangensteuerung

Beim Agenten der gruppeninternen Warteschlangensteuerung handelt es sich um die Task, die bei der Initialisierung des Warteschlangenmanagers gestartet wird, die darauf wartet, dass geeignete Nachrichten in der Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE eintreffen. Der Agent der gruppeninternen Warteschlangensteuerung ruft geeignete Nachrichten aus dieser Warteschlange ab und übermittelt sie an die Zielwarteschlangen.

Der Agent der gruppeninternen Warteschlangensteuerung für die einzelnen Warteschlangenmanager wird immer gestartet, da die gruppeninterne Warteschlangensteuerung vom Warteschlangenmanager selbst für seine eigene interne Verarbeitung verwendet wird.

## z/OS Vorteile der gruppeninternen Warteschlangensteuerung

Die gruppeninterne Warteschlangensteuerung bietet folgende Vorteile: weniger Systemdefinitionen, geringere Systemverwaltung, verbesserte Leistung, Migrationsunterstützung und die Zustellung von Nachrichten beim Multihopping zwischen Warteschlangenmanagern in einer Gruppe mit gemeinsamer Warteschlange.

Die gruppeninterne Warteschlangensteuerung bietet folgende Vorteile:

**Weniger Systemdefinitionen**
Bei der gruppeninternen Warteschlangensteuerung müssen keine Kanäle zwischen den Warteschlangenmanagern in einer Gruppe mit gemeinsamer Warteschlange definiert werden.

**Geringere Systemverwaltung**
Da zwischen den Warteschlangenmanagern in einer Gruppe mit gemeinsamer Warteschlange keine Kanäle definiert werden, besteht kein Aufwand hinsichtlich der Kanalverwaltung.

**Verbesserte Leistung**

Da für die Zustellung einer Nachricht an eine Zielwarteschlange nur ein Agent der gruppeninternen Warteschlangensteuerung erforderlich ist (anstelle zweier temporärer Absender- und Empfängeragenten), kann die Zustellung von Nachrichten über die gruppeninterne Warteschlangensteuerung weniger arbeitsintensiv sein als die Zustellung von Nachrichten über Kanäle. Bei der gruppeninternen Warteschlangensteuerung gibt es nur eine Empfangskomponente, da keine sendende Komponente erforderlich ist. Diese Einsparung ist darauf zurückzuführen, dass die Nachricht dem Agenten der gruppeninternen Warteschlangensteuerung beim Ziel-Warteschlangenmanager für die Zustellung zur Verfügung steht, sobald die Einreihung (PUT) beim lokalen Warteschlangenmanager abgeschlossen und - bei Nachrichten, die im Bereich eines Synchronisationspunkts eingereiht werden - festgeschrieben ist.

**Migrationsunterstützung**

Anwendungen außerhalb einer Gruppe mit gemeinsamer Warteschlange können Nachrichten an eine Warteschlange zustellen, die sich auf einem beliebigen Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange befindet, während sie nur mit einem bestimmten Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange verbunden sind. Dies wird dadurch ermöglicht, dass Nachrichten, die bei einem Empfängerkanal eingehen und für eine Warteschlange auf einem fernen Warteschlangenmanager bestimmt sind, über die gruppeninterne Warteschlangensteuerung auf transparente Weise an die Zielwarteschlange gesendet werden können. Mit dieser Funktion können Anwendungen in der Gruppe mit gemeinsamer Warteschlange implementiert werden, ohne dass Systeme geändert werden müssen, die sich außerhalb der Gruppe mit gemeinsamer Warteschlange befinden.

Im folgenden Diagramm ist eine typische Konfiguration dargestellt, die folgende Merkmale aufweist:

- Eine anfordernde Anwendung, die mit dem Warteschlangenmanager QMG1 verbunden ist, muss eine Nachricht an eine lokale Warteschlange auf dem Warteschlangenmanager QMG3 senden.
- Der Warteschlangenmanager QMG1 ist nur mit dem Warteschlangenmanager QMG2 verbunden.
- Die Warteschlangenmanager QMG2 und QMG3, die in der Vergangenheit über Kanäle verbunden waren, sind jetzt Mitglieder der Gruppe SQ26 mit gemeinsamer Warteschlange.

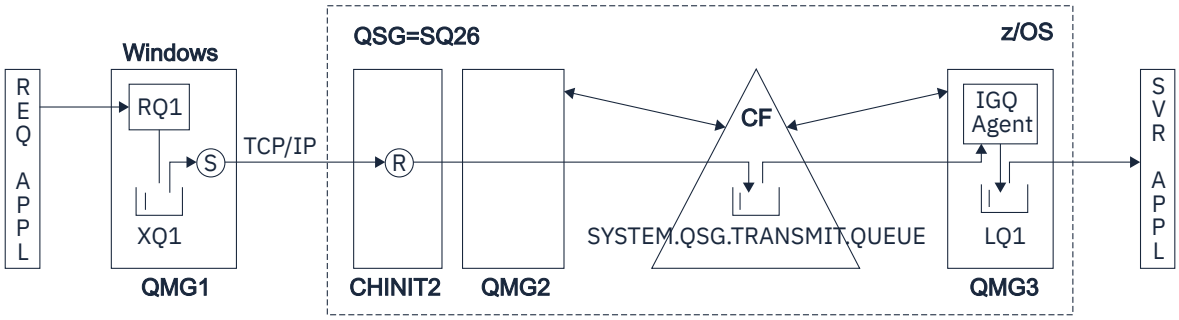



*Abbildung 68. Beispiel einer Migrationsunterstützung*

Der Ablauf der Operationen lautet wie folgt:

1. Die anfordernde Anwendung stellt eine Nachricht, die für die lokale Warteschlange LQ1 auf dem fernen Warteschlangenmanager QMG3 bestimmt ist, in die Definition einer fernen Warteschlange RQ1.
2. Der Warteschlangenmanager QMG1, der auf einer Windows NT-Workstation ausgeführt wird, stellt die Nachricht in die Übertragungswarteschlange XQ1.
3. Der sendende Nachrichtenkanalagent (S) auf QM1 überträgt die Nachricht unter Verwendung von TCP/IP an den empfangenden Nachrichtenkanalagenten (R) beim Kanalinitiator CHINIT2.
4. Der empfangende Nachrichtenkanalagent (R) beim Kanalinitiator CHINIT2 stellt die Nachricht in die gemeinsam genutzte Übertragungswarteschlange SYSTEM.QSG.TRANSMIT.QUEUE.

5. Der Agent der gruppeninternen Warteschlangensteuerung auf dem Warteschlangenmanager QMG3 ruft die Nachricht aus der Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE ab und stellt sie in die lokale Zielwarteschlange LQ1.
6. Die Serveranwendung ruft die Nachricht aus der lokalen Zielwarteschlange ab und verarbeitet sie.

**Nachrichtenübermittlung beim Multihopping zwischen Warteschlangenmanagern in einer Gruppe mit gemeinsamer Warteschlange**
Das vorherige Diagramm unter Migrationsunterstützung veranschaulicht außerdem die Nachrichtenübermittlung bei einem Multihopping zwischen Warteschlangenmanagern in einer Gruppe mit gemeinsamer Warteschlange. Nachrichten, die bei einem Warteschlangenmanager innerhalb der Gruppe mit gemeinsamer Warteschlange eingehen, aber für eine Warteschlange auf einem anderen Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange bestimmt sind, können mithilfe der gruppeninternen Warteschlangensteuerung ohne großen Aufwand an die Zielwarteschlange auf dem Zielwarteschlangenmanager übertragen werden.

## z/OS Einschränkungen der gruppeninternen Warteschlangensteuerung

Bei der gruppeninternen Warteschlangensteuerung bestehen folgende Einschränkungen: für die gruppeninterne Warteschlangensteuerung geeignete Nachrichten, Anzahl der Agenten der gruppeninternen Warteschlangensteuerung pro Warteschlangenmanager und das Starten bzw. Stoppen des Agenten der gruppeninternen Warteschlangensteuerung.

In diesem Abschnitt werden die Einschränkungen erläutert, die bei der gruppeninternen Warteschlangensteuerung bestehen.

**Nachrichten, die sich für die Übertragung mit der gruppeninternen Warteschlangensteuerung eignen**
Da die gruppeninterne Warteschlangensteuerung eine gemeinsame Übertragungswarteschlange nutzt, die in der Coupling-Facility (CF) definiert ist, ist die gruppeninterne Warteschlangensteuerung auf die Übermittlung von Nachrichten beschränkt, die die maximal unterstützte Nachrichtenlänge für gemeinsam genutzte Warteschlangen minus der Länge eines Übertragungswarteschlangenheaders (MQXQH) aufweisen.

**Anzahl der Agenten der gruppeninternen Warteschlangensteuerung pro Warteschlangenmanager**
In einer Gruppe mit gemeinsamer Warteschlange wird jeweils nur ein Agent der gruppeninternen Warteschlangensteuerung pro Warteschlangenmanager gestartet.

**Starten und Stoppen des Agenten der gruppeninternen Warteschlangensteuerung**
Während der Initialisierung des Warteschlangenmanagers wird ein Agent der gruppeninternen Warteschlangensteuerung gestartet und beim Herunterfahren des Warteschlangenmanagers beendet. Dabei handelt es sich naturgemäß um eine Task mit langer Ausführungsdauer, die sich (im Fall einer abnormalen Beendigung) selbst wiederherstellt. Falls im Zusammenhang mit der Definition der Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE ein Fehler auftritt (wenn beispielsweise Abrufoperationen (GET) bei dieser Warteschlange unterdrückt sind), versucht es der Agent der gruppeninternen Warteschlangensteuerung immer wieder erneut. Wenn er auf einen Fehler stößt, der zu einer normalen Beendigung des Agenten führt, solange der Warteschlangenmanager noch aktiv ist, kann er mit dem Befehl ALTER QMGR IGQ(ENABLED) erneut gestartet werden. Durch diesen Befehl muss der Warteschlangenmanager nicht gestoppt und erneut gestartet werden.

**Festlegung von ENABLED oder DISABLED für das Warteschlangenmanagerattribut**
Wenn für das Warteschlangenmanagerattribut IGQ ENABLED oder DISABLED festgelegt ist, kann es sein, dass bestehende Objektanfangspunkte mit dem Ursachencode MQRC_OBJECT_CHANGED ungültig werden. Weitere Informationen finden Sie im Abschnitt Gruppeninterne Warteschlangensteuerung - erste Schritte.

## z/OS Erste Schritte mit gruppeninterner Warteschlangensteuerung

Mithilfe der Beschreibungen in diesem Abschnitt können Sie die gruppeninterne Warteschlangensteuerung aktivieren, inaktivieren und verwenden.

**Gruppeninterne Warteschlangensteuerung aktivieren**

Zur Aktivierung der gruppeninternen Warteschlangensteuerung müssen Sie folgende Aktionen ausführen:

- Definieren Sie eine gemeinsame Übertragungswarteschlange namens SYSTEM.QSG.TRANSMIT.QUEUE. Die Definition dieser Warteschlange befindet sich in thlqual.SCSQPROC(CSQ4INSS), dem CSQINP2-Beispiel für SYSTEM-Objekte für Gruppen mit gemeinsamer Warteschlange. Diese Warteschlange muss mit den richtigen Attributen definiert werden (siehe thlqual.SCSQPROC(CSQ4INSS)), damit die gruppeninterne Warteschlangensteuerung ordnungsgemäß funktioniert.
- Da der Agent der gruppeninternen Warteschlangensteuerung bei der Initialisierung des Warteschlangenmanagers immer gestartet wird, steht die gruppeninterne Warteschlangensteuerung stets für die Verarbeitung eingehender Nachrichten zur Verfügung. Der Agent der gruppeninternen Warteschlangensteuerung verarbeitet alle Nachrichten, die in die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE gestellt werden. Zur Aktivierung der gruppeninternen Warteschlangensteuerung für die Ausgangsverarbeitung muss jedoch das Warteschlangenmanager-Attribut IGQ auf ENABLED gesetzt werden.

  **Wichtig:** Wenn für das Warteschlangenmanagerattribut IGQ ENABLED festgelegt ist, kann es sein, dass bestehende Objektanfangspunkte mit dem Ursachencode MQRC_OBJECT_CHANGED ungültig werden. Weitere Informationen finden Sie unter „Besondere Eigenschaften der gruppeninternen Warteschlangensteuerung” auf Seite 247. Wie im Abschnitt 'Programmiererantwort' für diesen Ursachencode beschrieben, müssen die Anwendungen zur Behandlung dieser Situation codiert werden. (Weitere Informationen finden Sie im Abschnitt 2041 (07F9) (RC2041): MQRC_OBJECT_CHANGED).

Da IGQ als dauerhaft laufende und sich selbst wiederherstellende Task entwickelt wurde, die bei der Initialisierung gestartet und beim Herunterfahren beendet wird, sollten Sie unbedingt auch den Abschnitt „Einschränkungen der gruppeninternen Warteschlangensteuerung” auf Seite 237 lesen.

**Gruppeninterne Warteschlangensteuerung inaktivieren**

Wenn Sie die gruppeninterne Warteschlangensteuerung für die abgehende Nachrichtenübertragung inaktivieren möchten, setzen Sie das Warteschlangenmanager-Attribut IGQ auf DISABLED. Ist die gruppeninterne Warteschlangensteuerung für einen bestimmten Warteschlangenmanager inaktiviert, kann der Agent der gruppeninternen Warteschlangensteuerung auf diesem Warteschlangenmanager nach wie vor eingehende Nachrichten verarbeiten, die von einem Warteschlangenmanager, bei dem die gruppeninterne Warteschlangensteuerung für die abgehende Übertragung aktiviert ist, in die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE gestellt wurden.

**Wichtig:** Wenn für das Warteschlangenmanagerattribut IGQ ENABLED festgelegt ist, kann es sein, dass bestehende Objektanfangspunkte mit dem Ursachencode MQRC_OBJECT_CHANGED ungültig werden. Weitere Informationen finden Sie unter „Besondere Eigenschaften der gruppeninternen Warteschlangensteuerung” auf Seite 247. Wie im Abschnitt 'Programmiererantwort' für diesen Ursachencode beschrieben, müssen die Anwendungen zur Behandlung dieser Situation codiert werden. (Weitere Informationen finden Sie im Abschnitt 2041 (07F9) (RC2041): MQRC_OBJECT_CHANGED).

Da IGQ als dauerhaft laufende und sich selbst wiederherstellende Task entwickelt wurde, die bei der Initialisierung gestartet und beim Herunterfahren beendet wird, sollten Sie unbedingt auch den Abschnitt „Einschränkungen der gruppeninternen Warteschlangensteuerung” auf Seite 237 lesen.

**Gruppeninterne Warteschlangensteuerung verwenden**

Sobald die gruppeninterne Warteschlangensteuerung aktiviert ist, steht sie zur Verwendung zur Verfügung und wird von einem Warteschlangensteuerung verwendet, sofern möglich. Wenn also eine Anwendung eine Nachricht in eine Definition einer fernen Warteschlange, in eine vollständig qualifizierte ferne Warteschlange oder in eine Clusterwarteschlange stellt, ermittelt der Warteschlangenmanager, ob sich die Nachricht für die Übermittlung mit der gruppeninternen Warteschlangensteuerung eignet; ist dies der Fall, stellt er die Nachricht in die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE. Es ist nicht erforderlich, Benutzeranwendungen zu ändern oder zu Anwendungswarteschlangen zu wechseln, da der Warteschlangenmanager bei entsprechend geeigneten Nachrichten die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE allen anderen Übertragungswarteschlangen vorzieht.

## z/OS Konfigurationen der gruppeninternen Warteschlangensteuerung

Neben der typischen Konfiguration mit der gruppeninternen Warteschlangensteuerung sind weitere Konfigurationen möglich.

In „Konzept der gruppeninternen Warteschlangensteuerung” auf Seite 232 wird die typische Konfiguration beschrieben.

**Zugehörige Konzepte**

„Verteilte Steuerung von Warteschlangen mit gruppeninterner Warteschlangensteuerung (mehrere Zustellungspfade)” auf Seite 239
Bei Anwendungen, die Kurznachrichten verarbeiten, ist die Konfiguration der gruppeninternen Warteschlangensteuerung nur für die Übermittlung von Nachrichten zwischen Warteschlangenmanagern in einer Gruppe mit gemeinsamer Warteschlange unter Umständen möglich.

„Clustering mit gruppeninterner Warteschlangensteuerung (mehrere Zustellungspfade)” auf Seite 242
Warteschlangenmanager können so konfiguriert werden, dass sie sowohl in einem Cluster als auch in einer Gruppe mit gemeinsamer Warteschlange enthalten sind.

„Clustering, gruppeninterne Warteschlangensteuerung und verteilte Steuerung von Warteschlangen” auf Seite 244
Es ist möglich, einen Warteschlangenmanager zu konfigurieren, der sowohl einem Cluster als auch einer Gruppe mit gemeinsamer Warteschlange angehört und unter Verwendung eines Sender-/Empfängerkanalpaars mit einem verteilten Warteschlangenmanager verbunden ist.

### z/OS *Verteilte Steuerung von Warteschlangen mit gruppeninterner Warteschlangensteuerung (mehrere Zustellungspfade)*

Bei Anwendungen, die Kurznachrichten verarbeiten, ist die Konfiguration der gruppeninternen Warteschlangensteuerung nur für die Übermittlung von Nachrichten zwischen Warteschlangenmanagern in einer Gruppe mit gemeinsamer Warteschlange unter Umständen möglich.

Die Auswahl der gruppeninternen Warteschlangensteuerung über Kanalübertragungen kann von der CFSTRUCT-Typstufe gesteuert werden (3 anstelle von 4 oder 5). Die maximale Nachrichtenlänge wie für die Warteschlange SYSTEM.QSQ.TRANSMIT.QUEUE festgelegt.

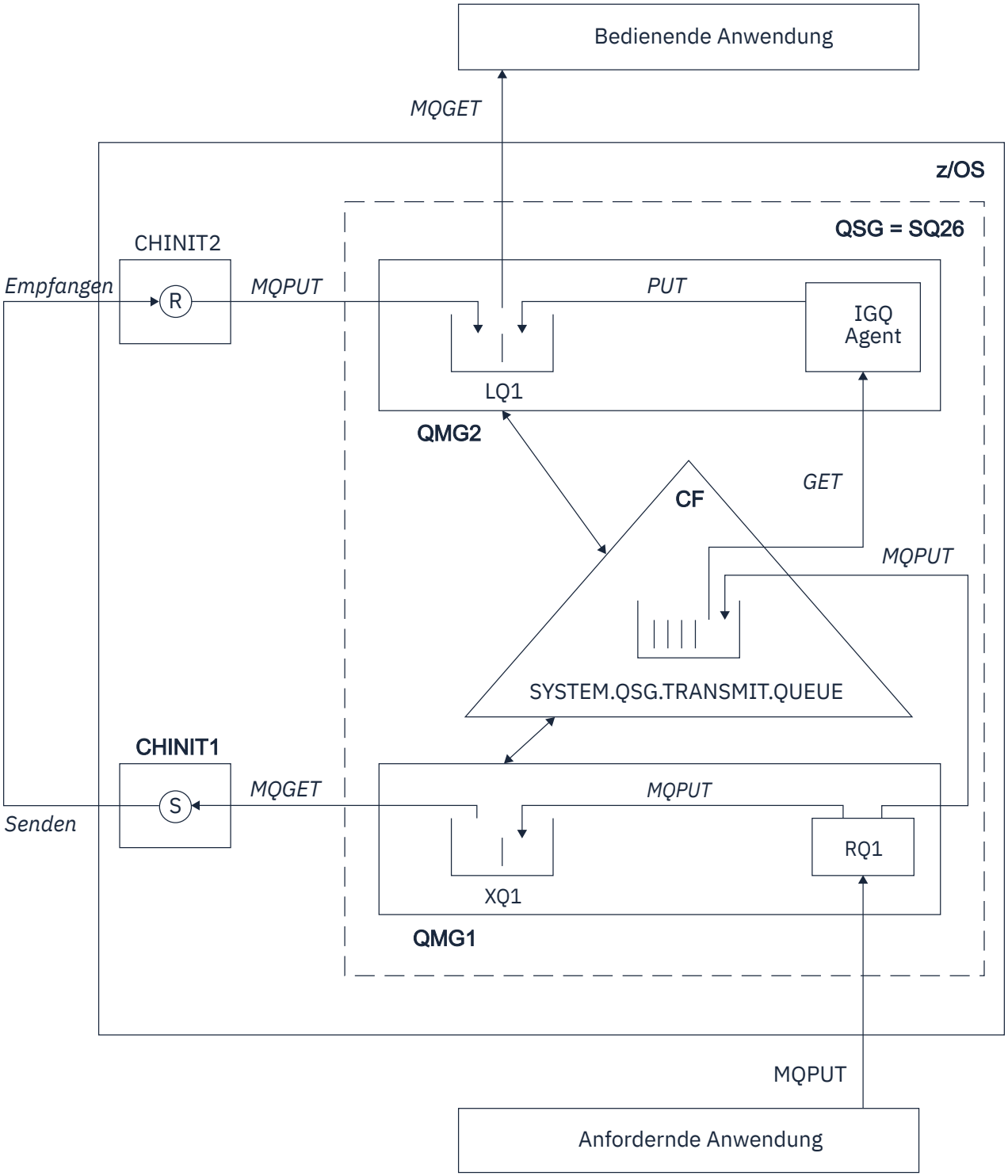



*Abbildung 69. Beispielkonfiguration*

## Verarbeitung bei Öffnungs-/Einreihungsoperationen

1. Es muss unbedingt beachtet werden, dass sowohl für die nicht gemeinsam genutzte Übertragungswarteschlange XQ1 als auch die gemeinsam genutzte Übertragungswarteschlange SYSTEM.QSG.TRANSMIT.QUEUE eine Namensauflösung erfolgt, wenn die anfordernde Anwendung die ferne Warteschlange RQ1 öffnet.
2. Wenn die anfordernde Anwendung eine Nachricht in die ferne Warteschlange einreiht, wird die Nachricht entweder in die Übertragungswarteschlange XQ1 oder in die Übertragungswarteschlange

SYSTEM.QSG.TRANSMIT.QUEUE gestellt. Dies ist abhängig davon, ob die gruppeninterne Warteschlangensteuerung für die abgehende Verarbeitung auf dem Warteschlangenmanager aktiviert ist und welche Merkmale die Nachricht aufweist. Der Warteschlangenmanager stellt alle langen Nachrichten in die Übertragungswarteschlange XQ1 und alle kurzen Nachrichten in die Übertragungswarteschlange SYSTEM.QSG.TRANSMIT.QUEUE.

3. Ist die Übertragungswarteschlange XQ1 voll oder nicht verfügbar, schlagen Einreihungsanforderungen für lange Nachrichten synchron mit einem entsprechenden Rückkehr- und Ursachencode fehl. Einreihungsanforderungen für kurze Nachrichten werden jedoch weiterhin erfolgreich verarbeitet und in die Übertragungswarteschlange SYSTEM.QSG.TRANSMIT.QUEUE gestellt.
4. Ist die Übertragungswarteschlange SYSTEM voll oder nicht für Einreihungen verfügbar, schlagen Einreihungsanforderungen für kurze Nachrichten synchron mit einem entsprechenden Rückkehr- und Ursachencode fehl. Einreihungsanforderungen für lange Nachrichten werden jedoch weiterhin erfolgreich verarbeitet und in die Übertragungswarteschlange XQ1 gestellt. In diesem Fall wird kein Versuch unternommen, die kurzen Nachrichten in eine Übertragungswarteschlange einzureihen.

## Verarbeitungsablauf bei langen Nachrichten

1. Die anfordernde Anwendung reiht lange Nachrichten in die ferne Warteschlange RQ1 ein.
2. Der Warteschlangenmanager QMG1 reiht die Nachrichten in die Übertragungswarteschlange XQ1 ein.
3. Der sendende Nachrichtenkanalagent (S) auf dem Warteschlangenmanager QMG1 ruft die Nachrichten aus der Übertragungswarteschlange XQ1 ab und sendet diese an den Warteschlangenmanager QMG2.
4. Der empfangende Nachrichtenkanalagent (R) auf dem Warteschlangenmanager QMG2 empfängt die Nachrichten und stellt diese in die Zielwarteschlange LQ1.
5. Die bereitstellende Anwendung ruft dann die Nachrichten aus der Warteschlange LQ1 ab und verarbeitet diese.

## Verarbeitungsablauf bei kurzen Nachrichten

1. Die anfordernde Anwendung reiht kurze Nachrichten in die ferne Warteschlange RQ1 ein.
2. Der Warteschlangenmanager QMG1 reiht die Nachrichten in die Übertragungswarteschlange SYSTEM.QSG.TRANSMIT.QUEUE ein.
3. Die gruppeninterne Warteschlangensteuerung auf dem Warteschlangenmanager QMG2 ruft die Nachrichten ab und stellt diese in die Zielwarteschlange LQ1.
4. Die bereitstellende Anwendung ruft die Nachrichten aus der Warteschlange LQ1 ab.

## Wichtige Aspekte

1. Die anfordernde Anwendung muss den zugrunde liegenden Mechanismus, der für die Nachrichtenübermittlung verwendet wird, nicht kennen.
2. Bei kurzen Nachrichten kann ein möglicherweise schnellerer Mechanismus zur Nachrichtenübermittlung erreicht werden.
3. Für die Nachrichtenübermittlung stehen mehrere Pfade zur Verfügung (also die normale Kanalroute und die Route der gruppeninternen Warteschlangensteuerung).
4. Da die Route der gruppeninternen Warteschlangensteuerung meist schneller ist, wird sie der normalen Kanalroute vorgezogen. Abhängig von den Nachrichtenmerkmalen kann die Nachrichtenübermittlung auf die beiden Pfade aufgeteilt werden. Nachrichten werden also möglicherweise in falscher Reihenfolge zugestellt (diese Übermittlung ist jedoch auch möglich, wenn Nachrichten nur unter Verwendung der normalen Kanalroute übermittelt werden).
5. Wurde eine Route ausgewählt und wurden die Nachricht in die Übertragungswarteschlangen gestellt, wird für die Nachrichtenübermittlung nur die ausgewählte Route verwendet. Nicht verarbeitete Nachrichten in der Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE werden nicht an die Übertragungswarteschlange XQ1 umgeleitet.

### z/OS *Clustering mit gruppeninterner Warteschlangensteuerung (mehrere Zustellungspfade)*

Warteschlangenmanager können so konfiguriert werden, dass sie sowohl in einem Cluster als auch in einer Gruppe mit gemeinsamer Warteschlange enthalten sind.

Werden Nachrichten an eine Clusterwarteschlange gesendet und befinden sich die lokalen und fernen Zielwarteschlangenmanager in derselben Gruppe mit gemeinsamer Warteschlange, wird die gruppeninterne Warteschlangensteuerung (mit der Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE) für die Übermittlung kurzer Nachrichten verwendet und für die Übermittlung langer Nachrichten, wenn die gruppeninterne Warteschlangensteuerung die Größe der Nachricht unterstützt. Außerdem wird die Warteschlange SYSTEM.CLUSTER.TRANSMIT.QUEUE für die Nachrichtenübermittlung an Warteschlangenmanager verwendet, die sich im Cluster befinden, jedoch nicht der Gruppe mit gemeinsamer Warteschlange angehören. Diese Konfiguration wird im folgenden Diagramm veranschaulicht (die Kanalinitiatoren sind nicht im Diagramm enthalten).

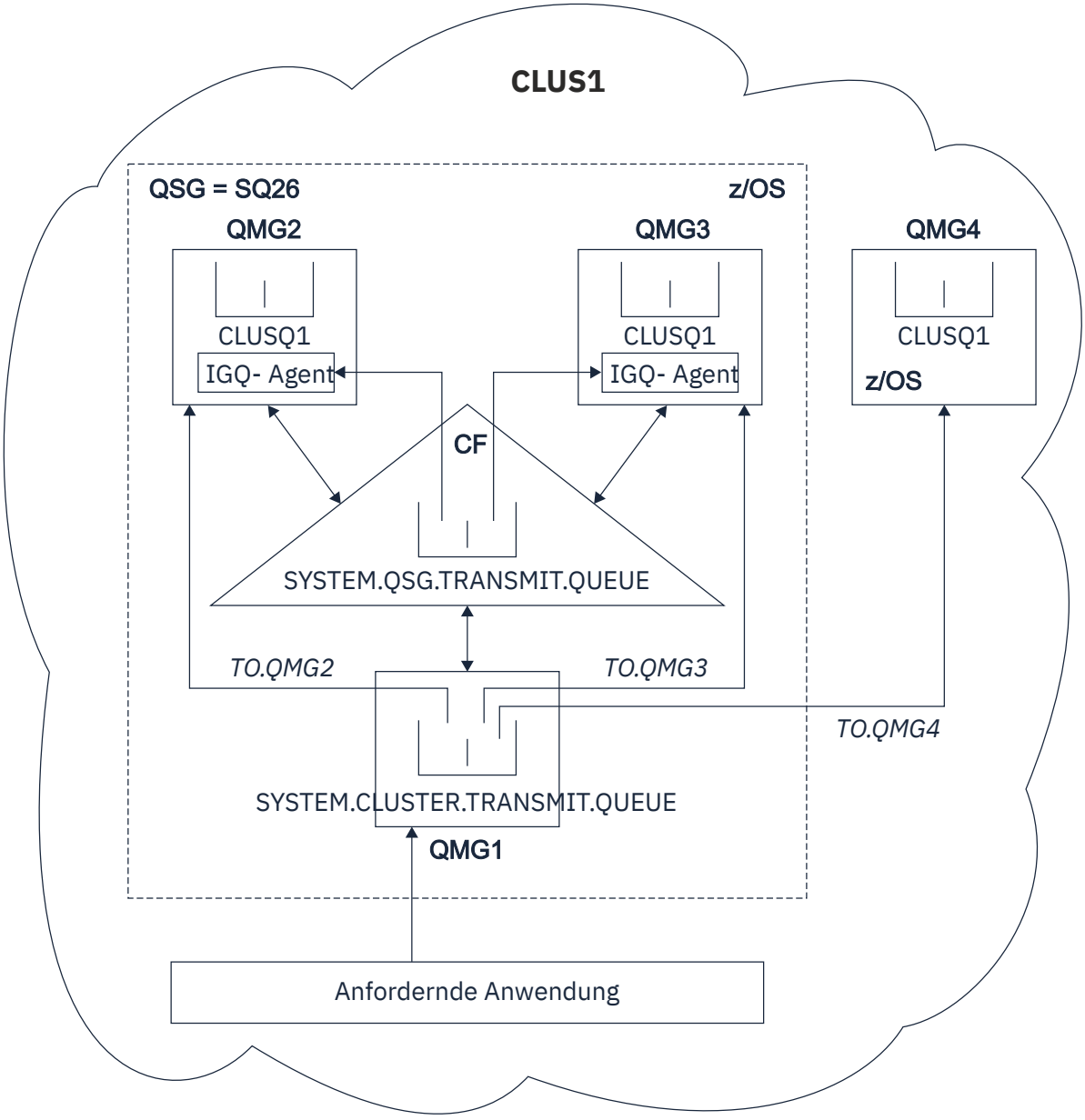



*Abbildung 70. Beispiel für ein Clustering mit der gruppeninternen Warteschlangensteuerung*

Das Diagramm enthält folgende Punkte:

- Die vier z/OS-Warteschlangenmanager QMG1, QMG2, QMG3 und QMG4, die im Cluster CLUS1 konfiguriert sind.
- Die Warteschlangenmanager QMG1, QMG2 und QMG3, die in der Gruppe SQ26 mit gemeinsamer Warteschlange konfiguriert sind.
- Agenten der gruppeninternen Warteschlangensteuerung, die auf den Warteschlangenmanagern QMG2 und QMG3 ausgeführt werden.
- Die im QMG1 definierte lokale Warteschlange SYSTEM.CLUSTER.TRANSMIT.QUEUE.

  **Anmerkung:** Zur besseren Übersichtlichkeit wird die SYSTEM.CLUSTER.TRANSMIT.QUEUE auf den anderen Warteschlangenmanagern nicht angezeigt.
- Das gemeinsam genutzte SYSTEM.QSG.TRANSMIT.QUEUE in der CF definiert, die sich in einer IBM MQ -Struktur befindet, die mit dem Attribut CFLEVEL (3) RECOVER (YES) konfiguriert ist.
- Die Clusterkanäle TO.QMG2 (für die Verbindung von QMG1 mit QMG2), TO.QMG3 (für die Verbindung von QMG1 mit QMG3) und TO.QMG4 (für die Verbindung von QMG1 mit QMG4).
- Die Clusterwarteschlange CLUSQ1, die auf den Warteschlangenmanagern QMG2, QMG3 und QMG4 gehostet werden.

Angenommen, die anfordernde Anwendung öffnet die Clusterwarteschlange mit der Option MQOO_BIND_NOT_FIXED, sodass zum Zeitpunkt der Einreihung der Ziel-Warteschlangenmanager für die Clusterwarteschlange ausgewählt wird.

Wenn als Ziel-Warteschlangenmanager QMG2 ausgewählt wird, gilt Folgendes:

- Alle langen Nachrichten, die von der anfordernden Anwendung eingereiht werden, werden
  - in die Warteschlange SYSTEM.CLUSTER.TRANSMIT.QUEUE auf QMG1 eingereiht, weil sich SYSTEM.QSG.TRANSMIT.QUEUE in einer CFLEVEL(3)-Struktur befindet; deshalb werden nur Nachrichten mit einer Größe von maximal 63 KB unterstützt.
  - unter Verwendung des Clusterkanals TO.QMG2 an die Clusterwarteschlange CLUSQ1 auf QMG2 übertragen
- Alle kurzen Nachrichten, die von der anfordernden Anwendung eingereiht werden, werden
  - in die gemeinsam genutzte Übertragungswarteschlange SYSTEM.QSG.TRANSMIT.QUEUE eingereiht. Diese Warteschlange befindet sich in einer mit dem Attribut RECOVER(YES) konfigurierten Struktur und wird daher sowohl für persistente als auch nicht persistente kurze Nachrichten verwendet.
  - vom Agenten der gruppeninternen Warteschlangensteuerung auf QMG2 abgerufen
  - in die Clusterwarteschlange CLUSQ1 auf QMG2 eingereiht

Wenn als Ziel-Warteschlangenmanager QMG4 ausgewählt wird, gilt Folgendes:

- Da QMG4 kein Mitglied der Gruppe SQ26 mit gemeinsamer Warteschlange ist, werden alle von der anfordernden Anwendung eingereihten Nachrichten
  - in die Warteschlange SYSTEM.CLUSTER.TRANSMIT.QUEUE auf QMG1 eingereiht
  - unter Verwendung des Clusterkanals TO.QMG4 an die Clusterwarteschlange CLUSQ1 auf QMG4 übertragen

## Wichtige Aspekte

- Die anfordernde Anwendung muss den zugrunde liegenden Mechanismus, der für die Nachrichtenübermittlung verwendet wird, nicht kennen.
- Für die Übertragung kurzer nicht persistenter Nachrichten zwischen Warteschlangenmanagern in einer Gruppe mit gemeinsamer Warteschlange wird ein wahrscheinlich schnellerer Übermittlungsmechanismus erreicht (dies gilt auch, wenn sich dieselben Warteschlangenmanager in einem Cluster befinden).
- Für die Nachrichtenübermittlung stehen mehrere Pfade zur Verfügung (also die Clusterroute und die Route der gruppeninternen Warteschlangensteuerung).

- Da die Route der gruppeninternen Warteschlangensteuerung meist schneller ist, wird sie der Clusterroute vorgezogen. Abhängig von den Nachrichtenmerkmalen kann die Nachrichtenübermittlung auf die beiden Pfade aufgeteilt werden. Nachrichten werden also möglicherweise in falscher Reihenfolge zugestellt. Sie müssen unbedingt beachten, dass diese Übermittlung ohne Berücksichtigung der von der Anwendung angegebenen Option MQOO_BIND_* möglich ist. Bei der gruppeninternen Warteschlangensteuerung werden Nachrichten genau wie beim Clustering verteilt, je nachdem, ob beim Öffnen MQOO_BIND_NOT_FIXED, MQOO_BIND_ON_OPEN, MQOO_BIND_ON_GROUP oder MQOO_BIND_AS_Q_DEF angegeben ist.
- Wurde eine Route ausgewählt und wurden die Nachricht in die Übertragungswarteschlangen gestellt, wird für die Nachrichtenübermittlung nur die ausgewählte Route verwendet. Nicht verarbeitete Nachrichten in der Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE werden nicht an die Übertragungswarteschlange SYSTEM.CLUSTER.TRANSMIT.QUEUE umgeleitet.

### z/OS *Clustering, gruppeninterne Warteschlangensteuerung und verteilte Steuerung von Warteschlangen*

Es ist möglich, einen Warteschlangenmanager zu konfigurieren, der sowohl einem Cluster als auch einer Gruppe mit gemeinsamer Warteschlange angehört und unter Verwendung eines Sender-/Empfängerkanalpaars mit einem verteilten Warteschlangenmanager verbunden ist.

Diese Konfiguration ist eine Kombination aus einer verteilten Steuerung von Warteschlangen mit gruppeninterner Warteschlangensteuerung und einem Clustering mit gruppeninterner Warteschlangensteuerung.

Die gruppeninterne Warteschlangensteuerung wird im Abschnitt „Verteilte Steuerung von Warteschlangen mit gruppeninterner Warteschlangensteuerung (mehrere Zustellungspfade)” auf Seite 239 beschrieben.

Das Clustering mit gruppeninterner Warteschlangensteuerung wird im Abschnitt „Clustering mit gruppeninterner Warteschlangensteuerung (mehrere Zustellungspfade)” auf Seite 242 beschrieben.

## z/OS Nachrichten der gruppeninternen Warteschlangensteuerung

In diesem Abschnitt werden die Nachrichten beschrieben, die in die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE eingereiht werden.

**Nachrichtenstruktur**

Wie alle anderen Nachrichten, die in Übertragungswarteschlangen eingereiht werden, erhalten Nachrichten, die in die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE eingereiht werden, das Präfix des Headers der Übertragungswarteschlange (MQXQH).

**Nachrichtenpersistenz**

In IBM WebSphere MQ 5.3 und höher werden von gemeinsam genutzten Warteschlangen sowohl persistente als auch nicht persistente Nachrichten unterstützt.

Wird der Warteschlangenmanager beendet, solange der Agent der gruppeninternen Warteschlangensteuerung nicht persistente Nachrichten verarbeitet oder wird der Agent der gruppeninternen Warteschlangensteuerung abnormal beendet, solange er noch Nachrichten verarbeitet, können nicht persistente Nachrichten unter Umständen verloren gehen. Anwendungen müssen Vorkehrungen für die Wiederherstellung nicht persistenter Nachrichten treffen, falls deren Wiederherstellung erforderlich wird.

Schlägt eine vom Agenten der gruppeninternen Warteschlangensteuerung ausgegebene Einreihungsanforderung unerwartet fehl, geht die gerade verarbeitete Nachricht verloren.

**Nachrichtenübermittlung**

Der Agent der gruppeninternen Warteschlangensteuerung ruft alle nicht persistenten Nachrichten außerhalb des Synchronisationspunktbereichs und alle persistenten Nachrichten innerhalb des Synchronisationspunktbereichs ab und übermittelt diese. In diesem Fall agiert der Agent der gruppeninternen Warteschlangensteuerung als Koordinator des Synchronisationspunkts. Der Agent der gruppeninternen Warteschlangensteuerung verarbeitet daher nicht persistente Nachrichten genauso, wie schnelle nicht persistente Nachrichten in einem Nachrichtenkanal verarbeitet werden. Weitere Informationen finden Sie unter Schnelle nicht persistente Nachrichten.

**Stapelverarbeitung von Nachrichten**
Der Agent der gruppeninternen Warteschlangensteuerung verwendet eine feste Stapelgröße von 50 Nachrichten. Alle persistenten Nachrichten, die innerhalb eines Stapels abgerufen werden, werden in Intervallen von 50 Nachrichten festgeschrieben. Wenn in der Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE keine Nachrichten mehr für den Abruf zur Verfügung stehen, schreibt der Agent einen Stapel fest, der aus persistenten Nachrichten besteht.

**Nachrichtengröße**
Die maximale Größe der Nachricht, die in die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE eingereiht werden kann, entspricht der maximal unterstützten Nachrichtenlänge für gemeinsam genutzte Warteschlangen minus der Länge eines Headers der Übertragungswarteschlange (MQXQH).

**Standardmäßige Nachrichtenpersistenz und standardmäßige Nachrichtenpriorität**
Wenn sich die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE in dem Auflösungspfad für Warteschlangennamen befindet, der zum Zeitpunkt der Öffnung eingerichtet war, gelten für Nachrichten, die mit einer Standardpersistenz und Standardpriorität (bzw. mit einer Standardpersistenz oder Standardpriorität) eingereiht werden, die normalen Regeln bei der Auswahl der Warteschlange, die verwendete Werte für die Standardpriorität und -persistenz aufweist. (Der Abschnitt IBM MQ-Nachrichten enthält weitere Informationen zu den Regeln der Warteschlangenauswahl.)

**Zugehörige Konzepte**

„Nicht zugestellte/nicht verarbeitete Nachrichten” auf Seite 245
In diesem Abschnitt wird beschrieben, was mit nicht zugestellten und nicht verarbeiteten Nachrichten in der Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE geschieht.

„Berichtsnachrichten - Gruppeninterne Warteschlangensteuerung” auf Seite 246
In diesem Abschnitt werden die Berichtsnachrichten beschrieben: Eingangsbestätigung, Zustellungsbestätigung, Verfallsbericht und Ausnahmebericht.

## z/OS *Nicht zugestellte/nicht verarbeitete Nachrichten*

In diesem Abschnitt wird beschrieben, was mit nicht zugestellten und nicht verarbeiteten Nachrichten in der Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE geschieht.

Wenn ein Agent der gruppeninternen Warteschlangensteuerung keine Nachricht an die Zielwarteschlange zustellen kann, verhält er sich wie folgt:

- Er berücksichtigt die Berichtsoption MQRO_DISCARD_MSG (wenn er durch das Feld mit den Berichtsoptionen des MQMD für die nicht zugestellte Nachricht entsprechend angewiesen wird) und löscht die nicht zugestellte Nachricht.
- Wenn die Nachricht noch nicht gelöscht wurde, versucht er, die nicht zugestellte Nachricht in die Warteschlange für nicht zustellbare Nachrichten für den Zielwarteschlangenmanager zu stellen. Der Agent der gruppeninternen Warteschlangensteuerung stellt der Nachricht das Präfix des Warteschlangenheaders für nicht zustellbare Nachrichten (MQDLH) voran.

Wenn keine Warteschlange für nicht zustellbare Nachrichten definiert ist oder eine nicht zugestellte Nachricht nicht in die Warteschlange für nicht zustellbare Nachrichten eingereiht werden kann und die Nachricht

- persistent ist, setzt der Agent der gruppeninternen Warteschlangensteuerung den aktuellen Stapel der persistenten Nachrichten zurück, die gerade von ihm verarbeitet werden, und unternimmt einen Neuversuch. Weitere Informationen finden Sie unter „Besondere Eigenschaften der gruppeninternen Warteschlangensteuerung” auf Seite 247.
- nicht persistent ist, löscht der Agent der gruppeninternen Warteschlangensteuerung die Nachricht und fährt mit der Verarbeitung der nächsten Nachricht fort.

Wenn ein Warteschlangenmanager in einer Gruppe mit gemeinsamer Warteschlange beendet wird, bevor sein zugehöriger Agent der gruppeninternen Warteschlangensteuerung die Verarbeitung all seiner Nachrichten abgeschlossen hat, verbleiben die nicht verarbeiteten Nachrichten in der Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE, bis der Warteschlangenmanager das nächste Mal gestartet wird. Der Agent der gruppeninternen Warteschlangensteuerung ruft dann die Nachrichten ab und stellt sie der Zielwarteschlange zu.

Schlägt die Coupling-Facility fehl, bevor alle Nachrichten in der Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE verarbeitet wurden, gehen alle nicht persistenten Nachrichten verloren, die bis zu diesem Zeitpunkt nicht verarbeitet wurden.

IBM empfiehlt, dass Nachrichten nicht von den Anwendungen direkt in Übertragungswarteschlangen eingereiht werden sollen. Reiht eine Anwendung dennoch Nachrichten direkt in die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE ein, kann der Agent der gruppeninternen Warteschlangensteuerung diese Nachrichten möglicherweise nicht verarbeiten und sie verbleiben in der Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE. Benutzer müssen diese nicht verarbeiteten Nachrichten dann mit ihren eigenen Methoden handhaben.

### z/OS *Berichtsnachrichten - Gruppeninterne Warteschlangensteuerung*

In diesem Abschnitt werden die Berichtsnachrichten beschrieben: Eingangsbestätigung, Zustellungsbestätigung, Verfallsbericht und Ausnahmebericht.

**Berichtsnachrichten über die Eingangsbestätigung/Zustellungsbestätigung**

Eingangs- und Zustellungsbestätigungsnachrichten werden vom Warteschlangenmanager generiert, wenn die gruppeninterne Warteschlangensteuerung verwendet wird.

**Berichtsnachrichten über den Verfall**

Der Warteschlangenmanager generiert Verfallsberichtsnachrichten.

**Berichtsnachrichten über Ausnahmebedingungen**

Abhängig davon, welche Berichtsoption MQRO_EXCEPTION_* im Feld *Berichtsoptionen* des Nachrichtendeskriptors für die nicht zugestellte Nachricht angegeben ist, generiert der Agent der gruppeninternen Warteschlangensteuerung den erforderlichen Ausnahmebericht und stellt diesen in die angegebene Empfangswarteschlange für Antworten. Mit der gruppeninternen Warteschlangensteuerung kann der Ausnahmebericht an die Zielempfangswarteschlange für Antworten zugestellt werden.

Die Persistenz der Berichtsnachricht entspricht der Persistenz der nicht zugestellten Nachricht. Wenn der Agent der gruppeninternen Warteschlangensteuerung den Namen der Zielempfangswarteschlange für Antworten nicht auflösen kann oder er die Antwortnachricht nicht in eine Übertragungswarteschlange einreihen kann (für die nachfolgende Übertragung an die Zielempfangswarteschlange für Antworten), versucht er, den Ausnahmebericht in die Warteschlange für nicht zustellbare Nachrichten des Warteschlangenmanagers einzureihen, auf dem die Berichtsnachricht generiert wird. Falls dies nicht möglich ist, ist die nicht zugestellte Nachricht

- persistent, der Agent der gruppeninternen Warteschlangensteuerung löscht den Ausnahmebericht, setzt den aktuellen Stapel der Nachrichten zurück und unternimmt einen Neuversuch. Weitere Informationen finden Sie unter „Besondere Eigenschaften der gruppeninternen Warteschlangensteuerung” auf Seite 247.
- nicht persistent, der Agent der gruppeninternen Warteschlangensteuerung löscht den Ausnahmebericht und fährt mit der Verarbeitung der nächsten Nachricht in der Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE fort.

## z/OS Sicherheit für gruppeninterne Warteschlangensteuerung

In diesem Abschnitt werden die Sicherheitsvorkehrungen bei der gruppeninternen Warteschlangensteuerung beschrieben.

Die Warteschlangenmanager-Attribute IGQAUT (Berechtigung eine Agenten der gruppeninternen Warteschlangensteuerung) und IGQUSER (Benutzer-ID des Agenten der gruppeninternen Warteschlangensteuerung) können festgelegt werden, um die Stufe der Sicherheitsprüfung zu steuern, die ausgeführt wird, wenn der Agent der gruppeninternen Warteschlangensteuerung Zielwarteschlangen öffnet.

**Berechtigung der gruppeninternen Warteschlangensteuerung (IGQAUT)**

Das IGQAUT-Attribut kann festgelegt werden, um die Art der auszuführenden Sicherheitsprüfungen festzulegen. Damit werden die Benutzer-IDs bestimmt, die vom Agenten der gruppeninternen Warteschlangensteuerung verwendet werden sollen, wenn er die Berechtigung zum Einreihen von Nachrichten in die Zielwarteschlange einrichtet.

Das Attribut IGQAUT entspricht dem Attribut PUTAUT, das in Kanaldefinitionen verfügbar ist.

**Benutzer-ID der gruppeninternen Warteschlangensteuerung (IGQUSER)**

Mit dem Attribut IGQUSER kann eine Benutzer-ID angegeben werden, die vom Agenten der gruppeninternen Warteschlangensteuerung verwendet werden soll, wenn er die Berechtigung zum Einreihen von Nachrichten in eine Zielwarteschlange einrichtet.

Das Attribut IGQUSER entspricht dem Attribut MCAUSER, das in Kanaldefinitionen verfügbar ist.

## z/OS Besondere Eigenschaften der gruppeninternen Warteschlangensteuerung

In diesem Abschnitt werden die besonderen Eigenschaften der gruppeninternen Warteschlangensteuerung beschrieben, einschließlich Invalidierung von Objektkennungen, eigene Wiederherstellung und Neuversuchsfunktion des Agenten für gruppeninterne Warteschlangensteuerung sowie der Agent für gruppeninterne Warteschlangensteuerung und Serialisierung.

### Invalidierung von Objektkennungen (MQRC_OBJECT_CHANGED)

Wenn nach dem Öffnen des Objekts festgestellt wird, dass sich die Attribute eines Objekts geändert haben, macht der Warteschlangenmanager die Objektkennung mit MQRC_OBJECT_CHANGED bei seiner nächsten Verwendung ungültig.

In der gruppeninternen Warteschlangensteuerung gelten folgende Regeln für die Invalidierung von Objektkennungen:

- Wenn die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE während der Öffnungsverarbeitung im Pfad der Namensauflösung enthalten war, da die gruppeninterne Warteschlangensteuerung zum Zeitpunkt der Öffnung aktiviert war (ENABLED), die gruppeninterne Warteschlangensteuerung bei der Einreihung jedoch inaktiviert (DISABLED) war, macht der Warteschlangenmanager die Objektkennung ungültig und gibt für die Einreihungsanforderung mit MQRC_OBJECT_CHANGED einen Fehler aus.
- Wenn die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE während der Öffnungsverarbeitung im Pfad der Namensauflösung nicht enthalten war, da die gruppeninterne Warteschlangensteuerung zum Zeitpunkt der Öffnung inaktiviert war (DISABLED), die gruppeninterne Warteschlangensteuerung bei der Einreihung jedoch aktiviert (ENABLED) war, macht der Warteschlangenmanager die Objektkennung ungültig und gibt für die Einreihungsanforderung mit MQRC_OBJECT_CHANGED einen Fehler aus.
- Wenn die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE während der Öffnungsverarbeitung im Pfad der Namensauflösung enthalten war, da die gruppeninterne Warteschlangensteuerung zum Zeitpunkt der Öffnung aktiviert war, die Definition von SYSTEM.QSG.TRANSMIT.QUEUE bei der Einreihung jedoch geändert war, macht der Warteschlangenmanager die Objektkennung ungültig und gibt für die Einreihungsanforderung mit MQRC_OBJECT_CHANGED einen Fehler aus.

### Eigene Wiederherstellung des Agenten der gruppeninternen Warteschlangensteuerung

Wenn der Agent der gruppeninternen Warteschlangensteuerung abnormal beendet wird, wird die Nachricht CSQM067E ausgegeben, und der Agent der gruppeninternen Warteschlangensteuerung wird erneut gestartet.

### Neuversuchsfunktion des Agenten der gruppeninternen Warteschlangensteuerung

Wenn der Agent der gruppeninternen Warteschlangensteuerung beim Zugriff auf die Warteschlange SYSTEM.QSG.TRANSMIT.QUEUE ein Problem feststellt (da diese beispielsweise nicht definiert oder mit den falschen Attributen definiert ist, in ihr Einreihungen unterdrückt sind oder ein anderer Grund vorliegt), unternimmt der Agent der gruppeninternen Warteschlangensteuerung einen Neuversuch.

Der Agent der gruppeninternen Warteschlangensteuerung beobachtet kurze und lange Wiederholungszähler und -intervalle. Diese unveränderlichen Werte für Zähler und Intervalle lauten wie folgt:

| *Tabelle 20. Werte für kurze und lange Wiederholungszähler und -intervalle* | |
|---|---|
| **Konstant** | **Wert** |
| Zähler für kurze Wiederholungsversuche | 10 |
| Intervall für kurze Wiederholungsversuche | 60 seconds = 1 min |
| Zähler für lange Wiederholungsversuche | 999,999,999 |
| Intervall für lange Wiederholungsversuche | 1200 seconds = 20 min |

### Agent für gruppeninterne Warteschlangensteuerung und Serialisierung

Der Versuch vom Agenten der gruppeninternen Warteschlangensteuerung, den Zugriff auf die gemeinsam genutzten Warteschlangen während einer gerade ausgeführten Peerwiederherstellung zu serialisieren, schlägt möglicherweise fehl.

Wenn ein Warteschlangenmanager in einer Gruppe mit gemeinsamer Warteschlange fehlschlägt, solange der Agent der gruppeninternen Warteschlangensteuerung nicht festgeschriebene Nachrichten in gemeinsam genutzten Warteschlangen verarbeitet, wird der Agent der gruppeninternen Warteschlangensteuerung beendet und die Peerwiederherstellung der gemeinsam genutzten Warteschlange wird für den fehlschlagenden Warteschlangenmanager ausgeführt. Da es sich bei der Peerwiederherstellung der gemeinsam genutzten Warteschlange um eine asynchrone Aktivität handelt, können der fehlschlagende Warteschlangenmanager und auch der Agent der gruppeninternen Warteschlangensteuerung für diesen Warteschlangenmanager bereits neu gestartet werden, wenn die Peerwiederherstellung der gemeinsam genutzten Warteschlange noch nicht abgeschlossen ist. Dies ermöglicht wiederum die außerordentliche Vorabverarbeitung von festgeschriebenen Nachrichten, solange die Nachrichten noch wiederhergestellt werden. Um sicherzustellen, dass die Nachrichten nicht in falscher Reihenfolge verarbeitet werden, serialisiert der Agent der gruppeninternen Warteschlangensteuerung gemeinsam genutzte Warteschlangen, indem er den API-Aufruf MQCONNX ausgibt.

Der Versuch vom Agenten der gruppeninternen Warteschlangensteuerung, den Zugriff auf die gemeinsam genutzten Warteschlangen während einer gerade ausgeführten Peerwiederherstellung zu serialisieren, schlägt möglicherweise fehl. Es wird eine Fehlernachricht ausgegeben und der Agent der gruppeninternen Warteschlangensteuerung wird in den Neuversuchsstatus versetzt. Sobald die Peerwiederherstellung des Warteschlangenmanagers abgeschlossen ist (beispielsweise beim nächsten Neuversuch), kann der Agent der gruppeninternen Warteschlangensteuerung gestartet werden.

## z/OS Speicherverwaltung unter z/OS

IBM MQ for z/OS erfordert permanente und temporäre Datenstrukturen und speichert diese Dateien in Seitengruppen und Hauptspeicherpuffern. In den folgenden Abschnitten wird ausführlich erläutert, wie IBM MQ diese Seitengruppen und Puffer verwendet.

**Zugehörige Konzepte**

„Seitengruppen für IBM MQ for z/OS" auf Seite 249
In diesem Abschnitt wird erläutert, wie IBM MQ for z/OS Seitengruppen zum Speichern von Nachrichten verwendet.

„Speicherklassen für IBM MQ for z/OS" auf Seite 250
Eine Speicherklasse ist ein IBM MQ for z/OS-Konzept, das dem Warteschlangenmanager die Zuordnung zwischen Warteschlangen und Seitengruppen ermöglicht. Mithilfe von Speicherklassen können Sie steuern, welche Dateien von welchen Warteschlangen verwendet werden.

„Puffer und Pufferpools für IBM MQ for z/OS" auf Seite 251
In IBM MQ for z/OS werden Puffer und Pufferpools zur vorübergehenden Zwischenspeicherung von Daten verwendet. In diesem Abschnitt wird erläutert, wie Puffer organisiert und verwendet werden.

**Zugehörige Verweise**

„Weitere Informationen zur Speicherverwaltung für IBM MQ for z/OS" auf Seite 253

In diesem Abschnitt finden Sie Verweise auf weitere Informationen zur Speicherverwaltung in IBM MQ for z/OS.

## z/OS Seitengruppen für IBM MQ for z/OS

In diesem Abschnitt wird erläutert, wie IBM MQ for z/OS Seitengruppen zum Speichern von Nachrichten verwendet.

Eine *Seitengruppe* ist eine lineare VSAM-Datei (Virtual Storage Access Method), die speziell für die Verwendung durch IBM MQ formatiert wurde. In Seitengruppen werden die meisten Nachrichten und Objektdefinitionen gespeichert.

Ausnahmen davon sind globale Definitionen, die in einem gemeinsam genutzten Repository in Db2 gespeichert werden, sowie Nachrichten in gemeinsam genutzten Warteschlangen. Diese Elemente werden nicht in den Seitengruppen eines Warteschlangenmanagers gespeichert. Weitere Informationen zu gemeinsam genutzten Warteschlangen finden Sie unter „Gemeinsam genutzte Warteschlangen und Gruppen mit gemeinsamer Warteschlange” auf Seite 181. Der Abschnitt Private und globale Definitionen enthält weitere Informationen zu globalen Definitionen.

IBM MQ-Seitengruppen können eine Größe von maximal 64 GB haben. Jede Seitengruppe wird durch eine Seitengruppen-ID (PSID) identifiziert, die einer ganzen Zahl zwischen 00 und 99 entspricht. Jeder Warteschlangenmanager muss über eigene Seitengruppen verfügen.

In IBM MQ werden Objektdefinitionen und andere wichtige Informationen, die für den Warteschlangenmanager relevant sind, in der Seitengruppe 0 (PSID=00) gespeichert. Um den normalen Betrieb von IBM MQ zu gewährleisten, dürfen keinerlei Nachrichten in der Seitengruppe 0 gespeichert werden, damit sie auf keinen Fall voll wird.

Um die Leistung des Systems zu verbessern, sollten Sie darüber hinaus kurzlebige Nachrichten von langlebigen trennen, indem Sie sie in unterschiedlichen Seitengruppen speichern.

Für die Formatierung von Seitengruppen, die unbedingt erforderlich ist, stellt IBM MQ das Dienstprogramm FORMAT zur Verfügung. Weitere Informationen finden Sie im Abschnitt Seitengruppen formatieren (FORMAT). Darüber hinaus müssen Seitengruppen für das IBM MQ-Subsystem definiert werden.

Sie können in der Konfiguration von IBM MQ for z/OS angeben, dass eine Seitengruppe dynamisch erweitert wird, sobald sie voll ist. IBM MQ erweitert weiterhin die Seitengruppe, wenn erforderlich, bis 123 logische Speicherbereiche vorhanden sind, wenn genügend Plattenspeicherbereich verfügbar ist. Die Speicherbereiche können auf mehrere Datenträger aufgeteilt werden, wenn die lineare Datei entsprechend definiert ist. IBM MQ kann Seitengruppen jedoch nur bis zu einer maximalen Größe von 64 GB erweitern.

Sie können die Seitengruppen eines IBM MQ-Warteschlangenmanagers nicht auf einem anderen IBM MQ-Warteschlangenmanager verwenden oder den Namen des Warteschlangenmanagers ändern. Wenn Sie die Daten von einem Warteschlangenmanager zu einem anderen übertragen möchten, müssen Sie alle Objekte und Nachrichten von dem einen Warteschlangenmanager herunterladen und erneut auf den anderen Warteschlangenmanager hochladen.

Auf einem Warteschlangenmanager mit einem früheren Release als Version 6 können keine Seitengruppen mit einer Größe von mehr als 4 GB verwendet werden. Während der Migrationsphase, wenn Sie möglicherweise noch einmal zu einem früheren Coderelease zurückkehren müssen, gilt Folgendes:

- Ändern Sie die Größe der Seitengruppe null (PSID=00) nicht auf einen Wert größer als 4 GB.
- Beim Neustart eines Warteschlangenmanagers mit einem früheren Release bleiben andere Seitengruppen mit einer Größe von mehr als 4 GB offline.

Weitere Informationen zur Migration vorhandener Seitengruppen, die auf eine Größe von mehr als 4 GB erweitert werden können, finden Sie unter Seitengruppen mit einer Größe von mehr als 4 GB definieren.

Administratoren können einem aktiven Warteschlangenmanager dynamisch Seitengruppen hinzufügen oder von ihm entfernen (mit Ausnahme von Seitengruppe null). Der Befehl 'DEFINE PSID' kann nach Abschluss des Warteschlangenmanager-Neustarts nur dann ausgeführt werden, wenn im Befehl das Schlüsselwort 'DSN' angegeben wird.

# z/OS Speicherklassen für IBM MQ for z/OS

Eine Speicherklasse ist ein IBM MQ for z/OS-Konzept, das dem Warteschlangenmanager die Zuordnung zwischen Warteschlangen und Seitengruppen ermöglicht. Mithilfe von Speicherklassen können Sie steuern, welche Dateien von welchen Warteschlangen verwendet werden.

## Speicherklassen - Einführung

Mithilfe einer *Speicherklasse* werden einer Seitengruppe eine oder mehrere Warteschlangen zugeordnet. Dies bedeutet, dass Nachrichten, die für diese Warteschlange bestimmt sind, in dieser Seitengruppe gespeichert werden.

Mithilfe von Speicherklassen können Sie steuern, wo nicht gemeinsam genutzte Nachrichtendaten zu Zwecken der Administration, der Dateienspeicherplatz- und Lastverwaltung oder der Anwendungsisolation gespeichert werden. Sie können mit Speicherklassen auch die XCF-Gruppe und den Mitgliedsnamen einer IMS-Region definieren, sofern Sie eine IMS-Bridge verwenden (siehe dazu „IBM MQ und IMS” auf Seite 314).

Gemeinsam genutzte Warteschlangen verwenden keine Speicherklassen zum Abrufen einer Seitengruppenzuordnung, weil die Nachrichten in diesen Warteschlangen nicht in Seitengruppen gespeichert sind.

## Funktionsweise von Speicherklassen

- Zum Definieren einer Speicherklasse geben Sie im Befehl 'DEFINE STGCLASS' eine Seitengruppen-ID (PSID) an.
- Wenn Sie eine Warteschlange definieren, geben Sie die Speicherklasse im Attribut 'STGCLASS' an.

Im folgenden Beispiel wird die lokale Warteschlange 'QE5' mithilfe der Speichergruppe 'ARC2' der Seitengruppe 21 zugeordnet.

```
DEFINE STGCLASS(ARC2) PSID(21)
DEFINE QLOCAL(QE5) STGCLASS(ARC2)
```

Dies bedeutet, dass Nachrichten, die in die Warteschlange 'QE5' eingereiht werden, in der Seitengruppe 21 gespeichert werden (wenn sie lang genug in der Warteschlange bleiben, um in die DASD-Einheit (Direct Access Storage Device) geschrieben zu werden).

Eine Speicherklasse kann von mehr als einer Warteschlange verwendet werden, und Sie können so viele Speicherklassen definieren, wie Sie möchten. Sie können das vorherige Beispiel wie folgt um weitere Speicherklassen- und Warteschlangendefinitionen erweitern:

```
DEFINE STGCLASS(ARC1) PSID(05)
DEFINE STGCLASS(ARC2) PSID(21)
DEFINE STGCLASS(MAXI) PSID(05)
DEFINE QLOCAL(QE1) STGCLASS(ARC1) ...
DEFINE QLOCAL(QE2) STGCLASS(ARC1) ...
DEFINE QLOCAL(QE3) STGCLASS(MAXI) ...
DEFINE QLOCAL(QE4) STGCLASS(ARC2) ...
DEFINE QLOCAL(QE5) STGCLASS(ARC2) ...
```

In Abbildung 71 auf Seite 251 werden die Speicherklassen 'ARC1' und 'MAXI' der Seitengruppe 05 zugeordnet. Deshalb werden die Warteschlangen 'QE1', 'QE2' und 'QE3' ebenfalls der Seitengruppe 05 zugeordnet. Entsprechend ordnet die Speicherklasse 'ARC2' der Seitengruppe 21 die Warteschlangen 'QE4' und 'QE5' zu.

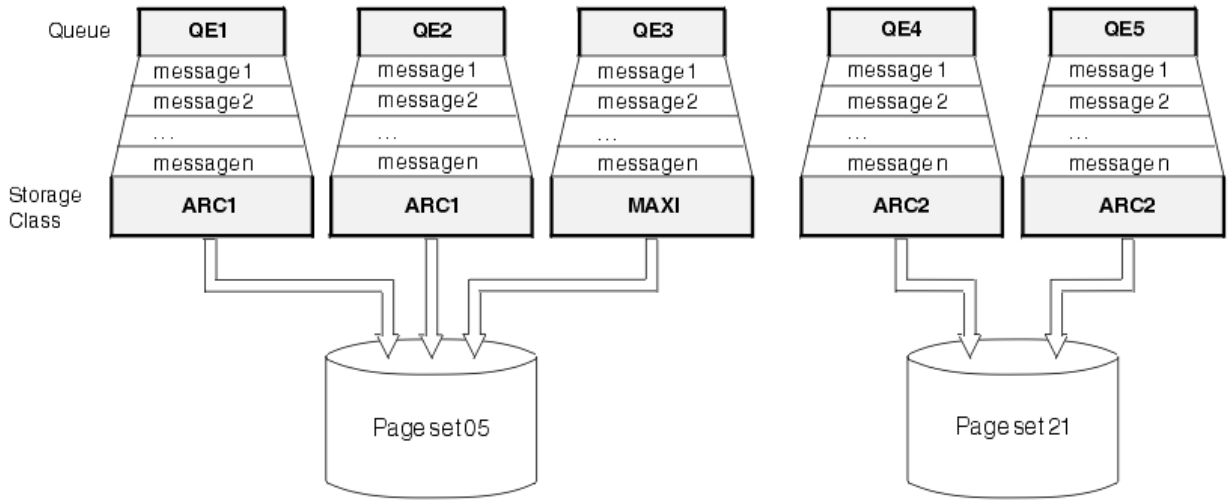



*Abbildung 71. Seitengruppen mithilfe von Speicherklassen Warteschlangen zuordnen*

Wenn Sie eine Warteschlange definieren, ohne eine Speicherklasse anzugeben, verwendet IBM MQ eine Standardspeicherklasse.

Wenn eine Nachricht in eine Warteschlange eingereiht wird, die eine nicht vorhandene Speicherklasse benennt, empfängt die Anwendung eine Fehlernachricht. In diesem Fall müssen Sie die Warteschlangendefinition ändern und den Namen einer vorhandenen Speicherklasse angeben oder eine Speicherklasse mit dem in der Warteschlange angegebenen Namen erstellen.

Sie können eine Speicherklasse nur in folgenden Fällen ändern:

- Alle Warteschlangen, die die zu ändernde Speicherklasse verwenden, sind leer und haben keinerlei nicht festgeschriebenen Aktivitäten.
- Alle Warteschlangen, die die zu ändernde Speicherklasse verwenden, sind geschlossen.

## z/OS Puffer und Pufferpools für IBM MQ for z/OS

In IBM MQ for z/OS werden Puffer und Pufferpools zur vorübergehenden Zwischenspeicherung von Daten verwendet. In diesem Abschnitt wird erläutert, wie Puffer organisiert und verwendet werden.

Zur Effizienzsteigerung wird in IBM MQ eine Form der Zwischenspeicherung verwendet, bei der Nachrichten (und Objektdefinitionen) vorübergehend in Puffern zwischengespeichert werden, bevor sie in Seitengruppen auf DASD-Einheiten (Direct Access Storage Device) gespeichert werden. Kurzlebige Nachrichten, d. h. Nachrichten, die kurz nach ihrem Empfang wieder aus einer Warteschlange abgerufen werden, werden möglicherweise ausschließlich in Puffern gespeichert. Diese Caching-Aktivität wird durch einen Puffermanager gesteuert, der eine Komponente von IBM MQ ist.

Die Puffer werden in *Pufferpools* organisiert. Sie können für jeden Warteschlangenmanager bis zu 100 Pufferpools (0 bis 99) definieren.

Es wird empfohlen, die minimale Anzahl von Pufferpools zu verwenden, die mit der in Abbildung 72 auf Seite 252 beschriebenen Trennung von Objekt- und Nachrichtentyp konsistent sind, und alle Anforderungen an die Datenisolation, die Ihre Anwendung möglicherweise aufweist. Jeder Puffer ist 4 KB lang. Pufferpools verwenden standardmäßig 31-Bit-Speicher. In diesem Modus wird die maximale Anzahl der Puffer durch den im Adressraum des Warteschlangenmanagers verfügbaren 31-Bit-Speicher bestimmt. Es sollten jedoch maximal 70 % des verfügbaren Speichers für Puffer verwendet werden. Alternativ kann die Pufferpoolspeicherzuordnung aus 64-Bit-Speicher erfolgen (verwenden Sie das Attribut LOCATION des Befehls **DEFINE BUFFPOOL** ). Die Zuweisung von 64-Bitspeicher durch LOCATION(ABOVE) hat zwei Vorteile. Erstens steht wesentlich mehr 64-Bitspeicher zur Verfügung, die Pufferpools können also größer sein, und zweitens wird der 31-Bitspeicher von anderen Funktionen zur Verfügung gestellt. Generell gilt

aber: Je mehr Puffer Ihnen zur Verfügung steht, desto effizienter ist die Pufferung und desto besser die Leistung von IBM MQ.

Abbildung 72 auf Seite 252 zeigt die Beziehung zwischen Nachrichten, Puffern, Pufferpools und Seitengruppen. Ein Pufferpool ist einer oder mehreren Seitengruppen zugeordnet, während eine Seitengruppe immer nur einem einzelnen Pufferpool zugeordnet sein darf.

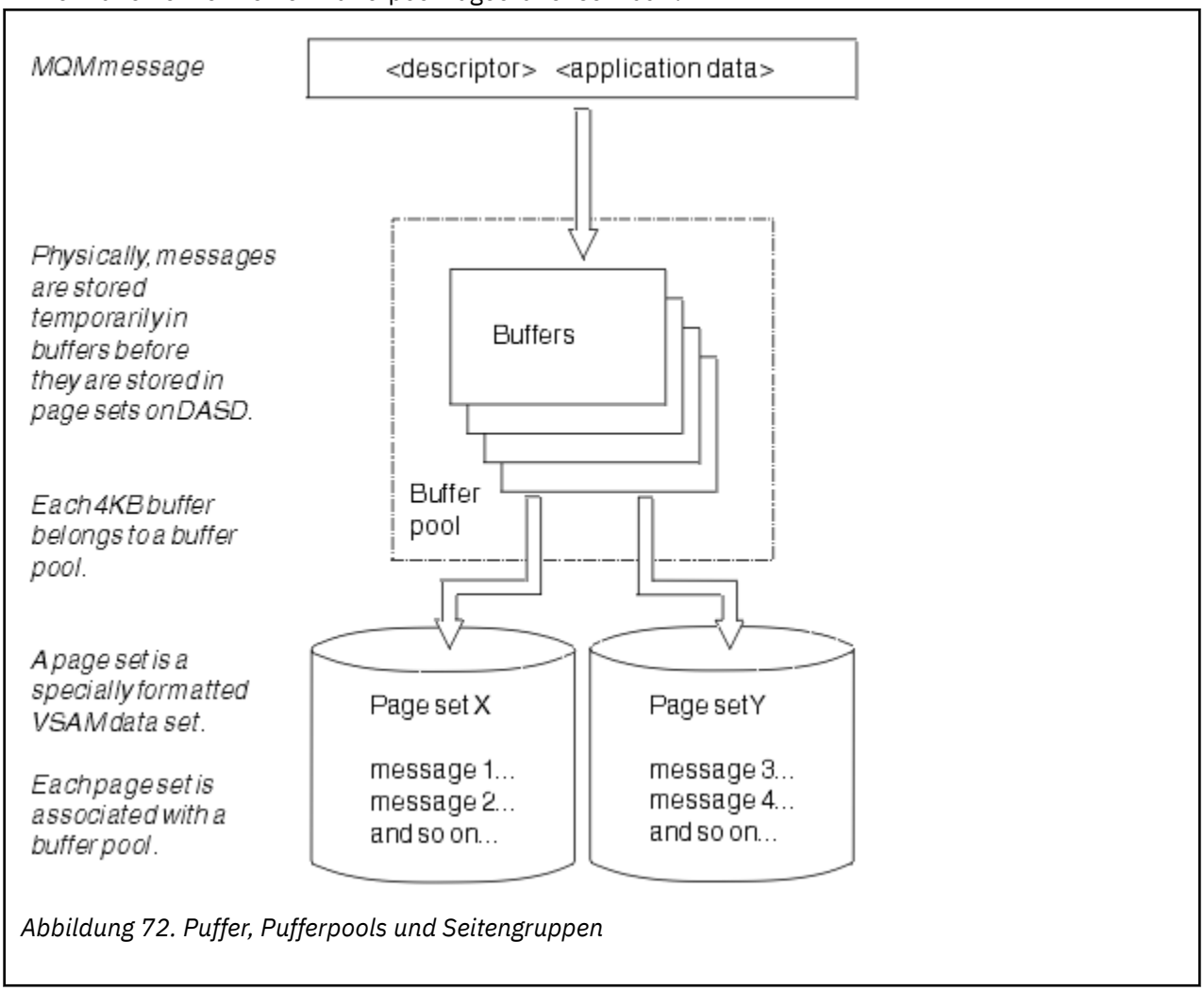



*Abbildung 72. Puffer, Pufferpools und Seitengruppen*

Sie können dynamisch Befehle ausgeben, um die Pufferpoolgröße und -position zu ändern. Dabei wird der Befehl **ALTER BUFFPOOL** verwendet. Mit dem Befehl **DEFINE PSID** können Seitengruppen dynamisch hinzugefügt werden, während sie mit dem Befehl **DELETE PSID** gelöscht werden können.

Wenn ein Pufferpool zu klein ist, gibt IBM MQ die Nachricht CSQP020E aus. Anschließend können Sie dem betroffenen Pufferpool weitere Puffer hinzufügen (dabei ist zu beachten, dass dazu möglicherweise Puffer aus anderen Pufferpools entfernt werden müssen).

Die Anzahl der Puffer in einem Pool wird mit dem Befehl **DEFINE BUFFPOOL** angegeben, während zur dynamischen Änderung der Größe eines Pufferpools der Befehl **ALTER BUFFPOOL** verwendet wird. Sie können die aktuelle Anzahl von Puffern in einem Pool dynamisch ermitteln, indem Sie mit dem Befehl **DISPLAY USAGE** eine Seitengruppe anzeigen, die den jeweiligen Pufferpool verwendet.

Um die Leistung nicht zu beeinträchtigen, sollten Nachrichten und Objektdefinitionen nicht in demselben Pufferpool gespeichert werden. Stattdessen sollte ein Pufferpool (z. B. Pufferpool null) ausschließlich für die Seitengruppe null verwendet werden, in der die Objektdefinitionen gespeichert sind. In ähnlicher Weise sollten kurzlebige und langlebige Nachrichten in unterschiedlichen Pufferpools und damit auch in unterschiedlichen Seitengruppen und unterschiedlichen Warteschlangen gespeichert werden.

Der Befehl **DEFINE BUFFPOOL** kann nach dem Neustart nicht zum Erstellen eines neuen Pufferpools verwendet werden. Stattdessen kann im Befehl **DEFINE PSID** das Schlüsselwort 'DSN' angegeben werden,

um explizit einen Pufferpool zu identifizieren, der derzeit noch nicht definiert ist. Dieser neue Pufferpool wird dann erstellt.

## z/OS Weitere Informationen zur Speicherverwaltung für IBM MQ for z/OS

In diesem Abschnitt finden Sie Verweise auf weitere Informationen zur Speicherverwaltung in IBM MQ for z/OS.

Weitere Informationen zu den Themen in diesem Abschnitt finden Sie in folgenden Quellen:

*Tabelle 21. Weitere Informationen zur Speicherverwaltung*

| **Thema** | **Quelle** |
|---|---|
| Benötigter Speicherplatz | Speicher- und Leistungsanforderungen unter z/OS planen |
| Größe von Seitengruppen und Pufferpools festlegen | Seitengruppen und Pufferpools planen |
| Seitengruppen verwalten | Verwaltung von Seitengruppen |
| MQSC-Befehle | WebSphere MQ-Scriptbefehle |

## z/OS Protokollierung in IBM MQ for z/OS

IBM MQ verwaltet *Protokolle*, in denen Datenänderungen und signifikante Ereignisse aufgezeichnet werden, sobald sie auftreten. Anhand dieser Protokolle können die Daten bei Bedarf auf einen vorherigen Stand zurückgesetzt werden.

Im *Bootstrap-Data-Set* (BSDS) werden Informationen zu den Dateien gespeichert, die diese Protokolle enthalten.

Die Protokolle enthalten keine Informationen für Statistiken, Traces oder Leistungsbewertungen. Weitere Informationen zu den Statistik- und Überwachungsinformationen, die IBM MQ erfasst, finden Sie im Abschnitt Überwachung und Statistik.

Weitere Informationen zur Protokollierung finden Sie in folgenden Abschnitten:

- „Protokolldateien in IBM MQ for z/OS” auf Seite 254
- „Aufbau des Protokolls” auf Seite 258
- „Verfahren beim Schreiben der IBM MQ for z/OS-Protokolle” auf Seite 259
- „Größeres Protokoll-Relative Byteadresse” auf Seite 262
- „Bootstrap-Dataset” auf Seite 263

**Zugehörige Tasks**

Protokollierungsumgebung planen

Protokolle mithilfe des Systemparametermoduls festlegen

z/OS z/OS verwalten

**Zugehörige Verweise**

z/OS Nachrichten für IBM MQ for z/OS

# z/OS Protokolldateien in IBM MQ for z/OS

Protokolldateien enthalten Informationen, die für die Wiederherstellung von Transaktionen erforderlich sind. Aktive Protokolldateien können archiviert werden, sodass die Protokolldaten über einen längeren Zeitraum aufbewahrt werden können.

## Was ist eine Protokolldatei?

IBM MQ zeichnet alle signifikanten Ereignisse bei ihrem Auftreten in einem *aktiven Protokoll* auf. Das Protokoll enthält die Informationen, die für eine Wiederherstellung erforderlich sind:

- Permanente Nachrichten
- IBM MQ-Objekte, zum Beispiel Warteschlangen
- IBM MQ-Warteschlangenmanager

Das aktive Protokoll umfasst eine Sammlung von Dateien (bis zu 310), die reihum abwechselnd verwendet werden.

Sie können die Protokollarchivierung aktivieren, damit in einer Archivdatei eine Kopie erstellt wird, wenn das aktive Protokoll voll wird. Mithilfe der Archivierung können Sie Protokolldaten über einen längeren Zeitraum aufbewahren. Wenn Sie die Protokollarchivierung nicht aktivieren, werden die Protokolldateien umlaufend verwendet und frühere Daten darin überschrieben. Zur Wiederherstellung einer Seitengruppe oder von Daten in einer Coupling-Facility-Struktur benötigen Sie die Protokolldaten von dem Zeitpunkt, zu dem die Sicherung der Seitengruppe oder Struktur vorgenommen wurde. Ein Archivprotokoll kann auf einer Festplatte oder einem Bandspeicher erstellt werden.

## Archivierung

Da das aktive Protokoll eine feste Größe hat, kopiert IBM MQ den Inhalt einer jeden Protokolldatei regelmäßig in ein *Archivprotokoll*, bei dem es sich normalerweise um eine Datei auf einer DASD-Einheit (Direct-Access Storage Device) oder auf einem Magnetband handelt. Wenn im Zusammenhang mit einem Subsystem oder einer Transaktion ein Fehler auftritt, verwendet IBM MQ das aktive Protokoll und bei Bedarf auch das Archivprotokoll für die Wiederherstellung.

Das Archivprotokoll kann bis zu 1000 sequenzielle Dateien enthalten. Jede Datei kann mit der integrierten Katalogfunktion (Integrated Catalog Facility, ICF) von z/OS katalogisiert werden.

Die Archivierung ist eine wesentliche Komponente der Wiederherstellung in IBM MQ. Wenn eine Arbeitseinheit mit Wiederherstellung eine lange Laufzeit hat, werden Protokolleinträge dieser Arbeitseinheit möglicherweise auch in das Archivprotokoll geschrieben. In diesem Fall werden zur Wiederherstellung auch Daten aus dem Archivprotokoll benötigt. Wenn die Archivierung jedoch inaktiviert ist, wird das aktive Protokoll umlaufend mit neuen Protokolleinträgen gefüllt, wobei frühere Protokolleinträge überschrieben werden. Dies bedeutet, dass IBM MQ die Arbeitseinheit mit Wiederherstellung möglicherweise nicht zurücksetzen kann und deshalb Nachrichten verloren gehen. In diesem Fall wird der Warteschlangenmanager abnormal beendet.

Deshalb sollten Sie in einer Produktionsumgebung die **Archivierung niemals inaktivieren**. Falls Sie dies dennoch tun, besteht nach einem System- oder Transaktionsfehler die Gefahr von Datenverlust. Deshalb sollte die Archivierung ausschließlich und allenfalls in einer Testumgebung inaktiviert werden. Falls die Inaktivierung erforderlich ist, verwenden Sie hierfür das Makro 'CSQ6LOGP', das unter CSQ6LOGP verwenden beschrieben ist.

Um Probleme mit ungeplanten lange laufenden Arbeitseinheiten mit Wiederherstellung zu vermeiden, gibt IBM MQ eine Nachricht aus (CSQJ160I oder CSQJ161I), wenn bei der Auslagerung des aktiven Protokolls eine solche Arbeitseinheit erkannt wird.

## Doppelte Protokollierung

Bei der doppelten Protokollierung wird jeder Protokolleintrag in zwei verschiedene aktive Protokolldateien geschrieben, um die Wahrscheinlichkeit eines Datenverlusts beim Neustart zu minimieren.

Sie können für IBM MQ die *einfache Protokollierung* oder *doppelte Protokollierung* konfigurieren. Bei der einfachen Protokollierung werden die Protokolleinträge nur in eine aktive Protokolldatei geschrieben. Jede aktive Protokolldatei ist eine lineare VSAM-Datei (Virtual Storage Access Method) mit einem einzelnen Speicherbereich. Bei der doppelten Protokollierung wird jeder Protokolleintrag in zwei verschiedene Datensätze in der aktiven Protokolldatei geschrieben. Auf diese Weise wird die Wahrscheinlichkeit von Datenverlust beim Neustart minimiert.

## Protokollverzögerung

Bei einer Protokollverzögerung (Shunting) werden die Protokolleinträge für einige Arbeitseinheiten weiter unten im Protokoll gespeichert. Dadurch wird die Menge der Protokolldaten verringert, die bei einem Neustart oder einer Auslagerung des Warteschlangenmanagers, für lange laufende oder über lange Zeit unbestätigte Arbeitseinheiten gelesen werden muss.

Wenn eine Arbeitseinheit als lang angesehen wird, wird eine Darstellung jedes Protokollsatzes weiter unten in das Protokoll geschrieben. Dieses Verfahren wird als *Verzögerung* (Shunting) bezeichnet. Nachdem die gesamte Arbeitseinheit verarbeitet wurde, befindet sie sich in einem *verzögerten* Zustand. Bei einer Auslagerungs- oder Neustartaktivität, die in Bezug auf die verzögerte Arbeitseinheit ausgeführt wird, können die verzögerten Protokolleinträge anstelle der ursprünglichen Protokolleinträge für die Arbeitseinheit verwendet werden.

Die Erkennung einer lange laufenden Arbeitseinheit ist eine Funktion der Prüfpunktverarbeitung. Zum Zeitpunkt des Prüfpunkts wird jede aktive Arbeitseinheit daraufhin überprüft, ob sie verzögert werden muss oder nicht. Wenn seit dem Erstellen der Arbeitseinheit oder seit ihrer letzten Verzögerung bereits zwei vorherige Prüfpunkte vergangen sind, wird diese Arbeitseinheit verzögert. Dies bedeutet, dass eine Arbeitseinheit auch mehr als einmal verzögert werden kann. Sie wird in diesem Fall als *mehrfach verzögerte* Arbeitseinheit bezeichnet.

Eine Arbeitseinheit wird alle drei Prüfpunkte verzögert. Allerdings wird der Prüfpunkt asynchron zum Protokollwechsel (oder dem Schreiben des Protokolleintrags, wodurch LOGLOAD überschritten wurde) ausgeführt.

Es findet immer nur ein Prüfpunkt statt, deshalb können möglicherweise mehrere Protokollwechsel stattfinden, bevor ein Prüfpunkt abgeschlossen ist.

Wenn also nicht ausreichend aktive Protokolle vorhanden sind oder falls diese zu klein sind, wird die Verzögerung einer umfangreichen Arbeitseinheit möglicherweise nicht abgeschlossen, bevor alle Protokolle voll sind.

Wenn die Verzögerung nicht beendet werden kann, wird die Nachricht CSQR027I ausgegeben.

Falls die Protokollarchivierung inaktiviert ist, tritt ABEND 5C6 mit Ursachencode 00D1032A auf, wenn versucht wurde, die Arbeitseinheit zurückzusetzen, für die die Verzögerung fehlgeschlagen ist. Verwenden Sie OFFLOAD=YES, um dieses Problem zu vermeiden.

Die Protokollverzögerung (Shunting) ist immer aktiviert und wird unabhängig davon ausgeführt, ob die Protokollarchivierung aktiviert ist oder nicht.

**Anmerkung:** Obwohl alle Protokolleinträge für eine Arbeitseinheit verzögert werden, wird nicht der gesamte Inhalt eines jeden Eintrags verzögert, sondern nur der Teil, der für die Auslagerung erforderlich ist. Dies bedeutet, dass die Menge der zu speichernden Daten zu klein wie möglich gehalten wird und dass verzögerte Protokolleinträge im Falle eines Seitengruppenfehlers nicht verwendet werden können. Die Laufzeit einer Arbeitseinheit wird als lang eingestuft, wenn die Arbeitseinheit seit mehr als drei Warteschlangenmanager-Prüfpunkten aktiv ist.

Weitere Informationen zur Protokollverzögerung finden Sie im Abschnitt Protokolle verwalten.

## Protokollkomprimierung

Sie können in der Konfiguration von IBM MQ for z/OS angeben, dass Protokolleinträge, wenn sie in die Protokolldateien geschrieben bzw. daraus gelesen werden, komprimiert bzw. dekomprimiert werden sollen.

Mithilfe der Protokollkomprimierung kann die Datenmenge, die in das Protokoll für permanente Nachrichten in privaten Warteschlangen geschrieben werden muss, verringert werden. Der Komprimierungsgrad, der erreicht werden kann, hängt vom Datentyp ab, der in den Nachrichten enthalten ist. Beispielsweise kann die Lauflängencodierung, die auf der Komprimierung wiederholter Instanzen von Bytes basiert, gute Ergebnisse bei strukturierten oder datensatzorientierten Daten liefern.

**Achtung:** Persistente Nachrichten, die in eine gemeinsam genutzte Warteschlange eingereiht werden, unterliegen nicht der Protokollkomprimierung.

Mithilfe von Feldern im Protokollmanagerabschnitt der Datensätze von System Management Facility 115 (SMF) können Sie den Umfang der erreichten Datenkomprimierung überwachen. Weitere Informationen zu SMF finden Sie unter Systemmanagementfunktion verwenden und Abrechnungs- und Statistikdaten.

Durch die Protokollkomprimierung wird die Prozessorauslastung des Systems erhöht. Die Protokollkomprimierung ist nur dann eine sinnvolle Maßnahme, wenn der Datendurchsatz des Warteschlangenmanagers durch die E/A-Bandbreite bei Schreibvorgängen in den Protokolldateien begrenzt ist oder Einschränkungen bezüglich des Plattenspeicherplatzes bestehen, der zum Speichern der Protokolldateien benötigt wird. Wenn Sie gemeinsam genutzte Warteschlangen verwenden, können Sie Einschränkungen der E/A-Bandbreite mindern, indem Sie der Gruppe mit gemeinsamer Warteschlange weitere Warteschlangenmanager hinzufügen und die Arbeitslast auf mehrere Warteschlangenmanager verteilen.

Die Option für Protokollkomprimierung kann nach Bedarf aktiviert und inaktiviert werden, ohne dass der Warteschlangenmanager gestoppt und neu gestartet werden muss. Der Warteschlangenmanager kann eventuell vorhandene komprimierte Protokolleinträge unabhängig von der aktuellen Protokollkomprimierungseinstellung lesen.

Der Warteschlangenmanager unterstützt 3 Einstellungen für die Protokollkomprimierung.

**KEINE**
Die Protokolldatenkomprimierung ist inaktiviert. Dies ist der Standardwert.

**RLE**
Die Protokolldatenkomprimierung wird durch Lauflängencodierung (Run-Length Encoding, RLE) ausgeführt.

**ANY**
Der Warteschlangenmanager erhält die Möglichkeit, den Komprimierungsalgorithmus auszuwählen, mit dem der höchste Komprimierungsgrad für Protokolleinträge erreicht wird. Diese Option führt zu einer RLE-Komprimierung.

Sie können die Komprimierung von Protokolleinträgen auf folgende Weisen steuern:

- Mit den Befehlen SET und DISPLAY LOG in MQSC; siehe SET LOG und DISPLAY LOG
- Mit den Funktionen 'Set Log' und 'Inquire Log' in der PCF-Schnittstelle; siehe Protokoll festlegen und Protokoll abfragen
- Mit dem Makro CSQ6LOGP im Systemparametermodul; siehe CSQ6LOGP verwenden

Darüber hinaus unterstützt das Druckdienstprogramm für Protokolle, 'CSQ1LOGP', das Aufheben der Komprimierung von komprimierten Protokolleinträgen.

## Protokolldaten

Das Protokoll kann bis zu 18 Trillionen ($1{,}8*10^{19}$) Byte enthalten. Jedes Byte kann durch seinen Abstand zum Beginn des Protokolls adressiert werden. Diese Art der Adressierung wird als *relative Byteadresse* (Relative Byte Address, RBA) bezeichnet.

Die relative Byteadresse wird mithilfe eines 6-Byte- oder 8-Byte-Feldes angegeben, sodass der adressierbare Gesamtbereich $2^{48}$ bzw. $2^{64}$ Byte beträgt, je nachdem, ob Protokoll-RBAs mit sechs oder mit acht Byte verwendet werden.

Wenn IBM MQ jedoch feststellt, dass der belegte Bereich über F00000000000 (bei Verwendung von Protokoll-RBAs mit sechs Byte) oder FFFF800000000000 (bei Verwendung von Protokoll-RBAs mit acht Byte) hinaus geht, werden Sie mit den Nachrichten CSQI045, CSQI046, CSQI047 und CSQJ032 gewarnt und dazu aufgefordert, die relative Byteadresse des Protokolls zurückzusetzen.

Wenn der Wert der relativen Byteadresse FFF800000000 (bei Verwendung von Protokoll-RBAs mit sechs Byte) oder FFFFFFC000000000 (bei Verwendung von Protokoll-RBAs mit acht Byte) erreicht, wird der Warteschlangenmanager mit dem Ursachencode 00D10257 beendet.

Nachdem die Warnungen bezüglich des belegten Protokollbereichs ausgegeben wurden, sollten Sie eine Betriebsunterbrechung des Warteschlangenmanagers planen, in der der Warteschlangenmanager für die Verwendung von Protokoll-RBAs konvertiert oder das Protokoll zurückgesetzt werden kann. Das Verfahren für die Zurücksetzung des Protokolls ist im Abschnitt Protokoll des Warteschlangenmanagers zurücksetzen dokumentiert.

Wenn Ihr Warteschlangenmanager relative Byteadressen mit sechs Byte verwendet, sollten Sie den Warteschlangenmanager für die Verwendung von Protokoll-RBAs mit acht Byte konvertieren, statt das Protokoll des Warteschlangenmanagers zurückzusetzen. Eine entsprechende Beschreibung dieses Verfahrens finden Sie im Abschnitt Größere relative Byteadresse für das Protokoll implementieren.

Das Protokoll besteht aus *Protokolleinträgen*, die jeweils einen Satz an Protokolldaten darstellen, die wie eine individuelle Einheit behandelt werden. Ein Protokolleintrag wird entweder durch die relative Byteadresse des ersten Byte seines Headers identifiziert oder durch seine Protokollsatzfolgenummer (Log Record Sequence Number, LRSN). Mit einer relativen Byteadresse oder Protokollsatzfolgenummer wird ein bestimmter Eintrag angegeben, der an einem bestimmten Punkt im Protokoll beginnt.

Ob zur Bezeichnung des Protokollpunkts die relative Byteadresse oder Protokollsatzfolgenummer erforderlich ist, hängt davon ab, ob Sie Gruppen mit gemeinsamer Warteschlange verwenden oder nicht. In einer Umgebung mit gemeinsam genutzten Warteschlangen kann die relative Byteadresse nicht zur eindeutigen Identifizierung eines Protokollpunkts verwendet werden, weil dieselbe Warteschlange gleichzeitig von mehreren Warteschlangenmanagern aktualisiert werden kann und jeder Warteschlangenmanager sein eigenes Protokoll hat. Um dieses Problem zu lösen, wird die Protokollsatzfolgenummer aus einem Zeitmarkenwert abgeleitet, die nicht unbedingt die physische Verschiebung des Protokolleintrags innerhalb des Protokolls darstellt.

Jeder Protokolleintrag hat einen Header, in dem dessen Typ, die IBM MQ-Unterkomponente, die den Eintrag erstellt hat, und die ID der Arbeitseinheit mit Wiederherstellung (sofern es sich um einen entsprechenden Protokolleintrag handelt) angegeben sind.

Es gibt vier Protokolleintragstypen, die in den folgenden Abschnitten beschrieben sind:

- Protokolleinträge für Arbeitseinheiten mit Wiederherstellung
- Prüfpunktsätze
- Seitengruppensteuersätze
- CF-Struktursicherungssätze

## Protokolleinträge für Arbeitseinheiten mit Wiederherstellung

Die meisten Protokolleinträge beschreiben Änderungen an IBM MQ-Warteschlangen. All solche Änderungen werden mithilfe von Arbeitseinheiten mit Wiederherstellung ausgeführt.

IBM MQ setzt spezielle Protokollierungsverfahren ein, bei denen Protokolleinträge *rückgängig gemacht/wiederholt* und *kompensiert* werden, um die Neustartdauer zu reduzieren und die Systemverfügbarkeit zu erhöhen.

Dies führt unter anderem dazu, dass die Neustartdauer gebunden ist. Wenn während des Neustarts ein Fehler auftritt, sodass der Warteschlangenmanager ein zweites Mal neu gestartet werden muss, muss keine der Wiederherstellungsaktivitäten, die bis zum Fehler beim ersten Neustart abgeschlossen wurden, beim zweiten Neustart erneut angewendet werden. Dies bedeutet, dass nachfolgende Neustarts nicht immer länger dauern, bis sie abgeschlossen sind.

### Prüfpunktsätze

Um die Neustartdauer zu verringern, setzt IBM MQ während des normalen Betriebs regelmäßig Prüfpunkte. Diese Prüfpunkte treten zu folgenden Zeitpunkten auf:

- Wenn eine vordefinierte Anzahl von Protokolleinträgen geschrieben wurde. Diese Anzahl wird mit dem Prüfpunktfrequenzoperanden 'LOGLOAD' des Systemparametermakros 'CSQ6SYSP' definiert, das unter 'CSQ6SYSP' verwenden beschrieben ist.
- Am Ende eines erfolgreichen Neustarts.
- Bei normaler Beendigung.
- Immer, wenn IBM MQ zur nächsten aktiven Protokolldatei in der Reihenfolge wechselt.

Wenn ein Prüfpunkt gesetzt wird, gibt IBM MQ intern den Befehl DISPLAY CONN aus (dieser wird im Abschnitt DISPLAY CONN beschrieben), damit eine Liste der Verbindungen mit aktuell unbestätigtem Status in das z/OS-Konsolenprotokoll geschrieben wird.

### Seitengruppensteuersätze

In diesen Einträgen werden die dem IBM MQ-Warteschlangenmanager bekannten Seitengruppen und Pufferpools am jeweiligen Prüfpunkt aufgezeichnet sowie Informationen zu den Protokollbereichen, die zum Ausführen der Datenträgerwiederherstellung der Seitengruppe zum Zeitpunkt des Prüfpunkts erforderlich sind.

Bestimmte dynamische Änderungen an Seitengruppen und Pufferpools werden ebenfalls als Seitengruppensteuersätze erstellt, damit die Änderungen zurückgeschrieben und beim nächsten Neustart des Warteschlangenmanagers automatisch wiederhergestellt werden können.

### CF-Struktursicherungssätze

Diese Protokolleinträge enthalten Daten, die als Reaktion auf den Befehl BACKUP CFSTRUCT aus einer Coupling-Facility-Listenstruktur gelesen werden. Im unwahrscheinlichen Fall eines Fehlers in Zusammenhang mit der Coupling-Facility-Struktur werden diese Protokolleinträge zusammen mit den Protokolleinträgen für eine Arbeitseinheit mit Wiederherstellung durch den Befehl RECOVER CFSTRUCT dazu verwendet, die Datenträgerwiederherstellung der Coupling-Facility-Struktur bis zum Fehlerpunkt durchzuführen.

**Zugehörige Tasks**

Relative Byteadresse für größere Protokolle implementieren

## z/OS Aufbau des Protokolls

In diesem Abschnitt wird die zur Beschreibung von Protokolleinträgen verwendet Terminologie erläutert.

Jede aktive Protokolldatei muss eine lineare VSAM-Datei (Virtual Storage Access Method) sein. Die physische Ausgabeeinheit, die in die aktive Protokolldatei geschrieben wird, ist ein 4 KB großes Steuerintervall. Jedes Steuerintervall enthält einen VSAM-Datensatz.

### Physische und logische Protokolleinträge

Ein VSAM-Steuerintervall entspricht einem *physischen* Datensatz- Die Informationen, die zu einem bestimmten Zeitpunkt protokolliert werden, bilden einen *logischen* Datensatz mit einer Länge, die unabhängig von dem im Steuerintervall verfügbaren Speicherplatz variiert. Ein physischer Datensatz könnte also Folgendes enthalten:

- mehrere logische Datensätze
- einen oder mehr logische Datensätze und einen Teil eines anderen logischen Datensatzes
- nur einen Teil eines logischen Datensatzes

Der Begriff *Protokolleintrag* bezieht sich auf einen *logischen* Datensatz, unabhängig davon, wie viele *physische* Datensätze zu seiner Speicherung erforderlich sind.

## z/OS Verfahren beim Schreiben der IBM MQ for z/OS-Protokolle

In diesem Abschnitt wird erläutert, wie Protokolldateieinträge von IBM MQ verarbeitet werden.

IBM MQ schreibt alle Protokolleinträge in eine DASD-Datei (Direct Access Storage Device), die als *aktives Protokoll* bezeichnet wird. Wenn das aktive Protokoll voll ist, kopiert IBM MQ dessen Inhalt in eine DASD-Einheit oder eine Banddatei, die als *Archivprotokoll* bezeichnet wird. Dieser Vorgang heißt *Auslagerung*.

Abbildung 73 auf Seite 260 zeigt den Protokollierungsprozess. Protokolleinträge durchlaufen normalerweise den folgenden Zyklus:

1. IBM MQ stellt Änderungen an Daten sowie signifikante Ereignisse in Wiederherstellungsprotokolleinträgen fest.
2. IBM MQ verarbeitet die Wiederherstellungsprotokolleinträge und teilt sie bei Bedarf in Segmente auf.
3. Die Protokolleinträge werden nacheinander in *Ausgabeprotokollpuffern* gespeichert, die als VSAM-Steuerintervalle (Virtual Storage Access Method) formatiert sind. Jeder Protokolleintrag wird durch eine relative Byteadresse im Bereich zwischen 0 und $2^{64}$-1 identifiziert.
4. Die Steuerintervalle werden in einen Satz vordefinierter aktiver DASD-Protokolldateien geschrieben, die nacheinander zur Speicherung genutzt und in zyklischen Abständen wiederverwendet (d. h. überschrieben) werden.
5. Wenn eine aktive Protokolldatei bei aktivierter Archivierungsfunktion voll wird, wird deren Inhalt automatisch in eine neue Archivprotokolldatei ausgelagert.

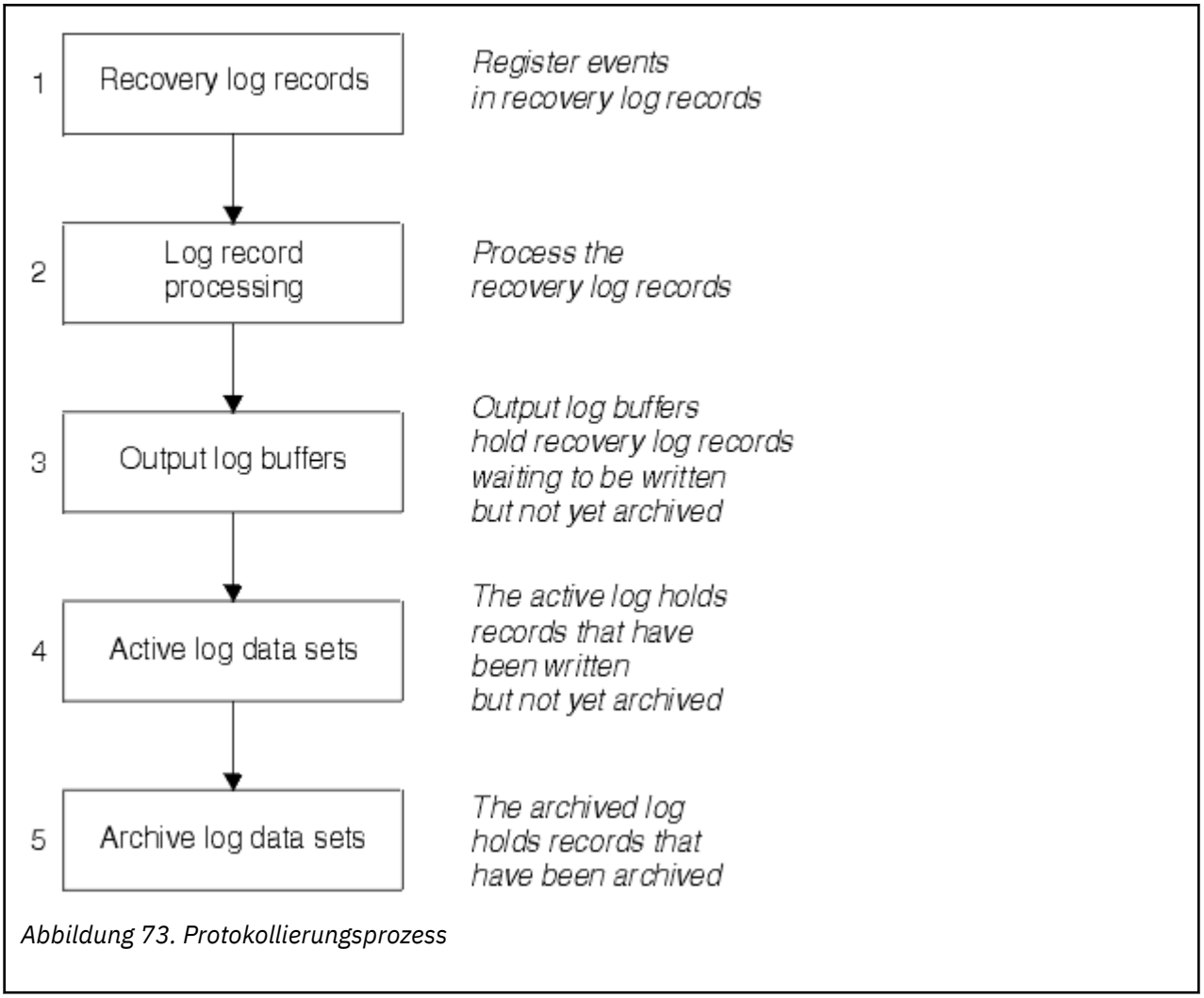



*Abbildung 73. Protokollierungsprozess*

## Zeitpunkt zum Schreiben des aktiven Protokolls

Die speicherinternen Protokollpuffer werden immer dann in eine aktive Protokolldatei geschrieben, wenn eines der folgenden Ereignisse eintritt:

- Die Protokollpuffer werden voll.
- Der Schwellenwert für Schreibvorgänge wird erreicht (wie im Makro 'CSQ6LOGP' angegeben).
- Bestimmte signifikante Ereignisse treten ein, z. B. das Erreichen eines Festschreibungspunkts oder das Ausführen des IBM MQ-Befehls BACKUP CFSTRUCT.

Wenn der Warteschlangenmanager initialisiert wird, werden die im Bootstrap-Data-Set (BSDS) genannten aktiven Protokolldateien dynamisch zur exklusiven Verwendung durch den Warteschlangenmanager zugeordnet und bleiben IBM MQ exklusiv zugeordnet, bis der Warteschlangenmanager beendet wird.

## Protokolldateien dynamisch hinzufügen

Neue aktive Protokolldateien können dynamisch definiert werden, während der Warteschlangenmanager aktiv ist. Durch diese Funktion wird das Problem eines blockierten Warteschlangenmanagers verringert, wenn die Archivierungsfunktion aktive Protokolle wegen eines vorübergehenden Fehlers nicht auslagern kann. Weitere Informationen hierzu finden Sie unter der Beschreibung des Befehls DEFINE LOG.

**Anmerkung:** Wenn Sie aktive Protokolle neu definieren oder entfernen möchten, müssen Sie den Warteschlangenmanager beenden und erneut starten.

## IBM MQ und Storage Management Subsystem

Mit IBM MQ -Parametern können Sie Speicherklassen des Speicherverwaltungssubsystems ( MVS/DFP SMS) angeben, wenn IBM MQ -Archivprotokolldateien dynamisch zugeordnet werden. IBM MQ leitet zwar die Archivierung von Protokolldateien ein, doch mit SMS können Sie die Zuordnung der Archivdateien vornehmen.

**Zugehörige Verweise**

„Schreiben des IBM MQ for z/OS-Archivprotokolls” auf Seite 261
In diesem Abschnitt wird der Prozess zum Kopieren aktiver Protokolle in Archivprotokolle sowie der Zeitpunkt für diesen Prozess erläutert.

### z/OS *Schreiben des IBM MQ for z/OS-Archivprotokolls*

In diesem Abschnitt wird der Prozess zum Kopieren aktiver Protokolle in Archivprotokolle sowie der Zeitpunkt für diesen Prozess erläutert.

Der Prozess zum Kopieren aktiver Protokolle in Archivprotokolle wird als *Auslagerung* bezeichnet. Die Beziehung der Auslagerung zu anderen Protokollierungsereignissen ist schematisch in Abbildung 74 auf Seite 261 dargestellt.

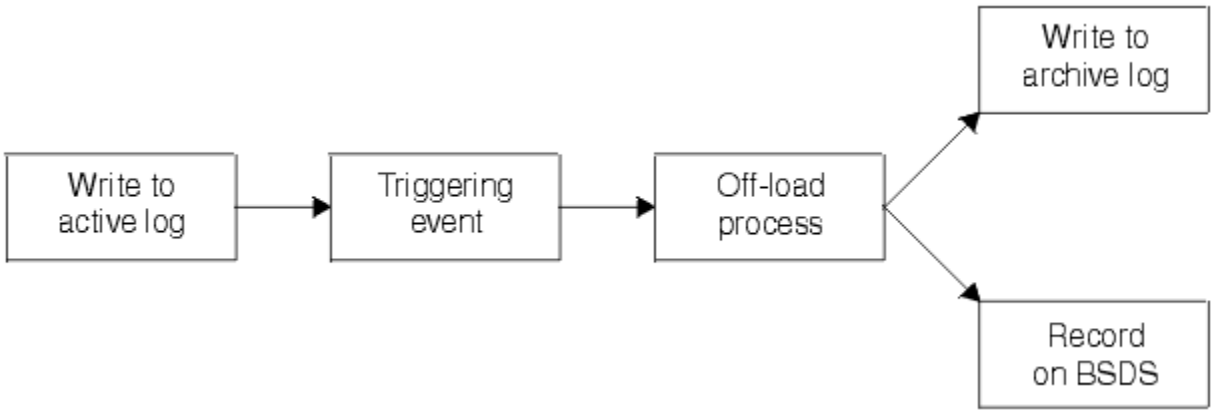



*Abbildung 74. Auslagerungsprozess*

### Auslagerungsprozess auslösen

Der Prozess zur Auslagerung eines aktiven Protokolls in ein Archivprotokoll kann durch verschiedene Ereignisse ausgelöst werden. For example:

- Die aktive Protokolldatei wird voll.
- Der Befehl 'MQSC ARCHIVE' wird ausgeführt.
- Bei einem Schreibvorgang in einer aktiven Protokolldatei tritt ein Fehler auf.

Die Datei wird vor dem Fehlerpunkt abgeschnitten, und der Datensatz, der dadurch nicht geschrieben wurde, wird zum ersten Datensatz in der neuen Datei. Die Auslagerung wird für die abgeschnittene Datei auf die gleiche Weise ausgelöst wie für eine normale, volle Protokolldatei. Wenn zwei aktive Protokolle vorhanden sind, werden beide Kopien abgeschnitten, damit die beiden Kopien synchronisiert bleiben.

Die Nachricht CSQJ110E wird ausgegeben, wenn das letzte verfügbare aktive Protokoll zu 5 % voll ist, und danach beim Erreichen eines jeden 5 %-Schritts, und gibt die aktuell belegte Kapazität des Protokolls als

Prozentsatz an. Wenn alle aktiven Protokolle voll werden, unterbricht IBM MQ die Verarbeitung, bis die Auslagerung einsetzt, und gibt folgende Nachricht aus:

```
CSQJ111A +CSQ1 OUT OF SPACE IN ACTIVE LOG DATA SETS
```

## Auslagerungsprozess

Wenn alle aktiven Protokolle voll werden, führt IBM MQ den Auslagerungsprozess durch und unterbricht bis zu dessen Abschluss die Verarbeitung. Wenn der Auslagerungsprozess fehlschlägt und die aktiven Protokolle voll sind, wird IBM MQ abnormal beendet.

Wenn ein aktives Protokoll für die Auslagerung bereit ist, wird eine Anforderung mit der Aufforderung zum Einlegen eines Bands oder zum Vorbereiten einer DASD-Einheit (Direct Access Storage Device) an den z/OS-Konsolenbediener gesendet. Der Wert der Protokollierungsoption 'ARCWTOR' bestimmt, ob die Anforderung empfangen wird (weitere Informationen hierzu finden Sie unter CSQ6ARVP verwenden). Wenn Sie ein Band für die Auslagerung verwenden, geben Sie 'ARCWTOR=YES' an. Wenn der Wert 'YES' ist, geht der Anforderung eine WTOR-Nachricht (Write to Operator with Reply), Nachrichtennummer CSQJ008E, voraus, die den Bediener anweist, eine Archivprotokolldatei vorzubereiten, die zugeordnet werden soll.

Der Bediener muss auf diese Nachricht nicht sofort reagieren. Durch eine Verzögerung der Antwort wird jedoch auch der Auslagerungsprozess verzögert. Dies hat zunächst jedoch keine Auswirkungen auf die Leistung von IBM MQ, es sei denn, der Bediener verzögert die Antwort so lange, bis IBM MQ keine aktiven Protokolle mehr zur Verfügung stehen.

Der Bediener kann auch mit dem Abbruch des Auslagerungsprozesses reagieren. Wenn die Zuordnung in diesem Fall für die erste Kopie der zwei Archivdateien vorgesehen ist, wird der Auslagerungsprozess nur so lange verzögert, bis die nächste aktive Protokolldatei voll wird. Wenn die Zuordnung hingegen für die zweite Kopie vorgesehen ist, wechselt der Archivierungsprozess zum Einzelkopiemodus, jedoch nur für diese Datei.

## Unterbrechungen und Fehler während der Auslagerung

Eine Anforderung zum Stoppen des Warteschlangenmanager wird erst nach Abschluss des Auslagerungsprozesses ausgeführt. Wenn IBM MQ während des Auslagerungsprozesses fehlschlägt, beginnt die Auslagerung erneut, sobald der Warteschlangenmanager neu gestartet wird.

## Nachrichten während des Auslagerungsprozesses

Ausgelagerte Nachrichten werden von IBM MQ und dem Auslagerungsprozess an die z/OS -Konsole gesendet. Anhand dieser Nachrichten können Sie die Bereiche der relativen Byteadressen in den verschiedenen Protokolldateien ermitteln.

## z/OS Größeres Protokoll-Relative Byteadresse

Diese Funktion verbessert die Verfügbarkeit des Warteschlangenmanager, da sich der Zeitraum bis zur erforderlichen Zurücksetzung des Protokolls verlängert.

Wiederherstellungsdaten werden in das Protokoll geschrieben, damit persistente Nachrichten bei einem Neustart des Warteschlangenmanagers verfügbar sind. Mit dem Begriff 'Relative Byteadresse des Protokolls' (Protokoll-RBA) wird die Position der Daten als relative Position ab Beginn des Protokolls bezeichnet.

Vor IBM MQ 8.0 konnte die 6-Byte-Protokoll-RBA bis zu 256 Terabyte an Daten adressieren. Bevor diese Menge an geschriebenen Protokolleinträgen erreicht wird, müssen Sie das Protokoll des Warteschlangenmanagers mit dem Verfahren zurücksetzen, das im Abschnitt Protokoll des Warteschlangenmanagers zurücksetzen dokumentiert ist.

Die Zurücksetzung der Protokolle von Warteschlangenmanagern dauert lange und kann eine lange Betriebsunterbrechung bedeuten. Dies ist darauf zurückzuführen, dass die Seitengruppen im Rahmen des

Prozesses zurückgesetzt werden müssen. Bei einem Warteschlangenmanager mit hoher Auslastung wird diese Operation in der Praxis häufig einmal jährlich ausgeführt.

Ab IBM MQ 8.0 kann die Protokoll-RBA eine Länge von acht Byte haben. Dies bedeutet, dass der Warteschlangenmanager nun 64.000 Mal mehr Daten (16 Exabyte) adressieren kann, bevor die Protokoll-RBA zurückgesetzt werden muss. Die Verwendung einer höheren Protokoll-RBA wirkt sich dahingehend aus, dass sich die Größe der geschriebenen Protokolldaten um einige Byte erhöht.

### Wann wird diese Funktion aktiviert?

V 9.3.0 Für Warteschlangenmanager, die mit IBM MQ 9.3.0 oder höher erstellt wurden, ist diese Funktion bereits aktiviert.

Sobald sich die aktuelle Protokoll-RBA dem Ende des Protokoll-RBA-Bereichs nähert, sollten Sie den Warteschlangenmanager für die Verwendung einer Protokoll-RBA mit acht Byte konvertieren, statt das Protokoll des Warteschlangenmanagers zurückzusetzen. Bei der Konvertierung eines Warteschlangenmanagers für die Verwendung von Protokoll-RBAs mit acht Byte ist die Betriebsunterbrechung kürzer als bei der Zurücksetzung des Protokolls. Außerdem erhöht sich der Zeitraum bis zu einer erforderlichen Zurücksetzung des Protokolls erheblich.

Die Nachricht CSQJ034I, die während der Initialisierung des Warteschlangenmanagers ausgegeben wird, zeigt das Ende des konfigurierten Protokoll-RBA-Bereichs für den Warteschlangenmanager an und gibt Auskunft darüber, ob Protokoll-RBAs mit sechs oder acht Byte verwendet werden.

### Wie wird diese Funktion aktiviert?

Eine Protokoll-RBA mit acht Byte wird aktiviert, indem der Warteschlangenmanager mit Bootstrap-Datasets im Versionsformat 2 gestartet wird. Zusammenfassend wird dies durch folgende Aktionen erreicht:

1. Sicherstellung, dass alle Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange die Voraussetzungen für die Aktivierung einer Protokoll-RBA mit acht Byte erfüllen
2. Saubere Beendigung des Warteschlangenmanagers
3. Ausführung des Konvertierungsdienstprogramms für Bootstrap-Datasets zur Erstellung einer Kopie des Bootstrap-Datasets im Versionsformat 2
4. Neustart des Warteschlangenmanagers mit dem konvertierten Bootstrap-Dataset

Sobald ein Warteschlangenmanager für die Verwendung von Protokoll-RBA-Werten mit acht Byte konvertiert wurde, kann er nicht mehr auf die Verwendung von Protokoll-RBAs mit sechs Byte zurückgesetzt werden.

Im Abschnitt Größere relative Byteadresse für Protokoll implementieren wird die Vorgehensweise für die Aktivierung von Protokoll-RBAs mit acht Byte ausführlich beschrieben.

**Zugehörige Tasks**

Planung für die Erhöhung des maximal adressierbaren Protokollbereichs

**Zugehörige Verweise**

Das BSDS-Konvertierungsdienstprogramm (CSQJUCNV)

## z/OS Bootstrap-Dataset

Das Bootstrap-Dataset ist in IBM MQ als Mechanismus zum Verweisen auf Protokolldateien und Protokolleinträge erforderlich. Diese Informationen werden während des normalen Betriebs und der Wiederherstellung bei Neustarts benötigt.

### Zweck des Bootstrap-Datasets

Das *Bootstrap-Dataset* (BSDS) ist eine schlüsseladressierte VSAM-Datei, die von IBM MQ benötigte Informationen enthält. Sie enthält Folgendes:

- Eine Bestandsliste aller aktiven und archivierten Protokolldateien, die IBM MQ bekannt sind. IBM MQ nutzt diese Bestandsliste zu folgenden Zwecken:
  - Überwachung der aktiven und archivierten Protokolldateien
  - Suche nach Protokolleinträgen, damit während des normalen Betriebs gestellte Anforderungen zum Lesen des Protokolls erfüllt werden können
  - Suche nach Protokolleinträgen, damit Neustartprozesse ausgeführt werden können

  IBM MQ speichert immer dann Informationen in der Bestandsliste, wenn eine Archivprotokolldatei definiert oder eine aktive Protokolldatei wiederverwendet wird. Für aktive Protokolle zeigt die Bestandsliste an, welche voll sind und welche für die Wiederverwendung verfügbar sind. Die Bestandsliste enthält die relative Byteadresse (Relative Byte Address, RBA) eines jeden Teils des Protokolls, das in der jeweiligen Datei gespeichert ist.
- Eine *umlaufende* Bestandsliste aller kürzlich ausgeführten IBM MQ-Aktivitäten. Diese wird beim Neustart eines Warteschlangenmanagers benötigt.

Das Bootstrap-Dataset ist erforderlich, wenn im Warteschlangenmanager ein Fehler auftritt und er neu gestartet werden muss. IBM MQ **muss** ein Bootstrap-Dataset haben. Um die Wahrscheinlichkeit eines Problems beim Neustart zu minimieren, können Sie IBM MQ mit zwei Bootstrap-Datasets konfigurieren, die beide dieselben Informationen aufzeichnen. Die Verwendung von zwei Bootstrap-Datasets wird auch als *Dualmodus* bezeichnet. Falls möglich, sollten die Kopien auf unterschiedlichen Datenträgern gespeichert werden. Dies verringert das Risiko, dass beide Kopien verloren gehen, wenn der Datenträger beschädigt oder gelöscht wird. Die Verwendung von zwei Bootstrap-Datasets ist vorteilhafter als doppelte Schreibvorgänge in einer DASD-Einheit (Direct Access Storage Device).

Das Bootstrap-Dataset wird beim Anpassen von IBM MQ konfiguriert, und Sie können die Bestandsliste mit dem Dienstprogramm zum Ändern des Protokollbestands (CSQJU003) verwalten. Weitere Informationen zu diesem Dienstprogramm finden Sie unter IBM MQ for z/OS verwalten. In der Startprozedur des Warteschlangenmanagers wird durch eine Datendefinitionsanweisung darauf verwiesen.

Normalerweise speichert IBM MQ zwei Kopien des Bootstrap-Datasets. Wenn ein E/A-Fehler auftritt, wird die Zuordnung der fehlgeschlagenen Kopie aufgehoben und der Betrieb mit einem einzelnen Bootstrap-Dataset fortgesetzt. Sie können den Betrieb im Dualmodus anhand der Anleitung unter IBM MQ for z/OS verwalten wiederherstellen.

Die aktiven Protokolle werden bei der Installation von IBM MQ zum ersten Mal im Bootstrap-Dataset registriert. Sie können die aktiven Protokolle nicht ersetzen, ohne den Warteschlangenmanager zu beenden und neu zu starten.

Archivprotokolldateien werden dynamisch zugeordnet. Bei Zuordnung einer solchen Datei wird ihr Name im Bootstrap-Dataset registriert. Die Liste der Archivprotokolldateien wird erweitert, wenn weitere Archive hinzugefügt werden. Wenn die benutzerdefinierte Anzahl von Dateien erreicht ist, werden die aufgelisteten Archivdateien umlaufend wiederverwendet. Die maximale Anzahl von Einträgen beträgt 1000 bei einfacher Archivprotokollierung und 2000 bei doppelter Archivprotokollierung.

Zum Löschen der Archivprotokolldateien können Sie ein Bandmanagementsystem verwenden (IBM MQ verfügt über kein eigenes automatisches Verfahren). Deshalb können sich die Informationen zu einer Archivprotokolldatei unter Umständen auch lange nach dem Löschen dieser Datei durch den Systemadministrator weiterhin im Bootstrap-Dataset befinden.

Umgekehrt ist es auch möglich, dass die maximale Anzahl von Archivprotokolldateien überschritten wurde und die Daten aus dem Bootstrap-Dataset gelöscht wurden, lange bevor die Datei das Ablaufdatum erreicht hat.

Mit dem folgenden WebSphere MQ-Scriptbefehl können Sie den Speicherbereich des Protokolls sowie den Namen der aktiven oder archivierten Protokolldatei angeben, die die niedrigste relative Byteadresse enthält, die für die Wiederherstellung verschiedener Datenträgertypen oder des Warteschlangenmanagers erforderlich ist:

```
DISPLAY USAGE TYPE(DATASET)
```

Wenn im Systemparametermodul angegeben ist, dass Archivprotokolldateien beim Zuordnen katalogisiert werden, verweist das Bootstrap-Dataset für die Informationen, die für spätere Zuordnungen benötigt werden, auf den Katalog der integrierten Katalogfunktion. Andernfalls werden in den Einträgen für den jeweiligen Datenträger im Bootstrap-Dataset die Datenträgerfolgenummer und die Einheiteninformationen registriert, die für spätere Zuordnungen erforderlich sind.

### Version des Bootstrap-Datasets

Das Format des Bootstrap-Datasets ist von seiner Version abhängig. Durch die Nutzung höherer Versionen des Bootstrap-Datasets können neue Funktionen verwendet werden. Folgende Versionen des Bootstrap-Datasets werden von IBM MQ unterstützt:

**Version 1**
Wird von allen IBM MQ-Releases unterstützt. Ein Bootstrap-Dataset der Version 1 unterstützt Protokoll-RBA-Werte mit sechs Byte.

**Version 2**
Unterstützt von IBM MQ 8.0 und höher. Ein Bootstrap-Dataset der Version 2 ermöglicht Protokoll-RBA-Werte mit acht Byte und bis zu 310 Datasets in jeder Kopie des aktiven Protokolls.

V 9.3.0 Standardmäßig aktiviert für Warteschlangenmanager, die in IBM MQ 9.3.0 und höher erstellt wurden.

**Version 3**
Unterstützt von IBM MQ 8.0 und höher. Das Bootstrap-Dataset wird automatisch von Version 2 in Version 3 konvertiert, wenn mehr als 31 Datasets einer Kopie des aktiven Protokolls hinzugefügt werden.

Sie können die Version des Bootstrap-Datasets feststellen, indem Sie das Dienstprogramm zum Ausdrucken der Protokollübersicht (CSQJU004) ausführen. Um ein Bootstrap-Dataset von der Version 1 in Version 2 zu konvertieren, müssen Sie das Bootstrap-Dataset-Konvertierungsdienstprogramm (CSQJUCNV) ausführen.

Weitere Informationen zu Protokoll-RBAs mit sechs Byte und acht Byte finden Sie unter „Größeres Protokoll-Relative Byteadresse” auf Seite 262.

### Archivprotokolldateien und Kopien des Bootstrap-Datasets

Mit jeder neu erstellten Archivprotokolldatei wird auch eine Kopie des Bootstrap-Datasets erstellt. Wenn das Archivprotokoll auf Band gespeichert wird, ist das Bootstrap-Dataset der erste Datensatz auf dem ersten Ausgabedatenträger. Wenn das Archivprotokoll in einer DASD-Einheit gespeichert wird, ist das Bootstrap-Dataset ein separater Datensatz.

Die Dateinamen des Archivprotokolls und der Kopie des Bootstrap-Datasets sind identisch, mit Ausnahme des Qualifikationsmerkmals auf niedrigster Ebene, das im Archivprotokollnamen mit 'A' und im Namen der Bootstrap-Dataset-Kopie mit 'B' beginnt. Beispiel:

**Archivprotokollname**
CSQ.ARCHLOG1.E00186.T2336229. *A* 0000001

**BSDS-Kopiename**
CSQ.ARCHLOG1.E00186.T2336229. *B* 0000001

Wenn beim Kopieren des Bootstrap-Datasets ein Fehler auftritt, wird die Kopie nicht erstellt, die Nachricht CSQJ125E wird ausgegeben und die Auslagerung der neuen Archivprotokolldatei wird ohne die Bootstrap-Dataset-Kopie fortgesetzt.

## z/OS Systemdefinition unter z/OS

In IBM MQ for z/OS werden viele Standardobjektdefinitionen verwendet und eine Beispiel-Jobsteuersprache zum Erstellen dieser Standardobjekte bereitgestellt. In diesem Abschnitt werden diese Standardobjekte sowie die Beispiel-Jobsteuersprache erläutert.

## Systemparameter festlegen

In IBM MQ for z/OS werden mithilfe eines Systemparametermoduls die Protokollierungs-, Archivierungs-, Tracefunktions- und Verbindungsumgebungen gesteuert, die beim Betrieb von IBM MQ verwendet werden. Die Systemparameter werden wie folgt durch drei Assembler-Makros angegeben:

**CSQ6SYSP**
Systemparameter, einschließlich der Einstellung für die Verbindungs- und Tracefunktionsumgebung.

**CSQ6LOGP**
Protokollierungsparameter.

**CSQ6ARVP**
Protokollarchivparameter.

Zum Lieferumfang von IBM MQ for z/OS gehören Standardparametermodule. Wenn darin nicht die von Ihnen gewünschten Werte angegeben sind, können Sie mithilfe des in IBM MQ bereitgestellten Beispiels eigene Parametermodule erstellen. Das Beispiel heißt `thlqual.SCSQPROC(CSQ4ZPRM)`.

Einige Systemparameter können geändert werden, während der Warteschlangenmanager aktiv ist. Siehe die Befehle SET SYSTEM, SET LOG und SET ARCHIVE in MQSC-Befehle.

Weitere Informationen zum Definieren der Parameter finden Sie in folgenden Abschnitten:

- „Systemobjekte für IBM MQ for z/OS definieren” auf Seite 266
- „Warteschlangenmanager unter IBM MQ for z/OS optimieren” auf Seite 271
- „Zum Lieferumfang von IBM MQ for z/OS gehörende Musterdefinitionen” auf Seite 272

**Zugehörige Konzepte**
Beispiel-Initialisierungseingabedateien anpassen
Quellen, aus denen Sie MQSC-und PCF-Befehle unter IBM MQ for z/OS ausgeben können
**Zugehörige Tasks**
z/OS verwalten
Cluster konfigurieren
IBM MQ überwachen

## z/OS Systemobjekte für IBM MQ for z/OS definieren

IBM MQ for z/OS benötigt weitere vordefinierte Objekte für Publish/Subscribe-Anwendungen, Cluster, Kanalsteuerung und andere Systemverwaltungsfunktionen.

Die von IBM MQ for z/OS benötigten Systemobjekte können in die folgenden Kategorien unterteilt werden :

- Publish/Subscribe-Objekte
- Systemstandardobjekte
- Systembefehlsobjekte
- Systemverwaltungsobjekte
- Kanalwarteschlangen
- Clusterwarteschlangen
- Warteschlangen von Gruppen mit gemeinsamer Warteschlange
- Speicherklassen
- Systemobjektwarteschlange für nicht zustellbare Nachrichten definieren
- Standardübertragungswarteschlange
- Interne Warteschlangen
- „Kanalauthentifizierungswarteschlange” auf Seite 271

## Publish/Subscribe-Objekte

Damit Sie Publish/Subscribe-Anwendungen zusammen mit IBM MQ for z/OS verwenden können, müssen zuerst verschiedene Systemobjekte definiert werden. Zum Lieferumfang von IBM MQ gehören Beispieldefinitionen, die Ihnen beim Definieren dieser Objekte helfen. Diese Beispiele werden unter CSQ4INSG beschrieben.

Um Publish/Subscribe-Objekte verwenden zu können, müssen Sie folgende Objekte definieren:

- Eine lokale Warteschlange mit dem Namen 'SYSTEM.RETAINED.PUB.QUEUE', die eine Kopie einer jeden ständigen Veröffentlichung auf dem Warteschlangenmanager enthält. Für jeden vollständigen Themennamen kann maximal eine ständige Veröffentlichung in dieser Warteschlange gespeichert werden. Wenn Ihre Anwendungen ständige Veröffentlichungen zu vielen verschiedenen Themen verwenden oder wenn die Nachrichten der ständigen Veröffentlichungen relativ groß sind, müssen die Speicheranforderungen für diese Warteschlange sorgfältig geplant werden. Dazu gehört im Falle von sehr hohen Speicheranforderungen auch die Erwägung der Maßnahme, die Warteschlange einer eigenen Seitengruppe zuzuordnen. Um die Leistung zu verbessern, sollten Sie für diese Warteschlange den Indextyp der Nachrichten-ID (MSGID) definieren (wie in der als Beispiel bereitgestellten Warteschlangendefinition dargestellt).
- Eine lokale Warteschlange mit dem Namen 'SYSTEM.DURABLE.SUBSCRIBER.QUEUE', die eine permanente Kopie der permanenten Subskriptionen auf dem Warteschlangenmanager enthält. Um die Leistung zu verbessern, sollten Sie für diese Warteschlange den Indextyp der Korrelations-ID (CORRELID) definieren (wie in der als Beispiel bereitgestellten Warteschlangendefinition dargestellt).
- Eine lokale Warteschlange mit dem Namen 'SYSTEM.DURABLE.MODEL.QUEUE', die als Modell für verwaltete permanente Subskriptionen verwendet wird.
- Eine lokale Warteschlange mit dem Namen 'SYSTEM.NDURABLE.MODEL.QUEUE', die als Modell für verwaltete, nicht permanente Subskriptionen verwendet wird.
- Eine Namensliste mit dem Namen 'SYSTEM.QPUBSUB.QUEUE.NAMELIST', die die Namen der von der eingereihten Publish/Subscribe-Schnittstelle überwachten Warteschlangen enthält.
- Eine Namensliste mit dem Namen 'SYSTEM.QPUBSUB.SUBPOINT.NAMELIST', die die Themenobjekte enthält, die von der eingereihten Publish/Subscribe-Schnittstelle zum Abgleichen der Themenobjekte mit den Subskriptionspunkten verwendet wird.
- Ein Thema mit dem Namen 'SYSTEM.BASE.TOPIC', das als Basisthema zum Auflösen von Attributen verwendet wird.
- Ein Thema mit dem Namen 'SYSTEM.BROKER.DEFAULT.STREAM', das von der eingereihten Publish/Subscribe-Schnittstelle als Standarddatenstrom verwendet wird.
- Ein Thema mit dem Namen 'SYSTEM.BROKER.DEFAULT.SUBPOINT', das von der eingereihten Publish/Subscribe-Schnittstelle als RFH2-Standardsubskriptionspunkt verwendet wird.
- Ein Thema mit dem Namen 'SYSTEM.BROKER.ADMIN.STREAM', das von der eingereihten Publish/Subscribe-Schnittstelle als Administratordatenstrom verwendet wird.
- Eine Subskription mit dem Namen 'SYSTEM.DEFAULT.SUB', die als Standardsubskriptionsobjekt Standardwerte für den Befehl 'DEFINE SUB' bereitstellt.

## Systemstandardwertobjekte

Mithilfe von Systemstandardobjekten werden Standardattribute bereitgestellt, wenn Sie beim Definieren eines Objekts nicht den Namen eines anderen Objekts angeben, auf dem die Definition basieren soll.

Die Namen der Standardsystemobjektdefinitionen beginnen mit 'SYSTEM.DEFAULT' oder 'SYSTEM.DEF'. Beispielsweise lautet der Name der lokalen Systemstandardwarteschlange 'SYSTEM.DEFAULT.LOCAL.QUEUE'.

Mit diesen Objekten werden die Systemstandardwerte für die Attribute der folgenden IBM MQ-Objekte definiert:

- Lokale Warteschlangen

- Modellwarteschlangen
- Aliaswarteschlangen
- Ferne Warteschlangen
- Prozesse
- Namenslisten
- Kanäle
- Speicherklassen
- Authentifizierungsdaten

Gemeinsam genutzte Warteschlangen sind ein besonderer Typ von lokalen Warteschlangen, deren Definition also auf 'SYSTEM.DEFAULT.LOCAL.QUEUE' basiert. Beim Definieren einer solchen Warteschlange müssen Sie daran denken, einen Wert für den Namen der Coupling-Facility-Struktur anzugeben, weil die Standarddefinition diesen Wert nicht enthält. Alternativ dazu können Sie eine eigene Standarddefinition als Basis für gemeinsam genutzte Warteschlangen erstellen, damit für alle diese Warteschlangen die erforderlichen Attribute übernommen werden. Dabei ist zu beachten, dass Sie eine gemeinsam genutzte Warteschlange nur auf einem Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange definieren müssen.

## Systembefehlsobjekte

Die Namen der Systembefehlsobjekte beginnen mit 'SYSTEM.COMMAND'. Diese Objekte müssen definiert werden, damit Sie mithilfe von Betriebs- und Steuerkonsolen in IBM MQ Befehle an ein IBM MQ-Subsystem ausgeben können.

Es gibt zwei Systembefehlsobjekte:

1. Die Eingabewarteschlange für Systembefehle ist eine lokale Warteschlange, in die Befehle eingereiht werden, bevor sie vom IBM MQ-Befehlsprozessor verarbeitet werden. Sie muss SYSTEM.COMMAND.INPUT heißen. Einen Aliasnamen namens SYSTEM.ADMIN.COMMAND.QUEUE sollte ebenfalls definiert werden, um die Kompatibilität mit IBM MQ for Multiplatformszu gewährleisten und von IBM MQ Console und administrative REST APIverwendet zu werden.
2. 'SYSTEM.COMMAND.REPLY.MODEL' ist eine Modellwarteschlange, mit der die Systembefehlswarteschlange für zu beantwortende Nachrichten definiert wird.

Für die Verwendung durch IBM MQ Explorer sind zwei zusätzliche Objekte erforderlich:

- die Warteschlange 'SYSTEM.MQEXPLORER.REPLY.MODEL'
- der Kanal 'SYSTEM.ADMIN.SVRCONN'

SYSTEM.REST.REPLY.QUEUE ist die von IBM MQ administrative REST APIverwendete Antwortwarteschlange.

Da Befehle normalerweise mithilfe von nicht permanenten Nachrichten gesendet werden, muss für beide Systembefehlsobjekte das Attribut 'DEFPSIST(NO)' festgelegt werden, damit Anwendungen, die die Objekte verwenden (einschließlich bereitgestellter Anwendungen wie das Dienstprogramm und die Betriebs- und Steuerkonsolen), standardmäßig nicht permanente Nachrichten empfangen. Wenn Sie eine Anwendung haben, die permanente Nachrichten für Befehle verwendet, legen Sie das Attribut 'DEFTYPE(PERMDYN)' für die Warteschlange für zu beantwortende Nachrichten fest, weil die Antwortnachrichten für solche Befehle permanent sind.

## Systemverwaltungsobjekte

Die Namen der Systemverwaltungsobjekte beginnen mit 'SYSTEM.ADMIN'.

Es gibt sieben Systemverwaltungsobjekte:

- die Warteschlange 'SYSTEM.ADMIN.CHANNEL.EVENT'

- die Warteschlange 'SYSTEM.ADMIN.COMMAND.EVENT'
- die Warteschlange 'SYSTEM.ADMIN.CONFIG.EVENT'
- die Warteschlange 'SYSTEM.ADMIN.PERFM.EVENT'
- die Warteschlange 'SYSTEM.ADMIN.QMGR.EVENT'
- die Warteschlange 'SYSTEM.ADMIN.TRACE.ROUTE.QUEUE'
- die Warteschlange 'SYSTEM.ADMIN.ACTIVITY.QUEUE'

## Kanalwarteschlangen

Für die verteilte Steuerung von Warteschlangen müssen folgende Objekte definiert werden:

- Eine lokale Warteschlange mit dem Namen 'SYSTEM.CHANNEL.SYNCQ', die zum Verwalten der Folgenummern und IDs der logischen Arbeitseinheiten (LUWIDs) von Kanälen verwendet wird. Um die Kanalleistung zu verbessern, sollten Sie für diese Warteschlange den Indextyp der Nachrichten-ID (MSGID) definieren (wie in der als Beispiel bereitgestellten Warteschlangendefinition dargestellt).
- Eine lokale Warteschlange mit dem Namen 'SYSTEM.CHANNEL.INITQ', die für Kanalbefehle verwendet wird.

Diese Warteschlangen können Sie nicht als gemeinsam genutzte Warteschlangen definieren.

## Clusterwarteschlangen

Um IBM MQ-Cluster verwenden zu können, müssen Sie folgende Objekte definieren:

- Eine lokale Warteschlange mit dem Namen 'SYSTEM.CLUSTER.COMMAND.QUEUE', die für die Übertragung von Repositoryänderungen zwischen den Warteschlangenmanagern verwendet wird. Nachrichten, die in diese Warteschlange eingereiht werden, enthalten Aktualisierungen der Repository-Daten, die auf die lokale Kopie des Repositorys angewendet werden sollen, oder Anforderungen von Repository-Daten.
- Eine lokale Warteschlange mit dem Namen 'SYSTEM.CLUSTER.REPOSITORY.QUEUE', in der eine permanente Kopie des Repositorys gespeichert wird.
- Eine lokale Warteschlange mit dem Namen 'SYSTEM.CLUSTER.TRANSMIT.QUEUE', die als Übertragungswarteschlange für alle Ziele im Cluster verwendet wird. Um die Leistung zu verbessern, sollten Sie für diese Warteschlange den Indextyp der Korrelations-ID (CORRELID) definieren (wie in der Beispiel-Warteschlangendefinition dargestellt).

Diese Warteschlangen enthalten normalerweise eine große Anzahl von Nachrichten.

Diese Warteschlangen können Sie nicht als gemeinsam genutzte Warteschlangen definieren.

## Warteschlangen von Gruppen mit gemeinsamer Warteschlange

Um gemeinsam genutzte Kanäle und die gruppeninterne Steuerung von Warteschlangen verwenden zu können, müssen Sie folgende Objekte definieren:

- Eine gemeinsam genutzte Warteschlange mit dem Namen 'SYSTEM.QSG.CHANNEL.SYNCQ', in der Synchronisationsinformationen für gemeinsam genutzte Kanäle gespeichert werden.
- Eine gemeinsam genutzte Warteschlange mit dem Namen 'SYSTEM.QSG.TRANSMIT.QUEUE', die als Übertragungswarteschlange für die gruppeninterne Steuerung von Warteschlangen verwendet wird. Wenn Sie eine Gruppe mit gemeinsamer Warteschlange verwenden, müssen Sie diese Warteschlange auf jeden Fall definieren, selbst wenn Sie die gruppeninterne Steuerung von Warteschlangen nicht nutzen.

## Speicherklassen

Es wird empfohlen, die folgenden sechs Speicherklassen zu definieren. Vier davon müssen auf jeden Fall definiert werden, weil sie von IBM MQ benötigt werden. Die Erstellung der anderen Speicherklassendefinitionen wird empfohlen, weil sie in den Beispiel-Warteschlangendefinitionen verwendet werden.

**DEFAULT (erforderlich)**
Diese Speicherklasse wird für alle Nachrichtenwarteschlangen verwendet, die nicht leistungskritisch sind und zu keiner der anderen Speicherklassen passen. Dabei handelt es sich außerdem um die bereitgestellte Standardspeicherklasse, die verwendet wird, wenn Sie beim Definieren einer Warteschlange keine Speicherklasse explizit angeben.

**NODEFINE (erforderlich)**
Diese Speicherklasse wird verwendet, wenn die Speicherklasse, die Sie beim Definieren einer Warteschlange angeben, nicht definiert ist.

**REMOTE (erforderlich)**
Diese Speicherklasse wird hauptsächlich für Übertragungswarteschlangen verwendet, d. h. für systembezogene Warteschlangen mit kurzlebigen, leistungskritischen Nachrichten.

**SYSLNGLV**
Diese Speicherklasse wird für langlebige, leistungskritische Nachrichten verwendet.

**SYSTEM (erforderlich)**
Diese Speicherklasse wird für leistungskritische, systembezogene Nachrichtenwarteschlangen verwendet, z. B. SYSTEM.CHANNEL.SYNQ und SYSTEM.CLUSTER.* Warteschlangen.

**SYSVOLAT**
Diese Speicherklasse wird für kurzlebige, leistungskritische Nachrichten verwendet.

Sie können die Attribute der Speicherklassen ändern und nach Bedarf weitere Speicherklassendefinitionen hinzufügen.

## Systemobjektwarteschlange für nicht zustellbare Nachrichten definieren

Die Warteschlange für nicht zustellbare Nachrichten wird verwendet, wenn das angegebene Nachrichtenziel nicht gültig ist. IBM MQ reiht solche Nachrichten in eine lokale Warteschlange ein, die als Warteschlange für nicht zustellbare Nachrichten bezeichnet wird. Obwohl das Definieren einer Warteschlange für nicht zustellbare Nachrichten nicht obligatorisch ist, sollten Sie dies dennoch als wesentliche Maßnahme betrachten, insbesondere dann, wenn Sie die verteilte Steuerung von Warteschlangen oder eine der IBM MQ-Bridge-Funktionen verwenden.

Die Warteschlange für nicht zustellbare Nachrichten darf **nicht** als gemeinsam genutzte Warteschlange definiert werden. Eine Einreihung in eine lokale Warteschlange auf einem einzigen Warteschlangenmanager kann zur Einreihung in die Warteschlange für nicht zustellbare Nachrichten führen. Wenn die Warteschlange für nicht zustellbare Nachrichten eine gemeinsam genutzte Warteschlange war, könnte eine Steuerroutine der Warteschlange für nicht zustellbare Nachrichten auf einem anderen System die Nachricht verarbeiten und in eine Warteschlange mit demselben Namen einreihen. Da sich diese jedoch auf einem anderen Warteschlangenmanager befindet, wäre es die falsche Warteschlange oder hätte ein anderes Sicherheitsprofil. Wäre die Warteschlange nicht vorhanden, würde keine erneute Verarbeitung stattfinden.

Wenn Sie sich dafür entscheiden, eine Warteschlange für nicht zustellbare Nachrichten zu definieren, müssen Sie dem Warteschlangenmanager auch deren Namen mitteilen, indem Sie ihn im Befehl 'ALTER QMGR DEADQ(*Warteschlangenname*)' angeben. Weitere Informationen finden Sie im Abschnitt Warteschlangenmanagerattribute anzeigen und ändern.

## Standardmäßige Übertragungswarteschlange

Die Standardübertragungswarteschlange wird verwendet, wenn keine andere geeignete Übertragungswarteschlange zum Senden von Nachrichten an einen anderen Warteschlangenmanager zur Verfügung steht. Wenn Sie eine Standardübertragungswarteschlange definieren, müssen Sie auch einen Kanal für die Warteschlange definieren. Wenn Sie dies nicht tun, werden Nachrichten, die in die Standardübert-

ragungswarteschlange eingereiht werden, nicht an den fernen Warteschlangenmanager übertragen, sondern verbleiben in der Warteschlange.

Wenn Sie sich dafür entscheiden, eine Standardübertragungswarteschlange zu definieren, müssen Sie dem Warteschlangenmanager auch deren Namen mitteilen, indem Sie den Befehl 'ALTER QMGR' ausführen.

### Interne Warteschlangen

- **Warteschlange für anstehende Daten**
  - Eine für die interne Verwendung definierte Warteschlange, SYSTEM.PENDING.DATA.QUEUEunterstützt die Verwendung permanenter Subskriptionen in einer Publish/Subscribe-Umgebung von JMS .
- **JMS 2.0 Bereitstellungswarteschlange für Zustellungsverzögerung**
  - Wenn die von JMS 2.0 bereitgestellte Zustellungsverzögerungsfunktion verwendet wird, wird eine interne Staging-Warteschlange ( SYSTEM.DDELAY.LOCAL.QUEUEmuss definiert werden. In dieser Warteschlange speichert der Warteschlangenmanager Nachrichten, die mit einer Zustellungsverzögerung gesendet wurden, so lange, bis die Verzögerung abgelaufen ist und die Nachricht Ihrem Ziel zugestellt werden kann. CSQ4INSG enthält eine Beispieldefinition für diese Warteschlange, die jedoch auskommentiert ist.
  - Wenn Sie eine SYSTEM.DDELAY.LOCAL.QUEUE-Warteschlange definieren, müssen Sie die Attribute STGCLASS, MAXMSGL und MAXDEPTH auf die erwartete Anzahl an Nachrichten setzen, die mit einer Zustellungsverzögerung gesendet werden. Stellen Sie außerdem bei Definition der SYSTEM.DDELAY.LOCAL.QUEUE-Warteschlange sicher, dass nur der Warteschlangenmanager Nachrichten in diese Warteschlange einreihen kann. Keinesfalls sollte eine Benutzer-ID die Berechtigung zum Einreihen von Nachrichten in diese Warteschlange haben.

### Kanalauthentifizierungswarteschlange

Zur internen Verwendung der Kanalauthentifizierung ist die Warteschlange SYSTEM.CHLAUTH.DATA.QUEUE erforderlich. Zum Lieferumfang von IBM MQ gehören Beispieldefinitionen, die Ihnen beim Definieren dieser Objekte helfen. Dieses Beispiel wird in CSQ4INSA beschrieben, wo auch einige Standardregeln definiert werden.

## z/OS Warteschlangenmanager unter IBM MQ for z/OS optimieren

Um grundlegende Leistungsprobleme zu vermeiden, können Sie mit wenigen einfachen Schritten sicherstellen, dass der Warteschlangenmanager optimiert ist.

Es gibt verschiedene Möglichkeiten, um mithilfe der Warteschlangenmanager-Attribute, die mit dem Befehl 'ALTER QMGR' festgelegt werden, die Leistung des Warteschlangenmanagers zu verbessern. In diesem Abschnitt wird erläutert, wie Sie eine Leistungsoptimierung erreichen können, indem Sie die für den Warteschlangenmanager maximal zulässige Anzahl von Nachrichten festlegen oder Systemverwaltungsaufgaben für den Warteschlangenmanager ausführen. IBM MQ SupportPac MP16 - IBM MQ for z/OS -Kapazitätsplanung und -optimierung enthält weitere Informationen zur Leistung und Optimierung.

### Synchronisationspunkte

Eine der Rollen des Warteschlangenmanagers umfasst die Synchronisationspunktsteuerung innerhalb einer Anwendung. Eine Anwendung erstellt eine Arbeitseinheit, die eine beliebige Anzahl von MQPUT- oder MQGET-Aufrufen enthält und mit einem MQCMIT-Aufruf beendet wird.

Wenn die Anzahl der MQPUT- oder MQGET-Aufrufe innerhalb des Geltungsbereichs eines MQCMIT-Aufrufs jedoch ansteigt, nehmen die Leistungseinbußen beim Ausführen der Festschreibung (Commmit)

erheblich zu. Anwendungen sollten im Allgemeinen so konzipiert sein, dass MQPUT/MQGET-Aufrufe nicht für eine große Anzahl von Nachrichten in einem einzigen Synchronisationspunk erfolgen.

Sie können die Anzahl der Nachrichten innerhalb eines einzelnen Synchronisationspunkts mit dem Attribut 'MAXUMSGS' des Warteschlangenmanagers administrativ begrenzen. Wenn dieser Grenzwert von einer Anwendung überschritten wird, empfängt sie MQRC_SYNCPOINT_LIMIT_REACHED für den Aufruf MQPUT, MQPUT1 oder MQGET, der den Grenzwert überschreitet. Die Anwendung sollte dann nach Bedarf MQCMIT oder MQBACK ausgeben.

Der Standardwert von MAXUMSGS ist 10000. Dieser Wert kann gesenkt werden, wenn Sie einen niedrigeren Grenzwert erzwingen möchten, was auch zum Schutz vor Anwendungsschleifen beitragen kann. Bevor Sie den Wert von MAXUMSGS verringern, müssen Sie Ihre vorhandenen Anwendungen kennen, um sicherzustellen, dass sie den Grenzwert nicht überschreiten oder dass sie den Rückkehrcode MQRC_SYNCPOINT_LIMIT_REACHED tolerieren.

## Abgelaufene Nachrichten

Nachrichten, die abgelaufen sind, werden durch den nächsten entsprechenden MQGET-Aufruf gelöscht. Wenn jedoch kein solcher Aufruf stattfindet, werden abgelaufene Nachrichten nicht gelöscht. Dies kann dazu führen, dass sich in manchen Warteschlangen, insbesondere solchen, in denen Nachrichten mithilfe der Nachrichten-ID, Korrelations-ID oder Gruppen-ID abgerufen werden und die für eine Leistungsüberwachung indexiert sind, viele abgelaufene Nachrichten ansammeln. Der Warteschlangenmanager kann alle Warteschlangen in regelmäßigen Abständen auf abgelaufene Nachrichten überprüfen und diese anschließend löschen. Sie können festlegen, ob und wie häufig solche Überprüfungen ausgeführt werden sollen. Hierzu gibt es zwei Möglichkeiten:

**Explizite Anforderung**
In diesem Fall können Sie angeben, welche Warteschlangen wann überprüft werden sollen. Führen Sie den Befehl 'REFRESH QMGR TYPE(EXPIRY)' unter Angabe der Warteschlangen aus, die überprüft werden sollen.

**Regelmäßige Überprüfung**
Sie können im Warteschlangenmanager-Objekt mithilfe des Attributs 'EXPRYINT' ein Ablaufintervall angeben. Der Warteschlangenmanager erfasst und verwaltet Informationen zu den abgelaufenen Nachrichten in der jeweiligen Warteschlange und kann anhand dieser Informationen einen geeigneten Zeitpunkt für die Überprüfung auf abgelaufene Nachrichten ermitteln. Jedes Mal, wenn das Intervall 'EXPRYINT' abläuft, sucht der Warteschlangenmanager nach Warteschlangenkandidaten, für die eine Überprüfung auf abgelaufene Nachrichten sinnvoll ist, und überprüft nur diese Warteschlangen. Es werden also nicht alle Warteschlangen überprüft. Auf diese Weise wird keine Prozessorzeit mit unnötigen Überprüfungen vergeudet.

Gemeinsam genutzte Warteschlangen werden nur von einem Warteschlangenmanager innerhalb der Gruppe mit gemeinsamer Warteschlange überprüft. Normalerweise wird die Überprüfung von dem Warteschlangenmanager ausgeführt, der zuerst neu gestartet wird oder für den das Intervall 'EXPRYINT' zuerst festgelegt wurde.

**Anmerkung:** Für alle Warteschlangenmanager innerhalb der Gruppe mit gemeinsamer Warteschlange muss derselbe Wert als Intervall 'EXPRYINT' festgelegt werden.

## z/OS Zum Lieferumfang von IBM MQ for z/OS gehörende Musterdefinitionen

Dieser Abschnitt dient als Referenz für die Beispiel-Jobsteuersprache und den zum Lieferumfang von IBM MQ for z/OS gehörenden Code.

Die folgenden Beispieldefinitionen gehören zum Lieferumfang von IBM MQ und werden in der Bibliothek 'thlqual.SCSQPROC' bereitgestellt. Mit deren Hilfe können Sie die Systemobjekte definieren und ihre eigenen Objekte anpassen. Einige davon können Sie in die Initialisierungseingabedateien einfügen (eine entsprechende Beschreibung finden Sie unter Initialisierungsbefehle).

| *Tabelle 22. IBM MQ-Beispieldefinitionen für Systemobjekte* | |
|---|---|
| **Initialisierungseingabedatei** | **Name des Beispiels** |
| CSQINP1 | CSQ4INP1<br>CSQ4INPR |
| CSQINP2 | CSQ4INSA<br>CSQ4INYS[1]<br>CSQ4INSX<br>CSQ4INSG<br>CSQ4INSR<br>CSQ4INSS<br>CSQ4INSJ<br>CSQ4INSM<br>CSQ4INYG<br>CSQ4INYR<br>CSQ4INYC<br>CSQ4INYD<br>CSQ4INSC |
| CSQINPT | CSQ4INST<br>CSQ4INYT |
| Andere | CSQ4DISP<br>CSQ4INPX<br>CSQ4IVPQ<br>CSQ4IVPG<br>CSQ4MSTR<br>CSQ4MSRR<br>CSQ4QMIN |

**Anmerkung:**

1. Die Reihenfolge dieser Beispieldefinitionen ist wichtig: Wenn 'INYS', 'INSX' und 'INSG' in falscher Reihenfolge angegeben werden, tritt ein Fehler auf.

## CSQINP1-Beispiele

Verwenden Sie die CSQINP1-Beispieldatei 'thlqual.SCSQPROC(CSQ4INP1)', wenn Sie eine Seitengruppe für jede Nachrichtenklasse verwenden, bzw. 'thlqual.SCSQPROC(CSQ4INPR)', wenn Sie mehrere Seitengruppen für die wichtigsten Nachrichtenklassen verwenden. Die Datei enthält Definitionen für Pufferpools, Zuordnungen zwischen Seitengruppen und Pufferpools und den Befehl 'ALTER SECURITY'. Fügen Sie das Beispiel in die CSQINP1-Verkettung des vom Warteschlangenmanager gestarteten Taskverfahrens ein.

## CSQINP2-Beispiele

### Systemobjektbeispiel 'CSQ4INSG'

Die CSQINP2-Beispieldatei 'thlqual.SCSQPROC(CSQ4INSG)' enthält Definitionen für folgende Systemobjekte, die für die allgemeine Verwendung vorgesehen sind:

- Systemstandardwertobjekte

- Systembefehlsobjekte
- Systemverwaltungsobjekte
- Sonstige Objekte für die Systemverwendung

Die Objekte in diesem Beispiel müssen definiert werden, jedoch nur einmal beim ersten Start des Subsystems. Am besten ist es, die Definitionen zu diesem Zweck in die CSQINP2-Datei einzufügen. Sie bleiben auch bei einer Beendigung und einem Neustart des Warteschlangenmanagers bestehen. Die Objektnamen dürfen nicht geändert werden, während die Attribute der Objekte bei Bedarf geändert werden können.

Wenn die folgenden Bedingungen zutreffen, wird eine Nachricht in die Warteschlange SYSTEM.DURABLE.SUBSCRIBER.QUEUE gestellt (selbst wenn Publish/Subscribe nicht aktiv ist):

- Das QMGR-Attribut PSMODE ist auf DISABLED gesetzt
- Die CSQ4INST-Anweisung `DEFINE SUB('SYSTEM.DEFAULT.SUB')` (des Beispielobjekts) ist vorhanden

Um die Generierung dieser Nachricht zu verhindern, löschen oder kommentieren Sie die Anweisung `DEFINE SUB('SYSTEM.DEFAULT.SUB')` aus.

Die Bereitstellungswarteschlange für die Zustellungsverzögerung von JMS 2.0 , SYSTEM.DDELAY.LOCAL.QUEUE muss nur definiert werden, wenn die Zustellungsverzögerung von JMS 2.0 verwendet wird. Standardmäßig wird die Warteschlangendefinition auskommentiert, wobei Sie diese Auskommentierung bei Bedarf entfernen können.

**Systemobjekt- und Authentifizierungsbeispiel 'CSQ4INSA'**

Die CSQINP2-Beispieldatei thlqual.SCSQPROC(CSQ4INSA) enthält die Definition der Kanalauthentifizierungssystemwarteschlange. Diese Warteschlange enthält die Kanalauthentifizierungssätze. Außerdem enthält sie die Standardkanalauthentifizierungsregeln.

Sie müssen die Objekte in diesem Beispiel definieren, wenn CHLAUTH auf dem Warteschlangenmanager auf ENABLED gesetzt ist und Sie Kanäle ausführen möchten oder wenn Sie CHLAUTH-Sätze festlegen (SET CHLAUTH) oder anzeigen (DISPLAY CHLAUTH) möchten. Sie müssen die Objekte nur einmal beim ersten Start des Subsystems definieren. Am besten ist es, die Definitionen zu diesem Zweck in die CSQINP2-Datei einzufügen. Sie bleiben auch bei einer Beendigung und einem Neustart des Warteschlangenmanagers bestehen, sodass der Warteschlangenmanager-Name nicht geändert werden muss.

**Systemobjektbeispiel 'CSQ4INSS'**

Wenn Sie Gruppen mit gemeinsamer Warteschlange verwenden, können Sie weitere Systemobjekte definieren.

Die Beispieldatei 'thlqual.SCSQPROC(CSQ4INSS)' enthält Beispielbefehle, die für Coupling-Facility-Strukturen ausgeführt werden können, sowie einen Satz an Definitionen für die Systemobjekte, die für gemeinsame Kanäle und die gruppeninterne Steuerung von Warteschlangen erforderlich sind.

Dieses Beispiel kann nicht unverändert verwendet werden, sondern muss vor der Verwendung angepasst werden. Erst dann können Sie dieses Element in die CSQINP2-Datendefinitionsverkettung der Startprozedur des Warteschlangenmanagers einfügen oder als Eingabe für die Funktion 'COMMAND' im Dienstprogramm 'CSQUTIL' verwenden, um die erforderlichen Befehle auszugeben.

Beim Definieren von Gruppenobjekten oder gemeinsam genutzten Objekten müssen Sie diese nur für einen Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange in die CSQINP2-Datendefinitionsverkettung einfügen.

**Systemobjektbeispiel 'CSQ4INSX'**

Wenn Sie die verteilte Steuerung von Warteschlangen und Clustering verwenden, müssen Sie weitere Systemobjekte definieren.

Die Beispieldatei 'thlqual.SCSQPROC(CSQ4INSX)' enthält die erforderlichen Warteschlangendefinitionen. Sie können dieses Element in die CSQINP2-Datendefinitionsverkettung der Startprozedur des Warteschlangenmanagers einfügen oder als Eingabe für die Funktion 'COMMAND' im Dienstprogramm 'CSQUTIL' verwenden, um die erforderlichen DEFINE-Befehle auszugeben.

Es gibt zwei Arten von Objektdefinitionen:

- SYSTEM.CHANNEL.xx - für jede Art der verteilten Steuerung von Warteschlangen
- SYSTEM.CLUSTER.xx - für Clustering

**JMS-Systemobjektbeispiel CSQ4INSJ**

Definiert Warteschlangen, die in der Publish/Subscribe-Domäne von JMS verwendet werden.

**Systemobjektbeispiel 'CSQ4INSM'**

Wenn Sie die erweiterte Nachrichtensicherheit verwenden, müssen Sie zusätzliche Systemobjekte definieren. Die Beispieldatei 'thlqual.SCSQPROC(CSQ4INSM)' enthält die erforderlichen Warteschlangendefinitionen.

**Objektbeispiel 'CSQ4INSR'**

Dieses Beispiel dient zum Definieren von Warteschlangen, die von WebSphere Application Server und Brokern verwendet werden.

**Objektbeispiel 'CSQ4INYD'**

Wenn Sie die verteilte Steuerung von Warteschlangen verwenden, können Sie mithilfe dieses Beispiels eigene Warteschlangen, Prozesse und Kanäle konfigurieren.

Die Beispieldatei 'thlqual.SCSQPROC(CSQ4INYD)' enthält Beispieldefinitionen, mit denen Sie Ihre Objekte für die verteilte Steuerung von Warteschlangen anpassen können. Im Einzelnen enthält sie Folgendes:

- einen Satz von Definitionen für die Sendeseite
- einen Satz von Definitionen für die Empfangsseite
- einen Satz von Definitionen für die Verwendung von Clients

Dieses Beispiel kann nicht unverändert verwendet werden, sondern muss vor der Verwendung angepasst werden. Erst dann können Sie dieses Element in die CSQINP2-Datendefinitionsverkettung der Startprozedur des Warteschlangenmanagers einfügen oder als Eingabe für die Funktion 'COMMAND' im Dienstprogramm 'CSQUTIL' verwenden, um die erforderlichen DEFINE-Befehle auszugeben. (Diese Vorgehensweise wird empfohlen, weil Sie in diesem Fall die Objekte nicht bei jedem Neustart des Warteschlangenmanagers neu definieren müssen.)

**Objektbeispiel 'CSQ4INYC'**

Wenn Sie Clustering verwenden, werden Definitionen, die funktional den Definitionen von Kanälen und fernen Warteschlangen bei der verteilten Steuerung von Warteschlangen entsprechen, bei Bedarf automatisch erstellt. Einige Kanaldefinitionen müssen jedoch auch manuell erstellt werden, z. B. die eines Clusterempfängerkanals für den Cluster sowie eine Clustersenderdefinition für mindestens einen Cluster-Repository-Warteschlangenmanager.

Die Beispieldatei 'thlqual.SCSQPROC(CSQ4INYC)' enthält folgende Beispieldefinitionen, mit denen Sie Ihre Clusteringobjekte anpassen können:

- Definitionen für den Warteschlangenmanager
- Definitionen für den empfangenden Kanal
- Definitionen für den sendenden Kanal
- Definitionen für Clusterwarteschlangen
- Definitionen für Listen von Clustern

Dieses Beispiel kann nicht unverändert verwendet werden, sondern muss vor der Verwendung angepasst werden. Erst dann können Sie dieses Element in die CSQINP2-Datendefinitionsverkettung der Startprozedur des Warteschlangenmanagers einfügen oder als Eingabe für die Funktion 'COMMAND' im Dienstprogramm 'CSQUTIL' verwenden, um die erforderlichen DEFINE-Befehle auszugeben. Diese Vorgehensweise wird empfohlen, weil Sie in diesem Fall die Objekte nicht bei jedem Neustart von IBM MQ neu definieren müssen.

**Objektbeispiel 'CSQ4INYG'**

Die Beispieldatei 'thlqual.SCSQPROC(CSQ4INYG)' enthält folgende Beispieldefinitionen, mit denen Sie Ihre eigenen Objekte für die allgemeine Verwendung anpassen können:

- Warteschlange für nicht zustellbare Nachrichten
- Standardmäßige Übertragungswarteschlange
- CICS-Adapterobjekte

Dieses Beispiel kann nicht unverändert verwendet werden, sondern muss vor der Verwendung angepasst werden. Erst dann können Sie dieses Element in die CSQINP2-Datendefinitionsverkettung der Startprozedur des Warteschlangenmanagers einfügen oder als Eingabe für die Funktion 'COMMAND' im Dienstprogramm 'CSQUTIL' verwenden, um die erforderlichen DEFINE-Befehle auszugeben. Diese Vorgehensweise wird empfohlen, weil Sie in diesem Fall die Objekte nicht bei jedem Neustart von IBM MQ neu definieren müssen.

Zusätzlich zu den hierin enthaltenen Beispieldefinitionen können Sie auch die Systemobjektdefinitionen als Basis für Ihre eigenen Ressourcendefinitionen verwenden. Beispielsweise können Sie eine Arbeitskopie der Warteschlange 'SYSTEM.DEFAULT.LOCAL.QUEUE' mit dem Namen 'MY.DEFAULT.LOCAL.QUEUE' erstellen. Anschließend können Sie alle Parameter der Arbeitskopie nach Bedarf ändern. Danach können Sie nach einer Methode Ihrer Wahl den Befehl 'DEFINE' ausgeben, vorausgesetzt Sie verfügen über die nötige Berechtigung zum Erstellen von Ressourcen dieses Typs.

**Standardmäßige Übertragungswarteschlange**
Lesen Sie zunächst die Beschreibung unter Standardübertragungswarteschlange, bevor Sie entscheiden, ob Sie eine Standardübertragungswarteschlange definieren möchten.

- Wenn Sie sich für die Definition einer Standardübertragungswarteschlange entscheiden, müssen Sie daran denken, auch einen Kanal für die Warteschlange zu definieren.
- Wenn Sie sich gegen die Definition einer Standardübertragungswarteschlange entscheiden, müssen Sie daran denken, im Beispiel die Anweisung 'DEFXMITQ' aus dem Befehl 'ALTER QMGR' zu entfernen.

**CICS-Adapterobjekte**
Dieses Beispiel dient zum Definieren einer Initialisierungswarteschlange mit dem Namen CICS01.INITQ. Diese Warteschlange wird von der in IBM MQ bereitgestellten CKTI-Transaktion verwendet. Sie können den Namen dieser Warteschlange ändern, er muss jedoch mit dem Namen übereinstimmen, der in der CICS-Systeminitialisierungstabelle oder als SYSIN-Korrekturwert in der Anweisung 'INITPARM' angegeben ist.

**Objektbeispiele CSQ4INYS/CSQ4INYR**

Diese Beispiele enthalten Speicherklassendefinitionen für die Verwendung von Folgendem:

- eine Seitengruppe für jede Nachrichtenklasse
- mehrere Seitengruppen für die wichtigsten Nachrichtenklassen

Beispiel: SYSTEM.COMMAND.INPUT verwendet STGCLASS ('SYSVOLAT') und SYSTEM.CLUSTER.TRANSMIT.QUEUE verwendet STGCLASS ('REMOTE'). In CSQ4INYSverwenden beide Speicherklassen dieselbe Seitengruppe. In CSQ4INYRverwenden diese Speicherklassen unterschiedliche Seitengruppen, um die Auswirkungen des Füllens der Übertragungswarteschlange zu verringern.

## CSQINPT-Beispiele

**CSQ4INST**

Die Beispieldatei 'thlqual.SCSQPROC(CSQ4INST)' enthält die Definition für die Standardsubskription des Systems.

Das Objekt in diesem Beispiel muss definiert werden, jedoch nur einmal beim ersten Start der Publish/Subscribe-Engine. Am besten ist es, die Definition zu diesem Zweck in die CSQINPT-Datei einzufügen. Sie bleibt auch bei einer Beendigung und einem Neustart des Warteschlangenmanagers bestehen. Der Objektname darf nicht geändert werden, seine Attribute hingegen schon.

**CSQ4INYT**

Die Beispieldatei 'thlqual.SCSQPROC(CSQ4INYT)' enthält einen Satz von Befehlen, die Sie beim Start der Publish/Subscribe-Engine ausführen können. Mit diesem Beispiel werden Informationen zu Themen und Subskriptionen angezeigt.

## Sonstiges

**Anzeigebeispiel 'CSQ4DISP'**

Die Beispieldatei 'thlqual.SCSQPROC(CSQ4DISP)' enthält einen Satz von generischen DISPLAY-Befehlen, mit denen alle definierten Ressourcen des Warteschlangenmanagers angezeigt werden. Dazu gehören die Definitionen aller IBM MQ-Objekte sowie Definitionen wie z. B. Speicherklassen und Traces. Diese Befehle können zu einer Ausgabe mit erheblichem Umfang führen. Sie können dieses Beispiel in der CSQINP2-Datei oder als Eingabe für die Funktion 'COMMAND' im Dienstprogramm 'CSQUTIL' verwenden.

**Beispiel 'CSQ4INPX'**

Die Beispieldatei 'thlqual.SCSQPROC(CSQ4INPX)' enthält einen Satz von Befehlen, die bei jedem Start des Kanalinitiators ausgeführt werden können. Sie müssen dieses Muster vor der Verwendung anpassen; Sie können es dann in den CSQINPX-Datensatz für den Kanalinitiator aufnehmen.

**Beispiele 'CSQ4IVPQ' und 'CSQ4IVPG'**

Die Beispieldateien 'thlqual.SCSQPROC(CSQ4IVPQ)' und 'thlqual.SCSQPROC(CSQ4IVPG)' enthalten Sätze von DEFINE-Befehlen, die zum Ausführen der Installationsprüfprogramme erforderlich sind.

Sie können diese Beispiele in die CSQINP2-Datei einfügen. Nach erfolgreicher Ausführung der Installationsprüfprogramme brauchen die Programme nicht bei jedem Neustart des Warteschlangenmanagers erneut ausgeführt zu werden. Deshalb müssen diese Beispiele nicht permanent in der CSQINP2-Verkettung verbleiben.

**Beispiele 'CSQ4MSTR' und 'CSQ4MSRR'**

Dies sind Beispiele für gestartete Taskverfahren für den Warteschlangenmanager: thlqual.SCSQPROC(CSQ4MSTR) und thlqual.SCSQPROC(CSQ4MSRR).

CSQ4MSRR verwendet CSQ4INYR in der CSQINP2 -Verkettung, sodass wichtige Warteschlangen auf verschiedene Seitengruppen verteilt werden.

Sie können die Kommentarzeichen entfernen, sodass Sie die CSQMINI-Karte bei Bedarf für neu erstellte Warteschlangenmanager verwenden können.

**Beispiel 'CSQ4QMIN'**

Eine QMINI-Beispieldatei, thlqual.SCSQPROC(CSQ4QMIN).

Details zum QMINI-Dataset und zur Zeilengruppe **`TransportSecurity`** finden Sie unter QMINI data set .

# z/OS Wiederherstellung und Neustart unter z/OS

Über die Links in diesem Abschnitt finden Sie weitere Informationen zu den Funktionen in IBM MQ for z/OS für Neustart und Wiederherstellung.

IBM MQ for z/OS verfügt über leistungsfähige Funktionen für Neustart und Wiederherstellung. In den folgenden Unterabschnitten finden Sie Informationen darüber, wie ein gestoppter Warteschlangenmanager wiederhergestellt wird und was nach seinem Neustart passiert:

- „Änderungen an Daten in IBM MQ for z/OS durchführen” auf Seite 278
- „Aufrechterhalten der Konsistenz in IBM MQ for z/OS” auf Seite 280
- „Vorgänge während der Beendigung in IBM MQ for z/OS” auf Seite 283
- „Vorgänge während des Neustarts und der Wiederherstellung in IBM MQ for z/OS” auf Seite 284
- „Unbestätigte Arbeitseinheiten mit Wiederherstellung lösen” auf Seite 286
- „Wiederherstellung gemeinsam genutzter Warteschlangen” auf Seite 289

**Zugehörige Konzepte**

z/OS IBM MQ for z/OS-Wiederherstellungsaktionen

**Zugehörige Tasks**

Planung von Backups und Wiederherstellung

z/OS z/OS verwalten

**Zugehörige Verweise**

z/OS Nachrichten für IBM MQ for z/OS

## z/OS Änderungen an Daten in IBM MQ for z/OS durchführen

IBM MQ for z/OS muss mit anderen Subsystemen interagieren, um alle Daten konsistent zu halten. Dieser Abschnitt enthält Informationen zu *Arbeitseinheiten mit Wiederherstellung*, zu deren Eigenschaften und ihrer Verwendung bei *Zurücksetzungen*.

### Arbeitseinheiten mit Wiederherstellung

Eine *Arbeitseinheit mit Wiederherstellung* ist die Verarbeitung, die von einem einzelnen Warteschlangenmanager für ein Anwendungsprogramm durchgeführt wird und mit der IBM MQ-Daten von einem Konsistenzzustand in einen anderen geändert werden. Ein *Konsistenzzustand*, der auch als *Synchronisationspunkt* oder *Festschreibungspunkt* bezeichnet wird, ist ein Zeitpunkt, zu dem alle wiederherstellbaren Daten, auf die ein Anwendungsprogramm zugreift, konsistent sind.

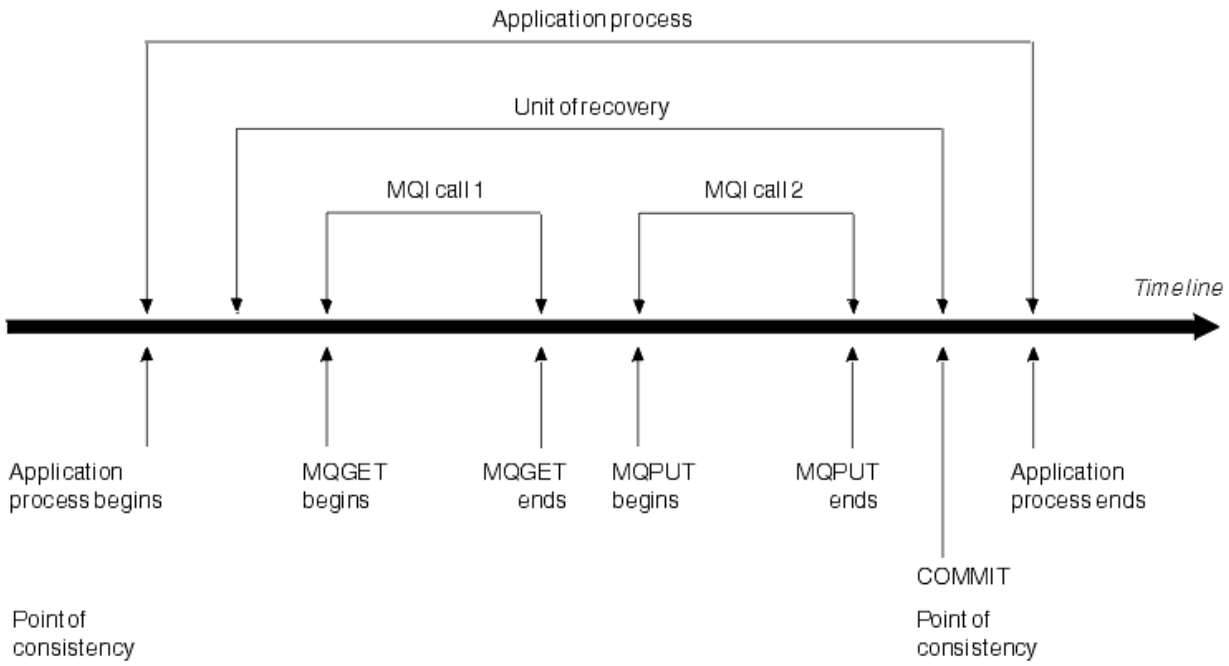



*Abbildung 75. Arbeitseinheit mit Wiederherstellung innerhalb eines Anwendungsprogramms*

Eine Arbeitseinheit mit Wiederherstellung beginnt mit der ersten Änderung an den Daten, die nach dem Start des Programms bzw. nach dem vorherigen Konsistenzzustand auftritt. Sie endet mit dem nächsten erreichten Konsistenzzustand. Abbildung 75 auf Seite 279 zeigt die Beziehung zwischen Arbeitseinheiten mit Wiederherstellung, dem Konsistenzzustand und einem Anwendungsprogramm. In diesem Beispiel nimmt das Anwendungsprogramm mithilfe der MQI-Aufrufe 1 und 2 Änderungen an Warteschlangen vor. Das Anwendungsprogramm kann eine oder mehrere Arbeitseinheiten mit Wiederherstellung enthalten. Jede abgeschlossene Arbeitseinheit mit Wiederherstellung endet jedoch mit einem Festschreibungspunkt.

Beispielsweise wird bei einer Finanztransaktion eine Geldsumme von einem Konto auf ein anderes übertragen. Zuerst hebt das Programm die Geldsumme vom ersten Konto, d. h. Konto A, ab. Dann zahlt es den Betrag auf das zweite Konto, d. h. Konto B, ein. Nachdem der Betrag von Konto A abgehoben wurde, sind die beiden Konten inkonsistent, sodass IBM MQ keine Festschreibung durchführen kann. Die Konten werden konsistent, wenn der Betrag Konto B hinzugefügt wird. Nachdem beide Schritte abgeschlossen wurden, kann das Programm mithilfe einer Festschreibung den Konsistenzzustand bekannt geben, wodurch die Änderungen für andere Anwendungsprogramme sichtbar werden.

Bei normaler Beendigung eines Anwendungsprogramms wird automatisch ein Konsistenzzustand festgeschrieben. Manche Programmanforderungen in CICS- und IMS-Programmen führen ebenfalls zu einem Konsistenzzustand, z. B. EXEC CICS SYNCPOINT.

## Arbeitseinheit zurücksetzen

Wenn innerhalb einer Arbeitseinheit mit Wiederherstellung ein Fehler auftritt, macht IBM MQ alle an Daten vorgenommenen Änderungen rückgängig und setzt die Daten auf den Zustand zu Beginn der Arbeitseinheit mit Wiederherstellung zurück, d. h., IBM MQ setzt die Arbeitseinheit zurück. Die Ereignisse werden in Abbildung 76 auf Seite 280 dargestellt.

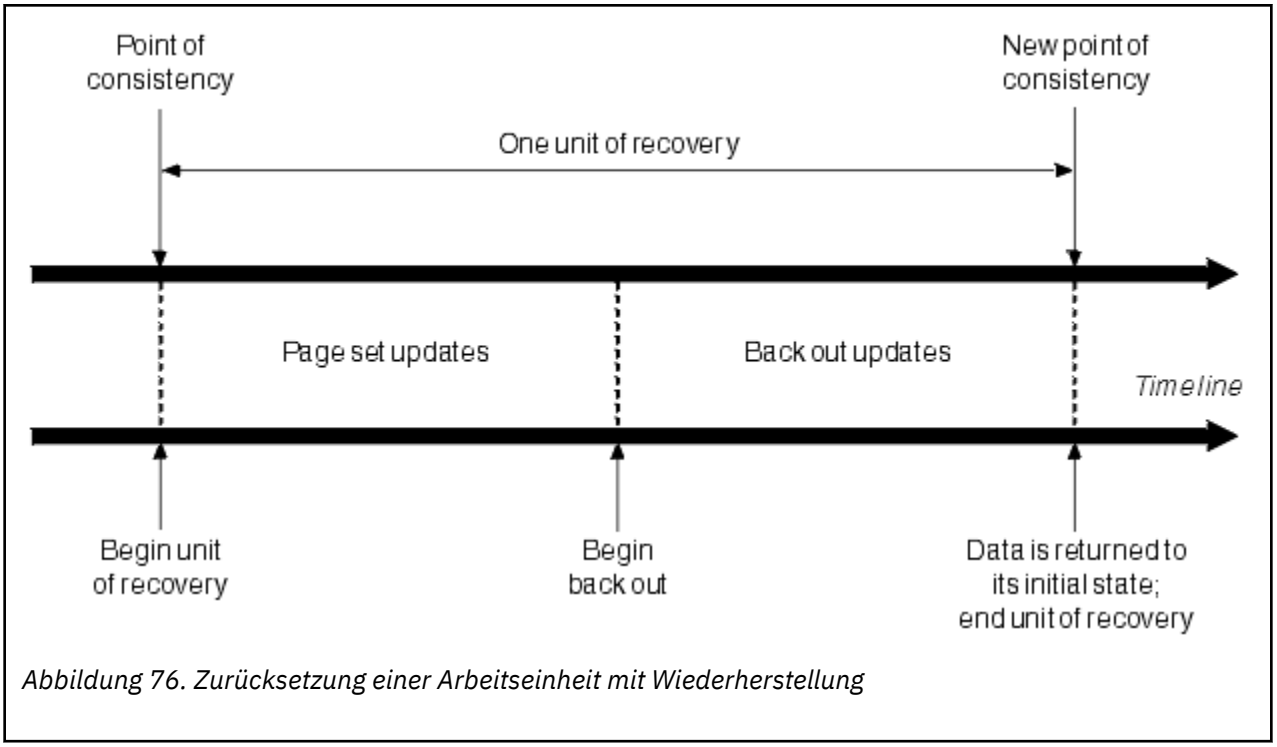



*Abbildung 76. Zurücksetzung einer Arbeitseinheit mit Wiederherstellung*

## z/OS Aufrechterhalten der Konsistenz in IBM MQ for z/OS

Daten in IBM MQ for z/OS müssen mit denen in Batch, CICS, IMS oder TSO konsistent sein. Daten, die in der einen Anwendung geändert werden, müssen auch in der anderen Anwendung geändert werden.

Bevor das eine System die geänderten Daten festschreibt, muss ihm bekannt sein, dass das andere System die entsprechende Änderung vornehmen kann. Deshalb müssen die Systeme miteinander kommunizieren.

Während einer *zweiphasigen Festschreibung* (z. B. in CICS) wird dieser Prozess von einem der Subsysteme koordiniert. Dieses Subsystem wird als *Koordinator* und das andere als *Teilnehmer* bezeichnet. CICS oder IMS ist immer der Koordinator in Interaktionen mit IBM MQund IBM MQ ist immer der Teilnehmer. In einer Batch- oder TSO-Umgebung kann IBM MQ an zweiphasigen Festschreibungsprotokollen teilnehmen, die von z/OS RRS (Resource Recovery Services) koordiniert werden.

Während einer *einphasigen Festschreibung* (z. B. in TSO oder Batch), übernimmt IBM MQ immer die Rolle des Koordinators für die Interaktionen und steuert den kompletten Festschreibungsprozess.

In einer WebSphere Application Server-Umgebung hängt es von der Semantik des JMS-Sitzungsobjekts ab, ob die einphasige oder zweiphasige Festschreibungskoordination verwendet wird.

### Konsistenz mit CICS oder IMS

Die Verbindung zwischen IBM MQ und CICS oder IMS unterstützt folgende Synchronisationspunktprotokolle:

- Zweiphasige Festschreibung - für Transaktionen zur Aktualisierung von Ressourcen, deren Besitzer mehr als ein Ressourcenmanager ist.

  Dies ist das standardmäßige verteilte Synchronisationspunktprotokoll. Es umfasst umfangreichere Protokollierungsfunktionen und Nachrichtenflüsse als die einphasige Festschreibung.
- Einphasige Festschreibung - für Transaktionen zur Aktualisierung von Ressourcen, deren Besitzer ein einzelner Ressourcenmanager (IBM MQ) ist.

  Dieses Protokoll ist für die Protokollierung und für Nachrichtenflüsse optimiert.

- Synchronisationspunktumgehung - für Transaktionen unter Einbindung von IBM MQ, die auf dem Warteschlangenmanager jedoch keine keine Vorgänge auslösen, die einen Synchronisationspunkt erfordern (z. B. Durchsuchen einer Warteschlange).

In jedem dieser Fälle dient CICS oder IMS als Synchronisationspunktmanager.

Bei der Kommunikation zwischen IBM MQ und CICS oder IMS werden folgende Stufen der zweiphasigen Festschreibung verwendet:

1. In Phase 1 ermittelt jedes System unabhängig, ob es genügend Wiederherstellungsinformationen für sein Protokoll erfasst hat und seine Arbeit festschreiben kann.

   Am Ende dieser Phase kommunizieren die Systeme miteinander. Wenn beide der Festschreibung zustimmen, tritt jedes in die nächste Phase ein.
2. In Phase 2 werden die Änderungen permanent gespeichert. Wenn eines der Systeme während der Phase 2 abnormal beendet wird, schließt der Wiederherstellungsprozess die Operation beim Neustart ab.

**Abbildung des zweiphasigen Festschreibungsprozesses**

Abbildung 77 auf Seite 281 zeigt den zweiphasigen Festschreibungsprozess. Im CICS- oder IMS-Koordinator stattfindende Ereignisse werden in der oberen Zeile, in IBM MQ stattfindende Ereignisse in der unteren Zeile dargestellt.

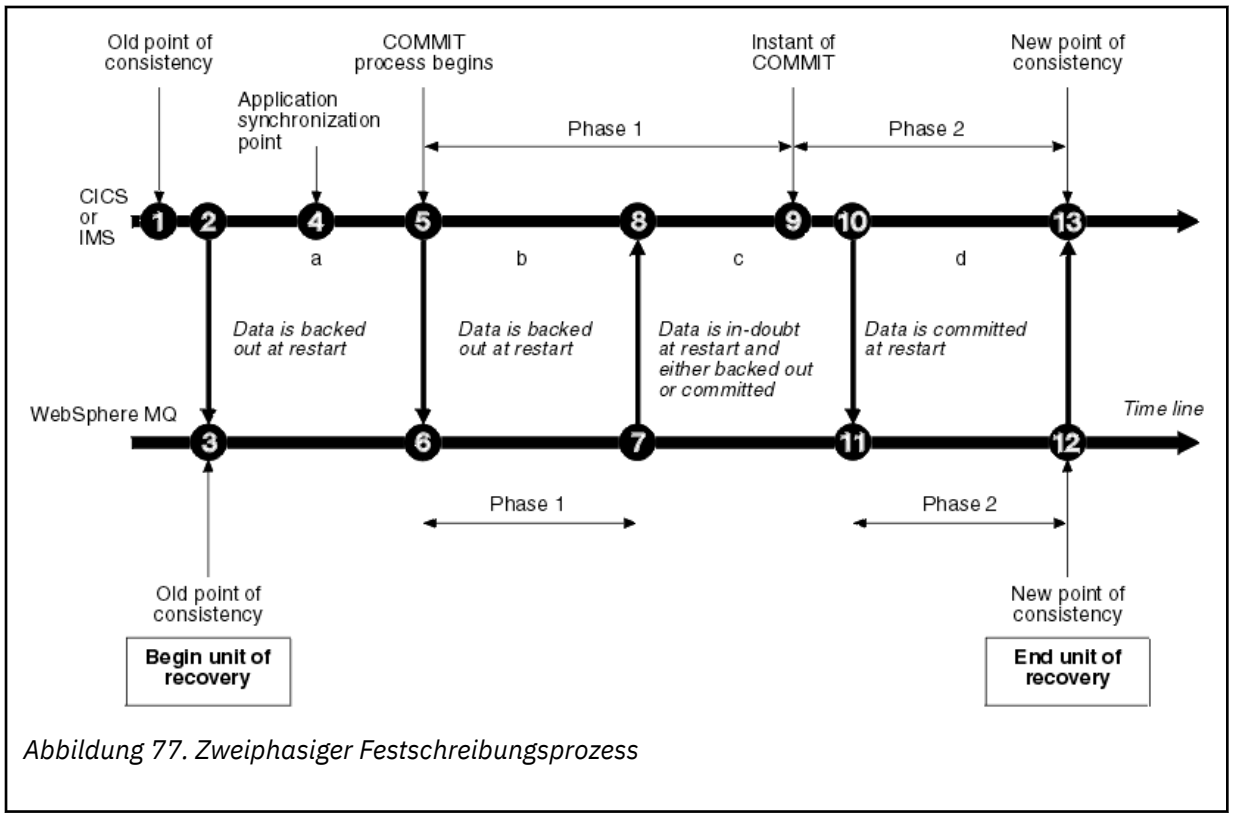



*Abbildung 77. Zweiphasiger Festschreibungsprozess*

Die im folgenden Abschnitt angegebenen Zahlen entsprechen denen in der Abbildung.

1. Die Daten im Koordinator befinden sich im Konsistenzzustand.
2. Ein Anwendungsprogramm im Koordinator ruft IBM MQ auf, um eine Warteschlange durch Hinzufügen einer Nachricht zu aktualisieren.
3. Dadurch wird in IBM MQ eine Arbeitseinheit mit Wiederherstellung gestartet.
4. Die Verarbeitung wird im Koordinator bis zum Erreichen eines Anwendungssynchronisationspunkts fortgesetzt.
5. Danach startet der Koordinator die Verarbeitung der Festschreibung. In CICS-Programmen wird die Festschreibung mit dem Befehl SYNCPOINT oder durch eine normale Beendigung der Anwen-

dung gestartet. In IMS-Programmen wird die Festschreibung mit dem Aufruf CHKP, dem Aufruf SYNC, dem an den Eingabe-/Ausgabe-PCB gerichteten Aufruf GET UNIQUE oder durch eine normale Beendigung der Anwendung gestartet. Daraufhin beginnt Phase 1 der Festschreibungsverarbeitung.

6. Wenn der Koordinator mit Phase 1 der Verarbeitung beginnt, ist dies auch für IBM MQ der Fall.
7. IBM MQ schließt Phase 1 erfolgreich ab, erfasst dies im Protokoll und benachrichtigt den Koordinator.
8. Der Koordinator empfängt die Benachrichtigung.
9. Der Koordinator schließt die Verarbeitung von Phase 1 erfolgreich ab. Jetzt stimmen beide Subsysteme zu, die Datenänderungen festzuschreiben, weil Phase 1 von beiden erfolgreich abgeschlossen wurde und im Falle von Fehlern eine Wiederherstellung möglich wäre. Der Koordinator zeichnet in seinem Protokoll den Festschreibungszeitpunkt auf, d. h. die unwiderrufliche Entscheidung der beiden Subsysteme für die Ausführung der Änderungen.

   Jetzt beginnt der Koordinator mit Phase 2 der Verarbeitung: die tatsächliche Festschreibung.
10. Der Koordinator weist IBM MQ mit einer Benachrichtigung an, Phase 2 zu starten.
11. IBM MQ protokolliert den Beginn von Phase 2.
12. Phase 2 wird erfolgreich abgeschlossen, sodass ein neuer Konsistenzzustand für IBM MQ eintritt. Anschließend benachrichtigt IBM MQ den Koordinator darüber, dass die Verarbeitung von Phase 2 abgeschlossen ist.
13. Der Koordinator schließt die Verarbeitung von Phase 2 ab. Die von den beiden Subsystemen gesteuerten Daten sind jetzt konsistent und stehen anderen Anwendungen zur Verfügung.

## Konsistenz nach abnormaler Beendigung aufrecht erhalten

Wenn ein Warteschlangenmanager nach einer abnormalen Beendigung neu gestartet wird, muss er ermitteln, ob Arbeitseinheiten mit Wiederherstellung, die zum Zeitpunkt der Beendigung aktiv waren, festgeschrieben oder zurückgesetzt werden müssen. Für einige Arbeitseinheiten mit Wiederherstellung verfügt IBM MQ über ausreichend Informationen, um diese Entscheidung zu treffen. Für andere, bei denen das nicht der Fall ist, müssen Informationen vom Koordinator abgerufen werden, sobald die Verbindung wieder hergestellt wird.

Abbildung 77 auf Seite 281 zeigt vier Zeiträume innerhalb der zwei Phasen: a, b, c und d. Der Status einer Arbeitseinheit mit Wiederherstellung hängt von dem Zeitraum ab, in dem die Beendigung erfolgt ist. Folgende Status sind möglich:

**Unvollständig**
Der Warteschlangenmanager wurde beendet, bevor Phase 1 abgeschlossen wurde (Zeitraum a oder b). Beim Neustart setzt IBM MQ die Aktualisierungen zurück.

**Unbestätigt**
Der Warteschlangenmanager wurde nach Abschluss von Phase 1, jedoch vor Beginn von Phase 2 beendet (Zeitraum c). Nur dem Koordinator ist bekannt, ob der Fehler vor oder nach der Festschreibung (Punkt 9) auftrat. Falls er davor auftrat, muss IBM MQ die Änderungen zurücksetzen, falls er danach auftrat, muss IBM MQ die Änderungen vornehmen und festschreiben. Beim Neustart wartet IBM MQ zunächst auf Informationen vom Koordinator, bevor diese Arbeitseinheit mit Wiederherstellung verarbeitet wird.

**Festgeschrieben**
Der Warteschlangenmanager wurde beendet, nachdem er selbst mit der Verarbeitung von Phase 2 begonnen hatte (Zeitraum 'd'). Deshalb nimmt er festgeschriebene Änderungen vor.

**Zurückgesetzt**
Der Warteschlangenmanager wurde beendet, nachdem das Zurücksetzen einer Arbeitseinheit mit Wiederherstellung begonnen hatte, jedoch bevor dieser Prozess abgeschlossen wurde (in der Abbildung nicht dargestellt). Beim Neustart fährt IBM MQ damit fort, die Änderungen zurückzusetzen.

## z/OS Vorgänge während der Beendigung in IBM MQ for z/OS

Ein Warteschlangenmanager wird normalerweise als Reaktion auf den Befehl 'STOP QMGR' beendet. Wenn ein Warteschlangenmanager aus einem anderen Grund beendet wird, handelt es sich um eine abnormale Beendigung.

Beachten Sie, dass IBM MQ während der Beendigung des Warteschlangenmanagers intern den Befehl

```
DISPLAY CONN(*) TYPE(CONN) ALL WHERE (APPLTYPE NE SYSTEMAL)
```

ausgibt, sodass Sie erfahren, welche Threads möglicherweise verhindern, dass der Warteschlangenmanager vollständig beendet wird.

SYSTEMAL gleicht APPLTYPES mit SYSTEM oder CHINIT ab. Daher werden im Fall von Anwendungstypen der Filterung des Befehls DISPLAY CONN, die SYSTEMAL nicht entsprechen, Informationen im Jobprotokoll zurückgegeben, aus denen Threads ersichtlich werden, die eine normale Beendigung verhindern könnten.

**Normale Beendigung**

Bei einer normalen Beendigung stoppt IBM MQ alle Aktivitäten ordnungsgemäß. Sie können IBM MQ im Modus QUIESCE (Wartemodus), FORCE (Erzwungener Modus) oder RESTART (Neustartmodus) stoppen. Die jeweiligen Auswirkungen sind unter Tabelle 23 auf Seite 283 beschrieben.

*Tabelle 23. Beendigung mit QUIESCE, FORCE oder RESTART*

| Thread-Typ | QUIESCE | FORCE | Neustart |
|---|---|---|---|
| Aktive Threads | Ausführen bis zum Abschluss | Zurücksetzen | Zurücksetzen |
| Neue Threads | Können starten | Nicht zugelassen | Nicht zugelassen |
| Neue Verbindungen | Nicht zugelassen | Nicht zugelassen | Nicht zugelassen |

Batch-Anwendungen werden benachrichtigt, wenn der Warteschlangenmanager beendet wird, während die Anwendung noch mit ihm verbunden ist.

In CICS wird ein aktueller Thread nur bis zum Ende der Arbeitseinheit mit Wiederherstellung ausgeführt. Wenn in CICS ein Warteschlangenmanager im Wartemodus (QUIESCE) gestoppt wird, führt dies auch zum Stoppen des CICS-Adapters, sodass eine aktive Task, die mehr als eine Arbeitseinheit mit Wiederherstellung enthält, nicht unbedingt bis zum Abschluss ausgeführt wird.

Wenn Sie einen Warteschlangenmanager im erzwungenen Modus (FORCE) oder Neustartmodus (RESTART) stoppen, werden keine neuen Threads zugewiesen und die Arbeit in verbundenen Threads wird zurückgesetzt. Bei diesen Modi können unbestätigte Arbeitseinheiten mit Wiederherstellung für Threads entstehen, die sich zwischen den Verarbeitungsstufen der Festschreibung befinden. Sie werden erst aufgelöst, wenn IBM MQ erneut mit dem steuernden CICS-, IMS- oder RRS-Subsystem verbunden wird.

Unabhängig vom gewählten Modus werden beim Stoppen eines Warteschlangenmanagers immer folgende Schritte ausgeführt:

1. Verbindungen werden beendet.
2. IBM MQ akzeptiert keine Befehle mehr.
3. IBM MQ stellt sicher, dass eventuell noch ausstehende Aktualisierungen an Seitengruppen abgeschlossen werden.
4. IBM MQ gibt intern den Befehl DISPLAY USAGE aus, damit die relative Byteadresse für den Neustart im z/OS-Konsolenprotokoll aufgezeichnet wird.
5. Der Systemabschlussprüfpunkt wird gesetzt und das Bootstrap-Data-Set entsprechend aktualisiert.

Beendigungen im Wartemodus (QUIESCE) haben keinen Einfluss auf unbestätigte Arbeitseinheiten mit Wiederherstellung. Alle unbestätigten Arbeitseinheiten bleiben unbestätigt.

**Abnormale Beendigung**

Bei einer abnormalen Beendigungen können beispielsweise in folgenden Fällen Daten in einem inkonsistenten Zustand entstehen:

- Eine Arbeitseinheit mit Wiederherstellung wurde unterbrochen, bevor ein Konsistenzzustand erreicht wurde.
- Es wurden keine festgeschriebenen Daten in Seitengruppen gespeichert.
- Es wurden nicht festgeschriebene Daten in Seitengruppen gespeichert.
- Ein Anwendungsprogramm wurde zwischen Phase 1 und 2 des Festschreibungsprozesses unterbrochen, sodass die Arbeitseinheit mit Wiederherstellung unbestätigt blieb.

IBM MQ löst alle aufgrund einer abnormalen Beendigung eventuell vorliegenden Dateninkonsistenzen während des Neustart- und Wiederherstellungsprozesses auf.

## z/OS Vorgänge während des Neustarts und der Wiederherstellung in IBM MQ for z/OS

IBM MQ ermittelt anhand des Wiederherstellungsprotokolls und des Bootstrap-Datasets, was bei einem Neustart wiederhergestellt werden soll. Das Bootstrap-Dataset identifiziert die aktiven und archivierten Protokolldateien sowie die Position des zuletzt erreichten IBM MQ-Prüfpunkts im Protokoll.

### Einführung zu Neustart und Wiederherstellung

Nachdem IBM MQ initialisiert wurde, wird der Neustart des Warteschlangenmanagers wie folgt ausgeführt:

- Protokollinitialisierung
- Wiederherstellung des aktuellen Status
- Aktualisierende Wiederherstellung über Protokolle
- Rückwärtswiederherstellung über Protokolle
- Wiederherstellung des Warteschlangenindexes

Nach Abschluss der Wiederherstellung:

- Festgeschriebene Änderungen spiegeln sich in den Daten wider.
- Unbestätigte Aktivitäten spiegeln sich in den Daten wider. Die Daten sind jedoch gesperrt und können erst verwendet werden, wenn IBM MQ die ausstehende Bestätigung erkannt und entschieden hat.
- Unterbrochene Änderungen, die unvollständig oder abgebrochen sind, wurden aus den Warteschlangen entfernt. Die Nachrichten sind konsistent und können verwendet werden.
- Es wird ein neuer Prüfpunkt gesetzt.
- Für indizierte Warteschlangen mit permanenten Nachrichten (siehe dazu „Warteschlangenindizes wiederherstellen” auf Seite 286) wurden neue Indizes erstellt.

Wenn zwei Bootstrap-Datasets verwendet werden, überprüft IBM MQ die Konsistenz der Zeitmarken in den Bootstrap-Datasets:

- Wenn beide Kopien des Bootstrap-Datasets aktuell sind, überprüft IBM MQ, ob die beiden Zeitmarken identisch sind. Wenn dies nicht der Fall ist, gibt IBM MQ die Nachricht CSQJ120E aus und wird beendet. Dies kann vorkommen, wenn die beiden Kopien des Bootstrap-Datasets sich auf separaten DASD-Datenträgern (Direct Access Storage Device) befinden und eine davon wiederhergestellt wurde, während der Warteschlangenmanager gestoppt wurde. IBM MQ erkennt diese Situation beim Neustart.
- Wenn eine Kopie des Bootstrap-Datasets freigegeben und die Protokollierung mit einem einzelnen Bootstrap-Dataset fortgesetzt wurde, können Fehler auftreten. Wenn beide Kopien auf einem einzelnen

Datenträger verwaltet werden und der Datenträger wiederhergestellt wurde oder wenn beide Kopien des Bootstrap-Datasets unabhängig von einander wiederhergestellt wurden, erkennt IBM MQ die Wiederherstellung möglicherweise nicht. In diesem Fall sind Protokolleinträge, die das Bootstrap-Dataset nicht erkennt, für das System unbekannt.

Batch-Anwendungen werden nicht benachrichtigt, wenn der Neustart ausgeführt wird, *nachdem* die Anwendung eine Verbindung angefordert hat.

## Informationen zum für die Wiederherstellung erforderlichen Protokollbereich

Beim Neustart hängt der Protokolldatenbereich, der gelesen werden muss, von vielen Faktoren ab:

- Zum Zeitpunkt einer abnormalen Beendigung befinden sich normalerweise viele unvollständige Arbeitseinheiten in einem System. Wie zuvor beschrieben, wird das System bei einem Neustart in einen Konsistenzzustand versetzt, wobei im Zuge dieser Verarbeitung gegebenenfalls unvollständige Arbeitseinheiten zurückgesetzt oder gesperrte, unbestätigte Arbeitseinheiten wiederhergestellt werden. Damit die Wiederherstellung von Arbeitseinheiten möglich ist, müssen für alle unvollständigen, zurückgesetzten und unbestätigten Arbeitseinheiten entsprechende Protokolleinträge vorhanden sein. Alte Arbeitseinheiten werden von IBM MQ verzögert (Shunting), damit für die Wiederherstellung der Arbeitseinheiten ein viel kleinerer Protokolldatenbereich benötigt wird.
- Zum Zeitpunkt einer abnormalen Beendigung gibt es normalerweise viele permanente Aktualisierungen, die bislang jedoch nur im Pufferpoolcache zwischengespeichert sind und noch nicht auf dem Datenträger gespeichert wurden. Diese Änderungen müssen aus dem Protokoll gelesen und erneut auf die in den Seitengruppen gespeicherten Daten angewendet werden. Die relationalen Byteadressen zur Wiederherstellung von Seitengruppen innerhalb des Prüfpunkts geben die niedrigste relationale Byteadresse im Protokoll an, die zum Aktualisieren der Seitengruppen in einem konsistenten Zustand erforderlich sind.
- Wenn alte Seitengruppen in das System eingefügt wurden, z. B. indem nach einem Datenträgerfehler eine Seitengruppensicherung eingefügt wurde, dann müssen alle Änderungen, die seit der Sicherung vorgenommen wurden, aus dem Protokoll gelesen werden. Diese Änderungen werden dann auf die Daten angewendet, die in den wiederhergestellten Seitengruppen gespeichert sind. Die relationalen Byteadressen zur Wiederherstellung von Seitengruppen, die sich auf der Seite null der Seitengruppe befinden, geben die niedrigste relationale Byteadresse im Protokoll an, die für die Datenträgerwiederherstellung einer Seitengruppe erforderlich sind.
- Wenn permanente Nachrichten in gemeinsam genutzten Warteschlangen verwendet werden, ist ein Protokolldatenbereich zum Wiederherstellen von Coupling-Facility-Strukturen (CFSTRUCTs) erforderlich, in denen permanente Nachrichten gespeichert sind. Die ältesten Protokolldaten, die zum Ausführen einer CFSTRUCT-Wiederherstellung erforderlich wären, stammen ungefähr aus dem Zeitraum, in dem mit dem Befehl 'CFSTRUCT BACKUP' eine Sicherung ausgeführt wurde.

Während des normalen Betriebs können Sie mit dem Befehl 'DISPLAY USAGE TYPE(DATASET)' den Wiederherstellungsprotokollbereich anzeigen, der diesen Faktoren zugeordnet ist (natürlich ist es auf diese Weise jedoch nicht möglich, Informationen zum erneuten Einfügen alter Seitengruppen anzuzeigen). Um Probleme zu vermeiden, die im Falle einer abnormalen Beendigung eines Warteschlangenmanagers eine Verzögerung bei dessen Neustart verursachen könnten, sollten Sie die Ausgabedaten des Befehls 'DISPLAY USAGE TYPE(DATASET)' regelmäßig überprüfen.

Darüber hinaus gibt der Warteschlangenmanager Informationsnachrichten in Zusammenhang mit diesen Faktoren aus:

- Die Nachrichten CSQJ160I und CSQJ161I warnen vor Arbeitseinheiten mit langen Laufzeiten.
- Die Nachrichten CSQR026I und CSQR027I enthalten Informationen darüber, ob diese Arbeitseinheiten mit langen Laufzeiten erfolgreich verzögert wurden.
- Die Nachrichten CSQE040I und CSQE041E warnen davor, dass die Struktursicherungen relativ alt sind und die Operation 'RECOVER CFSTRUCT' folglich sehr lange dauern würde.

## Anwendungen mit lange laufender Arbeitseinheit ermitteln

Es ist möglich, die Anwendung mit lange laufender Arbeitseinheit zu ermitteln. Führen Sie dazu den Befehl 'DISPLAY CONN' aus.

Der Befehl 'DISPLAY CONN' gibt Verbindungsinformationen für alle Anwendungen zurück, die mit dem Warteschlangenmanager verbunden sind, sowie zusätzliche Informationen, mit deren Hilfe Sie feststellen können, welche Anwendung aktuell eine Arbeitseinheit mit langer Laufzeit aufweist. Die vom Befehl 'DISPLAY CONN' zurückgegebenen Informationen ähneln den Informationen, die vom Befehl 'DISPLAY QSTATUS' zurückgegeben werden. Der Hauptunterschied besteht jedoch darin, dass der Befehl 'DISPLAY CONN' Informationen zu Objekten und transaktionsorientierte Informationen zu einer bestimmten Verbindung anzeigt und nicht Details dazu, welche Verbindungen einem bestimmten Objekt zugeordnet sind.

Der Befehl 'DISPLAY CONN' gibt für jede der verbundenen Anwendungen folgende Informationen zurück:

- Basisinformationen einschließlich der Verbindungs-ID und der PID.
- Transaktionsorientierte Informationen für diese Verbindung, einschließlich des Zeitpunkts (Uhrzeit und Datum), zu dem die Transaktion erstellt wurde (d. h. wann der erste MQGET/PUT-Aufruf innerhalb des Synchronisationspunkts ausgegeben wurde), und des Zeitpunkts, zu dem die Transaktion den ersten Schreibvorgang im Protokoll ausgeführt hat.
- Protokollzeitinformationen, die anzeigen, welche Anwendung noch über eine Arbeitseinheit mit langer Laufzeit verfügt.
- Eine Liste aller Objekte, die die Verbindung aktuell geöffnet hat. Details für jedes einzelne Objekt werden als separate Nachricht unter Angabe der Verbindungs-ID als Schlüssel zurückgegeben. Da es unterschiedliche Objekttypen gibt, z. B. Warteschlangen und Warteschlangenmanager, hängt es vom jeweiligen Objekttyp ab, welche Informationen angezeigt werden.

## Warteschlangenindizes wiederherstellen

Um die Verarbeitungsgeschwindigkeit von MQGET-Operationen in Warteschlangen mit nicht sequenziellem Abruf der Nachrichten zu erhöhen, können Sie angeben, dass IBM MQ einen Index der Nachrichten- oder Korrelations-IDs oder eine Gruppen-ID für alle Nachrichten in der Warteschlange erstellen soll.

Beim Neustart eines Warteschlangenmanagers wird für jede Warteschlange ein solcher Index neu erstellt. Dies gilt nur für permanente Nachrichten, während nicht permanente Nachrichten beim Neustart gelöscht werden. Wenn die indexierten Warteschlangen eine große Anzahl permanenter Nachrichten enthalten, verlängert sich dadurch die Zeit, die für den Neustart des Warteschlangenmanagers erforderlich ist.

Mit dem Parameter QINDXBLD im Makro CSQ6SYSP können Sie festlegen, das Indizes nicht zeitgleich zum Neustart des Warteschlangenmanagers wiederhergestellt werden. Wenn Sie den Parameterwert QINDXBLD=NOWAIT festlegen, wird beim Neustart von IBM MQ nicht auf die Wiederherstellung der Indizes gewartet.

## z/OS Unbestätigte Arbeitseinheiten mit Wiederherstellung lösen

Wenn IBM MQ die Verbindung zu einem anderen Ressourcenmanager verliert, wird normalerweise versucht, beim Neustart alle inkonsistenten Objekte wiederherzustellen.

Wenn IBM MQ die Verbindung zu CICS, IMS oder RRS verliert, wird normalerweise versucht, beim Neustart alle inkonsistenten Objekte wiederherzustellen. Die Informationen, die zum Auflösen unbestätigter Arbeitseinheiten mit Wiederherstellung erforderlich sind, müssen vom koordinierenden System geliefert werden. In den nächsten Abschnitten wird der Auflösungsprozess für verschiedene Umgebungen erläutert.

- Unbestätigte Arbeitseinheiten mit Wiederherstellung in CICS auflösen
- Unbestätigte Arbeitseinheiten mit Wiederherstellung in IMS auflösen
- Unbestätigte Arbeitseinheiten mit Wiederherstellung in RRS auflösen

- Unbestätigte, mit dem Dispositionsmerkmal 'GROUP' versehene Arbeitseinheiten mit Wiederherstellung auflösen

## Unbestätigte Arbeitseinheiten mit Wiederherstellung in CICS auflösen

Unter bestimmten Umständen kann CICS den IBM MQ-Prozess zum Auflösen unbestätigter Arbeitseinheiten mit Wiederherstellung nicht ausführen. Wenn dies der Fall ist, sendet IBM MQ eine der folgenden Nachrichten:

- CSQC404E
- CSQC405E
- CSQC406E
- CSQC407E

Anschließend wird die Nachricht CSQC408I gesendet.

Weitere Informationen zu Inhalt und Bedeutung dieser Nachrichten finden Sie im Handbuch IBM MQ for z/OS -Nachrichten, -Beendigungscodes und -Ursachencodes.

Die Auflösung von unbestätigten Arbeitseinheiten hat keinen Einfluss auf CICS-Ressourcen. CICS übernimmt die Koordination der Wiederherstellung. Beim Neustart von CICS werden die einzelnen Arbeitseinheiten automatisch entweder festgeschrieben oder zurückgesetzt, je nachdem, ob ein Protokolleintrag vorhanden war, der den Beginn der Festschreibung markiert. Auch wenn unbestätigte Objekte vorhanden sind, werden CICS-Ressourcen nicht gesperrt, während die Verbindung von IBM MQ wiederhergestellt wird.

Eine der Funktionen des CICS-Adapters ist die fortwährende Datensynchronisation zwischen CICS und IBM MQ. Wenn ein Warteschlangenmanager, der mit CICS verbunden ist, abnormal beendet wird, kann CICS Arbeitseinheiten festschreiben oder zurücksetzen, ohne dass IBM MQ diese Vorgänge bemerkt. Wenn der WS-Manager erneut gestartet wird, wird diese Arbeit als *unbestätigt* bezeichnet.

IBM MQ kann diese unbestätigten Arbeitseinheiten mit Wiederherstellung erst auflösen (d. h. die an den IBM MQ-Ressourcen vorgenommenen Änderungen festschreiben oder zurücksetzen), wenn die Verbindung zu CICS neu gestartet oder wiederhergestellt wird.

Ein Prozess zum Auflösen unbestätigter Arbeitseinheiten mit Wiederherstellung wird beim Start des CICS-Adapters eingeleitet. Der Prozess beginnt, wenn der Adapter eine Liste der unbestätigten Arbeitseinheiten mit Wiederherstellung anfordert. Danach geschieht Folgendes:

- Der Adapter empfängt von IBM MQ eine Liste der unbestätigten Arbeitseinheiten mit Wiederherstellung für diese Verbindungs-ID und übergibt sie zum Auflösen an CICS.
- CICS vergleicht die Einträge in dieser Liste mit den Einträgen im eigenen Protokoll. Ausgehend von der eigenen Liste ermittelt CICS dann, welche Aktion für die jeweilige unbestätigte Arbeitseinheit mit Wiederherstellung ausgeführt wurde.

Für alle aufgelösten Arbeitseinheiten nimmt IBM MQ die nötigen Aktualisierungen vor und hebt die entsprechenden Sperren auf. Auch nach dem Neustart können jedoch unaufgelöste Arbeitseinheiten bestehen bleiben. Diese müssen nach den Methoden aufgelöst werden, die im Handbuch IBM MQ for z/OS verwalten beschrieben sind.

## Unbestätigte Arbeitseinheiten mit Wiederherstellung in IMS auflösen

Die Auflösung von unbestätigten Arbeitseinheiten mit Wiederherstellung in IMS hat keinen Einfluss auf DL/I-Ressourcen. IMS übernimmt die Koordination der Wiederherstellung. Beim Neustart von IMS werden DL/I-Arbeitseinheiten automatisch entweder festgeschrieben oder zurückgesetzt. Die Entscheidung, ob Onlineregionen (ohne Direktaufruf) festgeschrieben oder zurückgesetzt werden, hängt davon ab, ob im IMS-Protokoll Einträge vom Typ X'3730' und X'3801' vorhanden sind oder nicht. Das Vorhandensein von unbestätigten Arbeitseinheiten mit Wiederherstellung impliziert nicht, dass DL/I-Datensätze gesperrt sind, bis die Verbindung zu IBM MQ wiederhergestellt wurde.

Beim Neustart des Warteschlangenmanagers erstellt IBM MQ eine Liste der unbestätigten Arbeitseinheiten mit Wiederherstellung. IMS erstellt eine eigene Liste mit RRE-Einträgen (Residual Recovery Entries). Die RRE-Einträge werden bei IMS-Prüfpunkten protokolliert, bis alle Einträge aufgelöst wurden.

Während die Verbindung zwischen einer IMS-Region und IBM MQ wiederhergestellt wird, gibt IMS Anweisungen an IBM MQ, ob die von IBM MQ als unbestätigt markierten Arbeitseinheiten festgeschrieben oder zurückgesetzt werden sollen.

Beim Auflösen der unbestätigten Arbeitseinheiten geschieht Folgendes:

1. Wenn IBM MQ erkennt, dass ein Eintrag für die Festschreibung markiert wurde, während er von IMS für die Zurücksetzung markiert wurde, gibt IBM MQ die Nachricht CSQQ010E aus. IBM MQ gibt diese Nachricht für alle zwischen IBM MQ und IMS bestehenden Inkonsistenzen dieses Typs aus.
2. Wenn in IBM MQ weiterhin unbestätigte Arbeitseinheiten verbleiben, gibt der Adapter die Nachricht CSQQ008I aus.

Für alle aufgelösten Arbeitseinheiten nimmt IBM MQ die nötigen Aktualisierungen vor und hebt die entsprechenden Sperren auf.

Für unbestätigte Arbeitseinheiten, die nicht aufgelöst wurden, behält IBM MQ die Sperren bei. Wenn auf diese Weise die Sperren wichtiger Arbeitseinheiten bestehen bleiben, kann es zu einem Rückstand im System kommen. Da die Verbindung aktiv bleibt, können Sie die von IMS erstellten RRE-Einträge auflösen. Die unbestätigten Threads müssen nach den Methoden wiederhergestellt werden, die im Handbuch IBM MQ for z/OS verwalten beschrieben sind.

Sofern keine Software- oder Betriebsprobleme bestehen, die z. B. auf einen Kaltstart von IMS zurückzuführen sind, sollten alle unbestätigten Arbeitseinheiten aufgelöst werden. Damit die Auflösung unbestätigter Arbeitseinheiten durch die IMS-Steuerregion ausgeführt wird, muss einer der folgenden Umstände vorliegen:

1. Beim Herstellen der Verbindung zu IBM MQ; in diesem Fall wird die Auflösung gleichzeitig (synchron) ausgeführt.
2. Bei abnormaler Beendigung eines Programms; in diesem Fall wird die Auflösung nicht gleichzeitig (asynchron) ausgeführt.

## Unbestätigte Arbeitseinheiten mit Wiederherstellung in RRS auflösen

Eine Funktion des RRS-Adapters ist die fortwährende Datensynchronisation zwischen IBM MQ und anderen in RRS eingebundenen Ressourcenmanagern. Wenn ein Fehler auftritt, nachdem IBM MQ Phase 1 des Festschreibungsprozesses abgeschlossen hat und nun auf eine Entscheidung von RRS (als Koordinator für die Festschreibung) wartet, wechselt die Arbeitseinheit mit Wiederherstellung zum unbestätigten Status.

Wenn die Kommunikation zwischen RRS und IBM MQ wiederhergestellt wurde, werden die einzelnen Arbeitseinheiten mit Wiederherstellung von RRS automatisch entweder festgeschrieben oder zurückgesetzt, je nachdem, ob ein Protokolleintrag vorhanden war, der den Beginn der Festschreibung markiert. IBM MQ kann diese unbestätigten Arbeitseinheiten mit Wiederherstellung erst auflösen (d. h. die an den IBM MQ-Ressourcen vorgenommenen Änderungen festschreiben oder zurücksetzen), wenn die Verbindung zu RRS wiederhergestellt wurde.

Unter gewissen Umständen kann RRS unbestätigte Arbeitseinheiten mit Wiederherstellung nicht auflösen. Wenn dies der Fall ist, sendet IBM MQ eine der folgenden Nachrichten an die z/OS-Konsole:

- CSQ3011I
- CSQ3013I
- CSQ3014I
- CSQ3016I

Weitere Informationen zu Inhalt und Bedeutung dieser Nachrichten finden Sie im Handbuch IBM MQ for z/OS -Nachrichten, -Beendigungscodes und -Ursachencodes.

Für alle aufgelösten Arbeitseinheiten mit Wiederherstellung nimmt IBM MQ die nötigen Aktualisierungen vor und hebt die entsprechenden Sperren auf. Auch nach dem Neustart können jedoch unaufgelöste Arbeitseinheiten mit Wiederherstellung bestehen bleiben. Diese müssen nach einer der Methoden aufgelöst werden, die im Handbuch IBM MQ for z/OS verwalten beschrieben sind.

### Unbestätigte, mit dem Dispositionsmerkmal 'GROUP' versehene Arbeitseinheiten mit Wiederherstellung auflösen

Unbestätigte Transaktionen, die mit dem Dispositionsmerkmal 'GROUP' für Arbeitseinheiten mit Wiederherstellung versehen sind, können entweder vom Transaktionskoordinator aufgelöst werden oder von jedem beliebigen Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange, für den das Warteschlangenmanager-Attribut 'GROUPUR' aktiviert ist. Wenn die Verbindung eines Transaktionskoordinators wiederhergestellt wird, fordert er normalerweise eine Liste der ausstehenden unbestätigten Transaktionen an und löst diese anhand der Informationen in seinem Protokoll auf.

Wenn ein Transaktionskoordinator, dessen Verbindung unter Angabe des Dispositionsmerkmals 'GROUP' für Arbeitseinheiten mit Wiederherstellung aufgebaut wurde, die Liste der unbestätigten Transaktionen anfordert, enthält die zurückgegebene Liste alle unbestätigten Transaktionen mit dem Dispositionsmerkmal 'GROUP' für Arbeitseinheiten mit Wiederherstellung, die in der gesamten Gruppe mit gemeinsamer Warteschlange vorhanden sind. Diese Liste wird unabhängig davon erstellt, auf welchem Warteschlangenmanager diese unbestätigten Transaktionen gestartet wurden. Ein Warteschlangenmanager, der eine solche Anforderung verarbeitet, kompiliert die Liste, indem er mit allen anderen aktiven Warteschlangenmanagern in der Gruppe mit gemeinsamer Warteschlange mithilfe der Warteschlange 'SYSTEM.QSG.UR.RESOLUTION.QUEUE' kommuniziert. Dann liest der Warteschlangenmanager die Protokolle aller inaktiven Warteschlangenmanager ab dem letzten Prüfpunkt, um weitere unbestätigte Transaktionen zu finden, die dokumentiert worden wären, sofern die Warteschlangenmanager zu diesem Zeitpunkt aktiv gewesen wären.

Wenn ein Transaktionskoordinator die Auflösung einer unbestätigten Transaktion anfordert, ermittelt der mit dem der Koordinator verbundene Warteschlangenmanager, ob die Transaktion von ihm selbst stammt. Wenn dies der Fall ist, löst er die Transaktion auf dieselbe Weise auf wie Transaktionen mit dem Dispositionsmerkmal 'QMGR' für Arbeitseinheiten mit Wiederherstellung. Wenn die Transaktion hingegen von einem anderen aktiven Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange stammt, wird eine Anforderung zum Ausführen der Auflösung über die Warteschlange 'SYSTEM.QSG.UR.RESOLUTION.QUEUE' an den entsprechenden Warteschlangenmanager weitergeleitet. Falls die Transaktion von einem inaktiven Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange stammt, werden jegliche in der gemeinsam genutzten Warteschlange befindlichen Arbeitseinheiten sofort aufgelöst und darüber hinaus eine Anforderung zur Auflösung aller eventuell in privaten Warteschlangen vorhandenen Arbeitseinheiten in die Warteschlange 'SYSTEM.QSG.UR.RESOLUTION.QUEUE' eingereiht. Der inaktive Warteschlangenmanager verarbeitet diese Anforderung bei seinem nächsten Start, bevor er neue Arbeitseinheiten akzeptiert. In diesem Prozess bleibt die Arbeitseinheit mit Wiederherstellung so lange im Protokoll des ursprünglichen Warteschlangenmanagers als unbestätigt markiert, bis dieser neu gestartet wurde und die Anforderung verarbeitet hat.

## z/OS Wiederherstellung gemeinsam genutzter Warteschlangen

In diesem Abschnitt wird die Wiederherstellung und Ausfallsicherheit verschiedener Komponenten in einer Umgebung mit Unterstützung von Gruppen mit gemeinsamer Warteschlange in IBM MQ beschrieben.


- „Transaktionsorientierte Wiederherstellung” auf Seite 290
- „Peerwiederherstellung” auf Seite 290
- „Definitionen für gemeinsam genutzte Warteschlangen” auf Seite 290
- „Protokollierung” auf Seite 291
- „Coupling-Facility- und Strukturfehler” auf Seite 291
- „Strukturfehlerszenarios” auf Seite 292
- „Ausfallsicherheit bei Coupling-Facility-Verbindungsfehlern” auf Seite 293

- „Ausfallsicherheit bei Coupling-Facility-Verbindungsfehlern verwalten” auf Seite 294
- „Verhalten im Betrieb” auf Seite 297

## Transaktionsorientierte Wiederherstellung

Wenn eine Anwendung einen MQBACK-Aufruf ausgibt oder abnormal beendet wird (z. B. wegen des Befehls EXEC CICS ROLLBACK oder wegen einer abnormalen Beendigung von IMS), sorgen die dreistufig im Warteschlangenmanager gespeicherten Informationen dafür, dass die momentan ausgeführte und deshalb unvollständige Arbeitseinheit rückgängig gemacht wird. Die innerhalb des Synchronisationspunkts für gemeinsam genutzte Warteschlangen ausgeführten Operationen MQPUT und MQGET werden genau so rückgängig gemacht wie die an nicht gemeinsam genutzten Warteschlangen vorgenommenen Aktualisierungen.

## Peerwiederherstellung

Wenn ein Warteschlangenmanager fehlschlägt, wird dessen aktuelle Verbindung zu den Coupling-Facility-Strukturen abnormal beendet. Wenn die Verbindung zwischen der z/OS-Instanz und der Coupling-Facility fehlschlägt (z. B. bei einem Fehler der physischen Verbindung oder bei einer Abschaltung der Coupling-Facility oder der Partition), wird dies auch als abnormale Beendigung der Verbindung zwischen dem Warteschlangenmanager und den entsprechenden Coupling-Facility-Strukturen erkannt. Alle anderen Warteschlangenmanager innerhalb derselben Gruppe mit gemeinsamer Warteschlange, deren Verbindungen zu dieser Struktur bestehen bleiben, erkennen den abnormalen Verbindungsabbau und versuchen eine *Peerwiederherstellung* für den fehlgeschlagenen Warteschlangenmanager in dieser Struktur einzuleiten. Nur einer dieser Warteschlangenmanager kann die Peerwiederherstellung erfolgreich einleiten, doch alle anderen Warteschlangenmanager kooperieren bei der Wiederherstellung der Arbeitseinheiten, deren Besitzer der fehlgeschlagene Warteschlangenmanager ist.

Wenn ein Warteschlangenmanager fehlschlägt, während keine anderen Peers mit dieser Struktur verbunden sind, wird die Wiederherstellung ausgeführt, sobald ein anderer Warteschlangenmanager eine Verbindung zu dieser Struktur herstellt oder der fehlgeschlagene Warteschlangenmanager neu gestartet wird.

Bei der Peerwiederherstellung, englisch auch Peer Level Recovery (PLR), werden die Strukturen nacheinander abgearbeitet, wobei jeder Warteschlangenmanager an der Wiederherstellung mehr als einer Struktur gleichzeitig beteiligt sein kann. Die Gruppe der Peers, die bei der Wiederherstellung unterschiedlicher Strukturen kooperieren, kann jedoch variieren, je nachdem, welche Warteschlangenmanager zum Zeitpunkt des Fehlers mit den jeweiligen Strukturen verbunden waren.

Beim Neustart des fehlgeschlagenen Warteschlangenmanagers wird dessen Verbindung zu den Strukturen, mit denen er zum Zeitpunkt des Fehlers verbunden war, wieder aufgebaut, und Arbeitseinheiten, die bei der Peerwiederherstellung nicht aufgelöst werden konnten, werden wiederhergestellt.

Die Peerwiederherstellung umfasst mehrere Phasen. Während der ersten Phase werden Arbeitseinheiten wiederhergestellt, deren Verarbeitung über den unvollständigen Status hinaus fortgeschritten ist. Dazu gehört möglicherweise das Festschreiben von Nachrichten für festgeschriebene Arbeitseinheiten und das Sperren von Nachrichten für unbestätigte Arbeitseinheiten. Während der zweiten Phase werden Warteschlange überprüft, für die Threads auf dem fehlgeschlagenen Warteschlangenmanager aktiv waren, in Zusammenhang mit unvollständigen Arbeitseinheiten stehende, nicht festgeschriebene Nachrichten rückgängig gemacht und Informationen zu Kennungen, mit denen gemeinsam genutzte Warteschlangen auf dem fehlgeschlagenen Warteschlangenmanager als aktiv gekennzeichnet sind, zurückgesetzt. Dies bedeutet, dass IBM MQ alle Indikatoren dafür, dass der fehlgeschlagene Warteschlangenmanager eine gemeinsam genutzte Warteschlange für die exklusive Eingabe geöffnet hatte und damit anderen aktiven Warteschlangenmanagern die Eingabe in dieser Warteschlange ermöglicht hatte, zurücksetzt.

## Definitionen für gemeinsam genutzte Warteschlangen

Die Warteschlangenobjekte, die die Attribute einer gemeinsam genutzten Warteschlange darstellen, werden in dem gemeinsam genutzten Db2-Repository gespeichert, das von der Gruppe mit gemeinsamer Warteschlange verwendet wird. Stellen Sie sicher, dass es geeignete Vorgehensweisen für die Sicherung und Wiederherstellung der Db2-Tabellen gibt, in denen die IBM MQ-Objekte gespeichert sind. Sie können mit dem IBM MQ-Dienstprogramm CSQUTIL auch WebSphere MQ-Scriptbefehle für die Wiedergabe in einem Warteschlangenmanager erstellen, um IBM MQ-Objekte, einschließlich der in Db2 gespeicherten, gemeinsam genutzten Warteschlangen und Gruppendefinitionen, neu zu definieren.

## Protokollierung

Gruppen mit gemeinsamer Warteschlange können permanente Nachrichten unterstützen, weil die Nachrichten in gemeinsam genutzten Warteschlangen in den Warteschlangenmanagerprotokollen erfasst werden können.

## Coupling-Facility- und Strukturfehler

Bei einer Coupling-Facility-Struktur können zwei verschiedene Fehlertypen auftreten: ein Fehler in Zusammenhang mit der Struktur oder ein Verlust der Konnektivität. Die Sysplex-Services für die gemeinsame Datennutzung (XES) informieren IBM MQ mithilfe eines Strukturfehlerereignisses über den Fehler der Coupling-Facility oder der Coupling-Facility-Struktur. Wenn XES hingegen ein Konnektivitätsverlustereignis erstellt, bedeutet dies nicht unbedingt, dass ein Problem mit der Struktur vorliegt, sondern möglicherweise ist einfach keine Verbindung verfügbar, um mit der Struktur zu kommunizieren. Es ist möglich, dass nicht für alle Warteschlangenmanager ein Konnektivitätsverlustereignis für die jeweilige Struktur auftritt, da dies von der Konfiguration der Verbindungen zur Coupling-Facility abhängt. Ein Konnektivitätsverlustereignis kann auch aufgrund von Bedienerbefehlen auftreten, z. B. 'VARY PATH OFFLINE' oder 'CONFIG CHP OFFLINE'.

Die von IBM MQ verwendeten Coupling-Facility-Strukturen können für die Verwendung einer systemverwalteten Duplexstruktur konfiguriert werden. Dies bedeutet, wenn nur ein einzelner Fehler auftritt, setzt die systemverwaltete Failover-Verarbeitung ein, die den Strukturfehler oder den Konnektivitätsverlust verbirgt und den Warteschlangenmanager nicht über den Fehler benachrichtigt. Wenn jedoch ein Fehler in beiden Instanzen einer Duplexstruktur oder -verbindung auftritt, empfängt der Warteschlangenmanager das entsprechende Ereignis und verarbeitet es in gleicher Weise wie Fehlereignisse für eine Simplexstruktur. Weitere Informationen zur Verarbeitung der Ereignisse durch den Warteschlangenmanager finden Sie unter Szenarien.

In dem unwahrscheinlichen Fall, dass ein Fehler der Coupling-Facility oder Coupling-Facility-Struktur auftritt, gehen alle nicht permanenten Nachrichten, die in den betroffenen Anwendungsstrukturen gespeichert waren, verloren. Permanente Nachrichten können hingegen mit dem Befehl 'RECOVER CFSTRUCT' wiederhergestellt werden. Wenn eine wiederherstellbare Anwendungsstruktur fehlgeschlagen ist, wird jede weitere auf diese Struktur gerichtete Anwendungsaktivität verhindert, bis die Struktur wiederhergestellt wurde.

Um sicherzustellen, dass eine Coupling-Facility-Struktur innerhalb eines angemessenen Zeitraums wiederhergestellt werden kann, sollten Sie häufig Sicherungen mit dem Befehl 'BACKUP CFSTRUCT' ausführen. Sie können entweder angeben, dass die Sicherungen auf beliebigen Warteschlangenmanagern in der Gruppe mit gemeinsamer Warteschlange ausgeführt werden, oder nur einen dedizierten Warteschlangenmanager zum Ausführen aller Sicherungen festlegen. Sie können den Sicherungsprozess auch automatisieren, um sicherzustellen, dass die Sicherungen in regelmäßigen Abständen ausgeführt werden.

Jede Sicherung wird in der aktiven Protokolldatei des Warteschlangenmanagers gespeichert, der die Sicherung ausführt. Das gemeinsam genutzte Db2-Repository protokolliert den Namen der Coupling-Facility-Struktur, die gesichert wird, den Namen des Warteschlangenmanagers, der die Sicherung ausführt, den relativen Byteadressenbereich für diese Sicherung im Warteschlangenmanager-Protokoll und die Uhrzeit der Sicherung.

Da die Verwaltungsstruktur Informationen zu unvollständigen Arbeitseinheiten in gemeinsam genutzten Warteschlangen zum Zeitpunkt eines etwaigen Anwendungsstrukturfehlers enthält, muss sie während der Verarbeitung des Befehls 'RECOVER CFSTRUCT' verfügbar sein. Wenn die Verwaltungsstruktur selbst fehl-

geschlagen ist, müssen alle Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange ihre Verwaltungsstruktureinträge bereits wiederhergestellt haben, bevor Sie den Befehl 'RECOVER CFSTRUCT' ausgeben können.

Warteschlangenmanager erstellen die zugehörigen Verwaltungsstruktureinträge ohne beendet zu werden automatisch erneut. Wenn ein Warteschlangenmanager zum Fehlerzeitpunkt nicht aktiv war, können die zugehörigen Verwaltungsstruktureinträge von einem anderen Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange wiederhergestellt werden, wenn dieser Warteschlangenmanager auf der gleichen oder einer höheren Version ausgeführt wird.

Geben Sie zum Wiederherstellen einer Anwendungsstruktur den Befehl 'RECOVER CFSTRUCT' für den Warteschlangenmanager aus, auf dem die Wiederherstellung ausgeführt werden soll. Sie können eine einzelne Coupling-Facility-Struktur oder mehrere Coupling-Facility-Strukturen gleichzeitig wiederherstellen. Die Wiederherstellung kann mithilfe eines beliebigen Warteschlangenmanagers in der Gruppe mit gemeinsamer Warteschlange ausgeführt werden, d. h., es muss nicht der Warteschlangenmanager sein, auf dem die Sicherung ausgeführt wurde, oder einer, der zuvor mit der fehlgeschlagenen Struktur verbunden war.

Der Befehl 'RECOVER CFSTRUCT' verwendet die Sicherung, deren Speicherposition im Db2-Repository gespeichert ist (deshalb muss Db2 auf dem zur Wiederherstellung verwendeten Warteschlangenmanager verfügbar sein), und stellt diese Sicherung bis zum Fehlerpunkt wieder her.

Dazu wendet der Befehl 'RECOVER CFSTRUCT' die Protokolleinträge aller Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange, die zwischen dem Beginn der Sicherung und dem Fehlerzeitpunkt eine MQPUT- oder MQGET-Operation ausgeführt haben, auf jede gemeinsam genutzte Warteschlange an, die der Coupling-Facility-Struktur zugeordnet ist. Bei der Zusammenführung der Protokolle, die sich dabei ergibt, muss möglicherweise eine erhebliche Menge von Protokolldaten gelesen werden, da sämtliche Protokolldaten aller teilnehmenden Warteschlangenmanager gelesen werden, die seit der Sicherung geschrieben wurden. Deshalb wird dringend empfohlen, häufig Sicherungen auszuführen (z. B. jede Stunde), insbesondere wenn sich große Nachrichten in der Sicherung befinden.

## Strukturfehlerszenarios

### Szenarios

Wenn für eine Coupling-Facility-Struktur ein Fehler festgestellt wird, hängt es von folgenden Faktoren ab, welche Aktion die mit ihr verbundenen Warteschlangenmanager ausführen:

- Der Fehlertyp, der von der systemübergreifenden erweiterten z/OS-Services-Komponente (XES-Komponente) an IBM MQ übermittelt wird.
- Der Strukturtyp (Anwendungs- oder Verwaltungsstruktur).
- Die Warteschlangenmanagerebene
- Der Wert des Parameters CFLEVEL für das IBM MQ CFSTRUCT-Objekt (2, 3, 4 oder 5. Nicht identisch mit dem Wert des Parameters CFLEVEL des CFCC-Mikrocodes.)
- Das Attribut RECAUTO eines IBM MQ CFSTRUCT-Objekts auf Ebene CFLEVEL(5)

In den folgenden Szenarien wird beschrieben, was passiert, wenn ein Fehler in der Verwaltungsstruktur festgestellt wird:

- Wenn ein Strukturfehlerereignis für die Verwaltungsstruktur empfangen wird, wird die Struktur erneut zugeordnet und automatisch wiederhergestellt, ohne dass der Warteschlangenmanager beendet wird. Eine neue Instanz der Struktur wird von XES zugeordnet, sobald ein Warteschlangenmanager versucht, eine Verbindung zu ihr herzustellen.

  Wenn der Warteschlangenmanager eine Verbindung zu der neuen Instanz der Struktur hergestellt hat, speichert er die Einträge für sich selbst in der Struktur. Dieser Verarbeitungsschritt wird vom Warteschlangenmanager ausgeführt und ist nicht Teil der XES-Wiederherstellungsverarbeitung.

  Wenn ein Warteschlangenmanager zum Fehlerzeitpunkt nicht aktiv war oder vor Abschluss der Wiederherstellung eines Teils der Verwaltungsstruktur beendet wird, können die Verwaltungsstruk-

tureinträge von einem anderen Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange wiederhergestellt werden.

Verwaltungsstruktureinträge eines Warteschlangenmanagers können nur von einem Warteschlangenmanager der gleichen oder einer höheren Version wiederhergestellt werden. Wenn Verwaltungsstruktureinträge eines Warteschlangenmanagers nicht von einem anderen Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange wiederhergestellt werden können, starten Sie den Warteschlangenmanager erneut, damit die Wiederherstellung seines Teils der Struktur abgeschlossen werden kann.

Bestimmte Aktionen werden ausgesetzt, bis die Verwaltungsstruktureinträge für alle Warteschlangenmanager wiederhergestellt wurden. Unter anderem werden folgende Aktionen ausgesetzt:

- Öffnen und Schließen von gemeinsam genutzten Warteschlangen.
- Festschreiben oder Auslagern von Arbeitseinheiten mit Wiederherstellung.
- Herstellen oder Trennen von Verbindungen zwischen serialisierten Anwendungen und dem Warteschlangenmanager.
- Sichern oder Wiederherstellen einer Anwendungsstruktur.

Alle serialisierten Anwendungen, die bereits eine Verbindung zum Warteschlangenmanager hergestellt haben, können weiterhin ausgeführt werden. Serialisierte Anwendungen, die versuchen, eine Verbindung mit den Parametern MQCNO_SERIALIZE_CONN_TAG_QSG oder MQCNO_RESTRICT_CONN_TAG_QSG herzustellen, empfangen den Rückgabecode MQRC_CONN_TAG_NOT_USABLE.

Nachdem die Verwaltungsstruktureinträge für den Warteschlangenmanager wiederhergestellt wurden, werden die ausgesetzten Aktionen fortgesetzt.

In den folgenden Szenarien wird beschrieben, was passiert, wenn ein Fehler in der Anwendungsstruktur festgestellt wird:

- Wenn ein Strukturfehlerereignis für eine Anwendungsstruktur auftritt und der Wert des Parameters 'CFLEVEL' 1 oder 2 ist, wird der Warteschlangenmanager beendet. Starten Sie den Warteschlangenmanager erneut. Der erste Warteschlangenmanager, der versucht, eine Verbindung zur Struktur herzustellen, löst die erneute Zuordnung einer neuen Instanz der Struktur durch die systemübergreifenden erweiterten Services (XES) aus.
- Wenn ein Strukturfehlerereignis für eine Anwendungsstruktur auftritt und der Wert des Parameters 'CFLEVEL' 3, 4 oder 5 ist, bleiben die mit der Struktur verbundenen Warteschlangenmanager weiterhin aktiv. Anwendungen, die keine Warteschlangen in der fehlgeschlagenen Struktur verwenden, können mit der normalen Verarbeitung fortfahren.

  Anwendungen, die versuchen, Operationen für die Warteschlangen in fehlgeschlagenen Strukturen auszuführen, empfangen jedoch die Fehlernachricht MQRC_CF_STRUC_FAILED, bis die Struktur erfolgreich wiederhergestellt wurde. Sobald dieser Zeitpunkt eintritt, können die Anwendungen diese Warteschlangen wieder öffnen.

  Die Wiederherstellung der Struktur wird für die mit RECAUTO(YES) definierten CFLEVEL(5)-Anwendungsstrukturen automatisch gestartet. Andernfalls wird die Struktur wiederhergestellt, wenn der Befehl RECOVER CFSTRUCT ausgegeben wird.

## Ausfallsicherheit bei Coupling-Facility-Verbindungsfehlern

### Was bedeutet Ausfallsicherheit bei Coupling-Facility-Verbindungsfehlern

Ausfallsicherheit bei Coupling-Facility-Verbindungsfehlern bezieht sich auf die Fähigkeit der Warteschlangenmanager in einer Gruppe mit gemeinsamer Warteschlange, die Trennung der Verbindung zu einer Coupling-Facility-Struktur zu tolerieren, ohne beendet zu werden. Diese Funktion versucht außerdem, die Struktur in einer anderen Coupling-Facility mit einer zuverlässigeren Verbindung erneut zu erstellen, um so schnell wie möglich wieder Zugriff auf die gemeinsam genutzten Warteschlangen zu erhalten.

**Was ist eine teilweise Trennung der Verbindung?**

IBM MQ definiert eine teilweise Trennung der Verbindung als eine Situation, in der ein oder mehrere Systeme im Sysplex die Verbindung zur Coupling-Facility mit der Struktur, auf die das System zugreift, verlieren, aber mindestens ein System im Sysplex die Verbindung zu der betreffenden Coupling-Facility aufrechterhält.

**Was ist eine vollständige Trennung der Verbindung?**

IBM MQ definiert eine vollständige Trennung der Verbindung als eine Situation, in der kein System im Sysplex eine Verbindung zu der Coupling-Facility und der darin befindlichen Struktur hat.

**Warum sollte diese Funktion aktiviert werden?**

Die Ausfallsicherheit bei Coupling-Facility-Verbindungsfehlern verbessert die Verfügbarkeit von IBM MQ, da nicht gemeinsam genutzte Warteschlangen verfügbar bleiben können, nachdem ein Warteschlangenmanager die Verbindung zu einer oder mehreren Coupling-Facility-Strukturen verloren hat. Warteschlangenmanager, die die Verbindung zu einer Coupling-Facility-Struktur verlieren, versuchen zudem automatisch, die Struktur in einer anderen, verfügbaren Coupling-Facility erneut zu erstellen und verbessern so die Verfügbarkeit der gemeinsam genutzten Warteschlangen in der Gruppe mit gemeinsamer Warteschlange.

**Überlegungen zur Aktivierung dieser Funktion**

Ein Warteschlangenmanager, der die Trennung der Verbindung zu Coupling-Facility-Strukturen toleriert, ohne beendet zu werden, ist möglicherweise längere Zeit nicht in der Lage, eine erneute Verbindung zu einer Coupling-Facility-Struktur herzustellen, wenn keine alternative Coupling-Facility verfügbar ist. Gemeinsam genutzte Warteschlangen, die in einer Struktur definiert sind, zu der die Verbindung getrennt wurde, sind so lange nicht verfügbar, bis die Verbindung zu der Struktur wiederhergestellt wird. In dieser Situation müssen Anwendungen, die Verbindungen zu Mitgliedern der Gruppe mit gemeinsamer Warteschlange herstellen, um mit einer gemeinsam genutzten Warteschlange zu arbeiten, möglicherweise erkennen, dass die gemeinsam genutzten Warteschlangen, auf die sie zugreifen müssen, nicht verfügbar sind. Um diese Situation zu verhindern, wird empfohlen, Warteschlangenmanager so zu konfigurieren, dass sie beendet werden, wenn die Verbindung zu einer Coupling-Facility-Struktur getrennt wird. Diese Beendigung zwingt Anwendungen, eine Verbindung zu einem anderen Mitglied der Gruppe mit gemeinsamer Warteschlange herzustellen, das über eine Verbindung zu den Coupling-Facility-Strukturen verfügt, in denen die von der Anwendung benötigten gemeinsam genutzten Warteschlangen definiert sind.

## Ausfallsicherheit bei Coupling-Facility-Verbindungsfehlern verwalten

**Wie aktiviere ich diese Funktion?**

Um die Ausfallsicherheit bei Coupling-Facility-Verbindungsfehlern zu aktivieren, müssen folgende Schritte ausgeführt werden:

1. Stellen Sie sicher, dass die CFRM-Koppeldatei so formatiert wird, dass eine systemverwaltete Neuerstellung unterstützt wird. Dies ermöglicht es Warteschlangenmanagern, eine systemverwaltete Neuerstellung einer Struktur in einer verfügbaren Coupling-Facility einzuleiten. Mit dem Befehl **DISPLAY XCF,COUPLE,TYPE=CFRM** können Sie das Format der CFRM-Koppeldatei ermitteln. Damit die systemverwaltete Neuerstellung unterstützt wird, sollte die CFRM-Koppeldatei durch folgende Angabe formatiert werden:

   ```
   "ITEM NAME(SMREBLD) NUMBER(1)"
   ```

   Weitere Informationen zum Formatieren einer CFRM-Koppeldatei finden Sie in der Dokumentation z/OS MVS Setting Up a Sysplex .
2. Stellen Sie sicher, dass eine alternative Coupling-Facility verfügbar und in der CFRM-Vorgabenliste für alle IBM MQ-Coupling-Facility-Strukturen enthalten ist. Dies gibt den Warteschlangenmanagern die Möglichkeit, Strukturen in einer alternativen, verfügbaren Coupling-Facility neu zu erstellen, um den Zugriff auf die Strukturen so schnell wie möglich wiederherzustellen.

IBM MQ-Strukturen müssen in der CFRM-Richtlinie mit ENFORCEORDER(NO) definiert werden, damit die Cross-System Coupling-Facility die optimale Coupling-Facility in der Konfiguration wählen kann, wenn IBM MQ die Struktur neu zuordnen muss.

Weitere Informationen zu Strukturvorgabenlisten finden Sie in der Dokumentation z/OS MVS Setting Up a Sysplex .

3. Ändern Sie alle Coupling-Facility-Anwendungsstrukturen, die eine Trennung der Verbindung zu CFLEVEL(5) tolerieren müssen. Dies ist die Mindestebene, die eine Verbindungstrennung tolerieren kann.
4. Ermitteln Sie die für die Attribute **QMGR CFCONLOS** und **CFSTRUCT CFCONLOS** erforderlichen Werte und ändern Sie diese entsprechend. Das Attribut **QMGR CFCONLOS** steuert, ob eine Trennung der Verbindung zur Verwaltungsstruktur toleriert wird, und das Attribut **CFSTRUCT CFCONLOS** steuert, ob eine Verbindungstrennung von jeder Coupling-Facility-Anwendungsstruktur toleriert wird. Wenn die Standardwerte für diese Attribute beibehalten werden, wird der Warteschlangenmanager nach einer Trennung der Verbindung zu einer Coupling-Facility-Struktur beendet.
5. Legen Sie die erforderlichen Werte für das Attribut **CFSTRUCT RECAUTO** für jede Coupling-Facility-Anwendungsstruktur fest und ändern Sie diese entsprechend. Dieses Attribut steuert, ob Coupling-Facility-Strukturen nach einer vollständigen Trennung der Verbindung automatisch mithilfe protokollierter Daten wiederhergestellt werden sollen. Wenn die Standardwerte für dieses Attribut beibehalten werden, findet nach einer vollständigen Trennung der Verbindung keine automatische Wiederherstellung für Anwendungsstrukturen statt.

**Szenario 1 - Trennung der Verbindung zur Verwaltungsstruktur**

Warteschlangenmanager können den Verlust der Verbindung zur Verwaltungsstruktur tolerieren, ohne beendet zu werden.

Wenn ein Warteschlangenmanager, der so konfiguriert wurde, dass er eine Trennung der Verbindung zur Verwaltungsstruktur toleriert, die Verbindung verliert, werden die Verbindungen aller Mitglieder der Gruppe mit gemeinsamer Warteschlange zur Verwaltungsstruktur getrennt. Alle aktiven Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange versuchen anschließend, die Verbindung zur Verwaltungsstruktur wiederherzustellen, was zu einer Neuzuordnung in der Coupling-Facility mit den besten Verbindungen zu allen Systemen im Sysplex und zur Neuerstellung der Verwaltungsstrukturdaten führt.

**Anmerkung:** Dabei muss es sich nicht notwendigerweise um die Coupling-Facility handeln, die über die besten Verbindungen zu allen Systemen verfügt, auf denen sich aktive Warteschlangenmanager befinden.

Falls ein Warteschlangenmanager die Verbindung zur Verwaltungsstruktur nicht wiederherstellen kann, z. B. weil keine der Coupling-Facilitys in der CFRM-Vorgabenliste für die Verwaltungsstruktur verfügbar ist, stehen einige Operationen für gemeinsam genutzte Warteschlangen so lange nicht zur Verfügung, bis der Warteschlangenmanager die Verbindung zur Verwaltungsstruktur erfolgreich wiederherstellen und seine Verwaltungsstrukturdaten neu erstellen kann. Die Verbindungswiederholung geschieht automatisch, sobald eine geeignete Coupling-Facility auf dem System verfügbar wird.

Ein Fehler bei der Herstellung einer Verbindung zur Verwaltungsstruktur während des Warteschlangenmanager-Starts aufgrund eines Mangels an Verbindungen zur Coupling-Facility oder weil keine geeignete Coupling-Facility zur Zuordnung der Struktur verfügbar ist, wird nicht toleriert. Alle aktiven Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange versuchen anschließend, die Verbindung zur Verwaltungsstruktur wiederherzustellen, was zu einer Neuzuordnung in einer anderen Coupling-Facility, falls verfügbar, und zur Neuerstellung der Verwaltungsstrukturdaten führt.

**Szenario 2- - Trennung der Verbindung zur Anwendungsstruktur**

Eine Trennung der Verbindung zur Anwendungsstruktur auf **CFLEVEL(5)** oder höher kann toleriert werden, ohne dass der Warteschlangenmanager beendet wird. Warteschlangenmanager, die mit Anwendungsstrukturen auf **CFLEVEL(4)** oder niedriger oder mit Strukturen auf **CFLEVEL(5)**, die nicht für die Tolerierung von Verbindungstrennungen konfiguriert wurden, verbunden sind, werden abnormal mit dem Ursachencode 00C510AB beendet, sobald die Verbindung zur Struktur verloren geht.

Wenn die Verbindung zu einer Anwendungsstruktur verloren geht, die für die Tolerierung von Verbindungstrennungen konfiguriert wurde, werden alle Warteschlangenmanager, die ihre Verbindung zur Struktur verloren haben, getrennt. Wie sich der Warteschlangenmanager anschließend verhält, hängt davon ab, ob es sich um eine teilweise oder vollständige Trennung der Verbindung handelt.

**Teilweise Trennung der Verbindung zu einer Anwendungsstruktur**

Wenn eine teilweise Trennung der Verbindung erkannt wird, versuchen Warteschlangenmanager, die ihre Verbindung zur Struktur verloren haben, eine systemverwaltete Wiederherstellung einzuleiten, um die Struktur in eine andere Coupling-Facility mit besseren Verbindungen zu verschieben. Ist dies erfolgreich, werden sowohl persistente als auch nicht persistente Nachrichten in der Struktur in die andere Coupling-Facility kopiert und wird der Zugriff auf Warteschlangen in der Struktur wiederhergestellt. Warteschlangenmanager, deren Verbindung nicht unterbrochen wurde, bleiben mit der Struktur verbunden, jedoch werden Operationen, die auf die Struktur zugreifen, während des vom System verwalteten Wiederherstellungsprozesses verzögert.

Wenn eine Anwendungsstruktur nicht in einer anderen Coupling-Facility mit besseren Verbindungen neu erstellt werden kann oder einige Warteschlangenmanager auch nach der Neuerstellung in einer anderen Coupling-Facility keine Verbindung zur Struktur haben, sind in der Struktur definierte Warteschlangen für die Warteschlangenmanager, die keine Verbindung zur Struktur haben, so lange weiterhin nicht verfügbar, bis die Verbindung zur Coupling-Facility wiederhergestellt wird. Warteschlangenmanager stellen automatisch eine erneute Verbindung zur Struktur her, sobald diese verfügbar wird, und auch der Zugriff auf gemeinsam genutzte Warteschlangen, die in der Struktur definiert sind, wird wiederhergestellt.

**Vollständige Trennung der Verbindung zu einer Anwendungsstruktur**

Wenn alle MVS-Systeme im Sysplex die Verbindung zu der Coupling-Facility verloren haben, in der die Anwendungsstruktur zugeordnet ist, hebt z/OS bei jedem Versuch, die Verbindung zur Struktur wiederherzustellen, die Zuordnung der Struktur zur Coupling-Facility auf. Der Warteschlangenmanager kann aus verschiedenen Gründen versuchen, die Verbindung zur Struktur wiederherzustellen; dies kann beispielsweise ein Versuch von einer Anwendung sein, eine gemeinsam genutzte Warteschlange zu öffnen, oder eine Benachrichtigung vom System, dass neue Coupling-Facility-Ressourcen zur Verfügung gestellt wurden. Es ist deshalb wahrscheinlich, dass nach einer vollständigen Trennung der Verbindung zu einer Anwendungsstruktur alle nicht persistenten Nachrichten in der betroffenen Struktur verloren gehen.

Wiederherstellbare Anwendungsstrukturen werden nach einer vollständigen Trennung der Verbindung automatisch wiederhergestellt, wenn Sie mit **RECAUTO(YES)** definiert wurden. Die Wiederherstellung wird nahezu unverzüglich gestartet, falls eine alternative Coupling-Facility zur Zuordnung der Struktur verfügbar ist, oder sobald eine solche Coupling-Facility verfügbar wird. Wenn eine Struktur nicht mit **RECAUTO(YES)** definiert wurde, kann die Wiederherstellung durch Absetzen des Befehls **RECOVER CFSTRUCT** gestartet werden. Dieser Befehl stellt alle persistenten Nachrichten in der Struktur wieder her, alle nicht persistenten Nachrichten gehen jedoch verloren. Da bei diesem Prozess das Warteschlangenmanager-Protokoll gelesen wird, kann er einige Zeit dauern. Deshalb wird empfohlen, regelmäßig Struktursicherungen durchzuführen, um die Zeit bis zur Wiederherstellung des Zugriffs auf die gemeinsam genutzten Warteschlangen in der Struktur zu verkürzen.

Warteschlangenmanager versuchen, die Verbindung zu nicht wiederherstellbaren Anwendungsstrukturen neu aufzubauen, sobald eine Anwendung versucht, eine gemeinsam genutzte Warteschlange, die in der Struktur definiert ist, zu öffnen oder eine Benachrichtigung vom System empfangen wird, dass neue Coupling-Facility-Ressourcen zur Verfügung gestellt wurden. Wenn eine geeignete Coupling-Facility zur Zuordnung der Struktur verfügbar ist, wird eine neue Struktur zugeordnet und der Zugriff auf die gemeinsam genutzten Warteschlangen, die in der Struktur definiert sind, wiederhergestellt. Da persistente Nachrichten nicht in Warteschlangen, die in nicht wiederherstellbaren Strukturen definiert sind, eingereiht werden können, gehen alle Nachrichten in den gemeinsam genutzten Warteschlangen verloren.

### Verhalten im Betrieb

Wenn ein Warteschlangenmanager der IBM WebSphere MQ 7.1 oder höher, der für die Tolerierung einer Trennung der Verbindung zu einer bestimmten Coupling-Facility-Struktur konfiguriert ist, die Verbindung verliert, versuchen die Mitglieder der Gruppe mit gemeinsamer Warteschlange, den Fehler automatisch zu beheben und die Verbindung zur Struktur wiederherzustellen. Dazu kann es gehören, die Struktur in einer anderen Coupling-Facility mit besseren Verbindungen neu zuzuordnen, sofern eine solche Coupling-Facility verfügbar ist. Es kann jedoch trotzdem ein Bedienereingriff erforderlich sein, um die Verbindung wiederherzustellen.

Normalerweise ist eine der folgenden Bedieneraktionen erforderlich:

1. Beseitigung der Ursache des Fehlers, der zur Trennung der Verbindung geführt hat
2. Sicherstellung, dass eine Coupling-Facility, in der die IBM MQ-Strukturen zugeordnet werden können, auf allen Systemen im Sysplex verfügbar ist

Alle Strukturen, die nach einer Trennung der Verbindung automatisch in einer anderen Coupling-Facility neu zugeordnet wurden, können in die Coupling-Facility mit der optimalen Verbindung für alle Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange verschoben werden. Bei Bedarf kann dies mit dem Befehl **SETXCF START,REBUILD** zur systemverwalteten Neuerstellung eingeleitet werden, wie in z/OS MVS- System-Befehlsreferenz dokumentiert.

Im Falle einer teilweisen Trennung der Verbindung zu einer Anwendungsstruktur versuchen die Warteschlangenmanager, die ihre Verbindung zur Struktur verloren haben, eine systemverwaltete Neuerstellung einzuleiten. Bei diesem Prozess wird die Struktur nur dann in einer anderen Coupling-Facility zugeordnet, wenn die betreffende Coupling-Facility über Verbindungen zu allen aktiven Warteschlangenmanagern, die aktuell mit der Struktur verbunden sind, verfügt. Deshalb ist es möglich, dass, wenn der Großteil der Warteschlangenmanager in einer Gruppe mit gemeinsamer Warteschlange die Verbindung zu einer Anwendungsstruktur verloren hat, diese Warteschlangenmanager nicht in der Lage sind, die Struktur in einer anderen Coupling-Facility neu zu erstellen, weil es noch Warteschlangenmanager gibt, die nach wie vor mit der ursprünglichen Struktur verbunden sind. In dieser Situation können die Warteschlangenmanager, die nach wie vor mit der ursprünglichen Struktur verbunden sind, heruntergefahren werden, damit die Struktur neu erstellt werden kann; alternativ kann der Befehl **RESET CFSTRUCT ACTION(FAIL)** ausgegeben werden, um die Struktur als fehlgeschlagen zu kennzeichnen. Die Wiederherstellung für Anwendungsstrukturen kann durch Ausgabe des Befehls **RECOVER CFSTRUCT** eingeleitet werden.

**Anmerkung:** Wenn die Struktur als fehlgeschlagen gekennzeichnet und wiederhergestellt wird, gehen alle nicht persistenten Nachrichten in der Struktur verloren.

## z/OS Sicherheitskonzepte in IBM MQ for z/OS

Mithilfe dieses Abschnitts können Sie den Stellenwert der Sicherheit für IBM MQ und auch die Auswirkungen verstehen, wenn das System über keine adäquaten Sicherheitseinstellungen verfügt.

### Gründe für den Schutz von IBM MQ-Ressourcen

IBM MQ wickelt den Transfer von potenziell wertvollen Informationen ab. Durch entsprechende Sicherheitsmaßnahmen wird gewährleistet, dass die Ressourcen, die IBM MQ besitzt und verwaltet, vor unbefugtem Zugriff geschützt sind. Ein solcher Zugriff könnte zum Verlust oder zur Offenlegung der Informationen führen.

Sie müssen sicherstellen, dass nicht berechtigte Benutzer oder Prozesse auf keine der folgenden Ressourcen zugreifen oder diese ändern können:

- Verbindungen zu IBM MQ
- IBM MQ-Objekte wie Warteschlangen, Prozesse und Namenslisten
- IBM MQ-Übertragungsverbindungen, d. h. IBM MQ-Kanäle
- IBM MQ-Systemsteuerbefehle
- IBM MQ-Nachrichten

- Kontextinformationen zu Nachrichten

Um die erforderliche Sicherheit zu bieten, verwendet IBM MQ die z/OS System Authorization Facility (SAF), um Berechtigungsanforderungen an einen externen Sicherheitsmanager (ESM), z. B. Security Server (früher bekannt als RACF), weiterzuleiten. IBM MQ führt keine eigene Sicherheitsprüfung aus. Wenn verteilte Steuerungen von Warteschlangen oder Clients verwendet werden, sind möglicherweise zusätzliche Sicherheitsmaßnahmen erforderlich, für die IBM MQ Kanalauthentifizierungsdatensätze, Kanalexits, das Kanalattribut MCAUSER sowie TLS bereitstellt.

Die Entscheidung, den Zugriff auf ein Objekt zu ermöglichen, trifft der ESM, und IBM MQ folgt dieser Entscheidung. Wenn der ESM keine Entscheidung treffen kann, verhindert IBM MQ den Zugriff auf das Objekt.

## Was geschieht, wenn IBM MQ-Ressourcen nicht geschützt werden?

Wenn Sie keine Sicherheitsmaßnahmen einrichten, können sehr wahrscheinlich *alle* Benutzer auf *jede* Ressource zugreifen und sie ändern. Dazu zählen neben lokalen Benutzern auch Benutzer von fernen Systemen, die eine verteilte Steuerung von Warteschlangen oder Clients verwenden, bei denen die Sicherheitsmaßnahmen zur Anmeldung weniger streng sind, als dies normalerweise für z/OS der Fall ist.

Gehen Sie wie folgt vor, um eine Sicherheitsprüfung einzurichten:

- Installieren und aktivieren Sie einen ESM (z. B. Security Server).
- Definieren Sie die MQADMIN-Klasse, wenn Sie nicht Security Server als ESM verwenden.
- Aktivieren Sie die MQADMIN-Klasse.

Überlegen Sie, ob Ressourcennamen mit Groß-/Kleinschreibung vorteilhaft für Ihr Unternehmen wären. Wenn Sie Ressourcennamen mit Groß-/Kleinschreibung in Ihren ESM-Profilen verwenden möchten, müssen Sie die MXADMIN-Klasse definieren und aktivieren.

## Verschlüsselung von z/OS -Dateien

Data Set Encryption (DSE) bietet die Möglichkeit, z/OS-Datasets zu verschlüsseln, sodass die darin enthaltenen Daten nur von Benutzer-IDs, die eine bestimmte Berechtigung besitzen, angezeigt oder geändert werden können. Dies ermöglicht die Verschlüsselung von ruhenden Daten im Dateisystem und verhindert eine unbeabsichtigte Offenlegung von sensiblen Informationen für Benutzer, die über eine berechtigte Geschäftsanforderung und Berechtigungen zum Verwalten der Datasets selbst verfügen.

Vor IBM MQ for z/OS 9.1.4 unterstützt IBM MQ for z/OS nicht die Verwendung von DSE mit den aktiven Protokollen, Seitengruppen und gemeinsam genutzten Nachrichtendateien (Shared Message Data Sets, SMDS), die die primären Persistenzmechanismen für IBM MQ-Nachrichten bereitstellen.

Stattdessen stellt Advanced Message Security eine End-to-End-Verschlüsselungslösung für IBM MQ-Messaging zur Verfügung, die das gesamte IBM MQ-Netz sowie die Verschlüsselung von momentan aktiven oder übertragenen Daten, ruhenden Daten und sogar Daten innerhalb der IBM MQ-Laufzeitprozesse umfasst.

Andere VSAM- und sequenzielle Datasets, die in einem IBM MQ-Subsystem verwendet werden, können mit DSE verschlüsselt werden. For example:

- Bootstrap-Dataset (BSDS)
- Sequenzielle Dateien mit Systemkonfigurationsbefehlen (MQSC-Befehle), die beim Systemstart mit CSQINPx DDNAMEs gelesen werden
- IBM MQ-Archivprotokolle, häufig genutzt zur Langzeitarchivierung von IBM MQ-Protokolldaten zu Prüfzwecken.

Sie können Verschlüsselungen mit DSE durchführen, indem Sie eine Datenklasse zuordnen, die mit einem Dataset-Schlüsselkennsatz definiert wird. Weitere Informationen finden Sie im Abschnitt Protokollarchivierungsspeicher planen.

Ab IBM MQ for z/OS 9.1.4 bietet IBM MQ for z/OS zusätzlich zu der Unterstützung, die von den früheren Versionen geboten wird, Unterstützung für die Verwendung von DSE mit den aktiven Protokollen und Seitengruppen.

Die Verwendung von DSE for gemeinsam genutzte Nachrichtendateien (Shared Message Data Sets, SMDS) wird von IBM MQ for z/OS nicht unterstützt.

Weitere Informationen finden Sie im Abschnitt zur Vertraulichkeit für ruhende Daten in IBM MQ for z/OS mit der Dataset-Verschlüsselung. weitere Informationen hierzu.

**Zugehörige Konzepte**

Sicherheitskonzepte

Kanalauthentifizierungsdatensätze

Berechtigung zum Arbeiten mit IBM MQ-Objekten in z/OS

Verschlüsselte Sicherheitsprotokolle: TLS

**Zugehörige Tasks**

Sicherheit unter z/OS einrichten

Sicherheit auf Verbindungsebene und Sicherheit auf Anwendungsebene vergleichen

**Zugehörige Verweise**

Nachrichten für IBM MQ for z/OS

## z/OS Sicherheitssteuerelemente und -optionen in IBM MQ for z/OS

Sie können angeben, ob die Sicherheit für das gesamte IBM MQ-Subsystem aktiviert werden soll und ob die Sicherheitsprüfungen auf der Ebene eines Warteschlangenmanagers oder einer Gruppe mit gemeinsamer Warteschlange ausgeführt werden sollen. Darüber hinaus können Sie Anzahl von Benutzer-IDs festlegen, die hinsichtlich der Sicherheit der API-Ressourcen überprüft wird.

### Subsystemsicherheit

Mithilfe des Steuerelements für Subsystemsicherheit wird festgelegt, ob überhaupt Sicherheitsprüfungen für den gesamten Warteschlangenmanager ausgeführt werden sollen. Wenn keine Sicherheitsprüfung erforderlich ist (z. B. auf einem Testsystem) oder wenn Sie mit der Sicherheitsstufe auf allen Ressourcen, die eine Verbindung zu IBM MQ herstellen können (einschließlich Clients und Kanälen), zufrieden sind, können Sie die Sicherheitsprüfung für den Warteschlangenmanager oder die Gruppe mit gemeinsamer Warteschlange inaktivieren, damit keine weiteren Sicherheitsprüfungen durchgeführt werden.

Dies ist die einzige Option, mit der die Sicherheit vollständig inaktiviert und darüber hinaus festgelegt werden kann, ob andere Sicherheitsprüfungen ausgeführt werden sollen oder nicht. Das heißt, wenn Sie die Sicherheitsprüfung für den Warteschlangenmanager oder die Gruppe mit gemeinsamer Warteschlange inaktivieren, werden auch keine anderen IBM MQ-Prüfungen durchgeführt. Wenn Sie die Sicherheitsprüfung aktiviert lassen, überprüft IBM MQ die Sicherheitsanforderungen der anderen IBM MQ-Ressourcen.

Sie können die Sicherheit auch für bestimmte Ressourcengruppen aktivieren bzw. inaktivieren, z. B. für Befehle.

### Sicherheitsprüfung auf der Ebene eines Warteschlangenmanagers oder einer Gruppe mit gemeinsamer Warteschlange

Sie können die Sicherheit auf der Ebene eines Warteschlangenmanagers oder einer Gruppe mit gemeinsamer Warteschlange implementieren. Wenn die Sicherheit auf der Ebene einer Gruppe mit gemeinsamer Warteschlange implementiert wird, haben alle Warteschlangenmanager in der Gruppe dieselben Profile. Dies erleichtert und vereinfacht das Sicherheitsmanagement, weil weniger Profile definiert und verwaltet werden müssen. Außerdem wird das Hinzufügen eines neuen Warteschlangenmanagers zur Gruppe mit

gemeinsamer Warteschlange auf diese Weise erleichtert, weil die vorhandenen Sicherheitsprofile dabei einfach übernommen werden.

Sie haben auch die Möglichkeit, eine Kombination aus beiden Verfahren zu implementieren, wenn dies für Ihre Installation erforderlich ist, z. B. bei einer Migration oder wenn ein Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange andere Sicherheitsstufen benötigt als die anderen Warteschlangenmanager in derselben Gruppe.

**Sicherheit auf Ebene der Gruppe mit gemeinsamer Warteschlange**

Die Sicherheitsprüfung auf der Ebene einer Gruppe mit gemeinsamer Warteschlange wird für die gesamte Gruppe ausgeführt. Dadurch wird das Sicherheitsmanagement vereinfacht, weil weniger Sicherheitsprofile definiert werden müssen. Die Berechtigung einer Benutzer-ID zur Verwendung einer bestimmten Ressource wird auf der Ebene der Gruppe mit gemeinsamer Warteschlange verwaltet und ist unabhängig davon, über welchen Warteschlangenmanager der Zugriff der Benutzer-ID auf die Ressource ausgeführt wird.

Wenn beispielsweise eine unter der Benutzer-ID 'SERVER' ausgeführte Serveranwendung Zugriff auf eine Warteschlange mit dem Namen 'SERVER.REQUEST' benötigt und Sie eine Instanz von 'SERVER' in jedem z/OS-Image des Sysplex ausführen möchten, brauchen Sie nicht der ID 'SERVER' die Berechtigung zu erteilen, die Warteschlange 'SERVER.REQUEST' auf jedem Warteschlangenmanager einzeln zu öffnen (Sicherheit auf Warteschlangenmanager-Ebene), sondern können stattdessen die Zugriffsberechtigung auf Ebene der Gruppe mit gemeinsamer Warteschlange erteilen.

Mit Sicherheitsprofilen auf Ebene der Gruppe mit gemeinsamer Warteschlange können Sie alle Arten von Ressourcen schützen, unabhängig davon, ob es sich um lokale oder gemeinsam genutzte Ressourcen handelt.

**Sicherheit auf WS-Managerebene**
Mit Sicherheitsprofilen auf Warteschlangenmanager-Ebene können Sie alle Arten von Ressourcen schützen, unabhängig davon, ob es sich um lokale oder gemeinsam genutzte Ressourcen handelt.

**Kombination beider Ebenen**
Sie können eine Kombination der Sicherheit auf Warteschlangenmanager-Ebene und auf Ebene der Gruppe mit gemeinsamer Warteschlange verwenden.

Sie können Sicherheitseinstellungen auf Ebene der Gruppe mit gemeinsamer Warteschlange für einen bestimmten Warteschlangenmanager überschreiben, der Mitglied dieser Gruppe ist. Dies bedeutet, dass auf einem einzelnen Warteschlangenmanager eine andere Sicherheitsprüfungsstufe ausgeführt wird als auf den anderen Warteschlangenmanagern in der Gruppe.

Weitere Informationen finden Sie unter Profile zur Steuerung der Sicherheit auf Ebene einer Gruppe mit gemeinsamer Warteschlange oder auf Warteschlangenmanager-Ebene.

## Anzahl der überprüften Benutzer-IDs festlegen

RESLEVEL ist ein Sicherheitsserverprofil, mit dem die Anzahl der Benutzer-IDs festgelegt wird, die hinsichtlich der Sicherheit von IBM MQ-Ressourcen überprüft wird. Normalerweise überprüft der Sicherheitsserver beim Zugriff eines Benutzers auf eine IBM MQ-Ressource die entsprechende Benutzer-ID (oder Benutzer-IDs), um festzustellen, ob der Zugriff auf diese Ressource berechtigt ist. Durch Definition eines RESLEVEL-Profils können Sie festlegen, ob keine Benutzer-ID, eine Benutzer-ID oder, falls zutreffend, zwei Benutzer-IDs überprüft werden.

Diese Einstellungen werden für jede Verbindung einzeln festgelegt und gelten nur für die Dauer der jeweiligen Verbindung.

Für jeden Warteschlangenmanager kann jeweils nur ein RESLEVEL-Profil festgelegt werden. Die Einstellung wird durch den Zugriff implementiert, den eine Benutzer-ID auf dieses Profil hat.

## IBM MQ RACF-Klassen mit Groß- und Kleinschreibung oder mit Großschreibung

Sie können jetzt die Unterstützung für RACF-Profile mit gemischter Groß-/Kleinschreibung nutzen, d. h. Sie können Ressourcennamen in gemischter Groß-/Kleinschreibung verwenden und IBM MQ RACF-Profile zu deren Schutz definieren.

Dabei können Sie zwischen folgenden Möglichkeiten wählen:

- Weiterhin nur IBM MQ RACF-Klassen mit reiner Großschreibung verwenden, wie in früheren Releases, oder
- IBM MQ RACF-Klassen mit neuer gemischter Groß-/Kleinschreibung verwenden.

Auch wenn Sie keine RACF-Profile mit gemischter Groß-/Kleinschreibung verwenden, können Sie dennoch Ressourcennamen mit gemischter Groß-/Kleinschreibung in IBM MQ for z/OS verwenden, wobei diese Ressourcennamen jedoch nur durch generische RACF-Profile in den IBM MQ-Klassen mit reiner Großschreibung geschützt werden können. Wenn Sie die Unterstützung für IBM MQ RACF-Profile mit gemischter Groß-/Kleinschreibung nutzen, können Sie differenziertere Schutzstufen bereitstellen, indem Sie IBM MQ RACF-Profile in den IBM MQ-Klassen mit gemischter Groß-/Kleinschreibung definieren.

## z/OS Schützbare Ressourcen in IBM MQ for z/OS

Beim Start eines Warteschlangenmanagers oder per Anweisung durch einen Bedienerbefehl bestimmt IBM MQ for z/OS, welche Ressourcen Sie schützen möchten.

Sie können steuern, welche Sicherheitsprüfungen für jeden einzelnen Warteschlangenmanager durchgeführt werden. Sie können beispielsweise eine Reihe von Sicherheitsprüfungen für einen Produktionswarteschlangenmanager, aber nicht für einen Testwarteschlangenmanager festlegen.

### Verbindungssicherheit

Die Verbindungssicherheit wird geprüft, wenn ein Anwendungsprogramm versucht, eine Verbindung zu einem Warteschlangenmanager herzustellen. Dazu wird entweder eine MQCONN- oder MQCONNX-Anforderung ausgegeben, die Prüfung wird aber auch durchgeführt, wenn der Kanalinitiator, der CICS- oder der IMS-Adapter eine Verbindungsanforderung ausgibt.

Wenn Sie Sicherheit auf Warteschlangenmanagerebene verwenden, können Sie die Überprüfung der Verbindungssicherheit für bestimmte Warteschlangenmanager inaktivieren. In diesem Fall kann jedoch jeder Benutzer eine Verbindung zu diesem Warteschlangenmanager herstellen.

Beim CICS-Adapter wird nur die Benutzer-ID des CICS-Adressraums zum Prüfen der Verbindungssicherheit verwendet, nicht die Benutzer-ID der einzelnen CICS-Terminals. Wenn beim IMS -Adapter die IMS -Steuerregionen oder abhängige Regionen eine Verbindung zu IBM MQherstellen, wird die Benutzer-ID des IMS -Adressraums überprüft. Beim Kanalinitiator wird die von seinem Adressraum verwendete Benutzer-ID geprüft.

Die Prüfung der Verbindungssicherheit können Sie entweder für Warteschlangenmanager oder für Gruppen mit gemeinsamer Warteschlange aktivieren bzw. inaktivieren.

### Befehlssicherheit

Die Befehlssicherheitsprüfung wird ausgeführt, wenn ein Benutzer einen MQSC-Befehl aus einer der im Abschnitt Quellen beschriebenen Quellen ausgibt, aus denen Sie MQSC-und PCF-Befehle unter IBM MQ for z/OSausgeben können. Sie können die vom Befehl angegebene Ressource wie unter „Befehlsressourcensicherheit” auf Seite 302 beschrieben separat prüfen.

Wenn Sie die Befehlsüberprüfung inaktivieren, wird nicht geprüft, ob Befehlsausgeber für die Ausgabe von Befehlen berechtigt sind.

Wenn WebSphere MQ-Scriptbefehle über eine Konsole eingegeben werden, muss diese das z/OS-Konsolenberechtigungsattribut SYS haben. Befehle, die über den Dataset CSQINP1 oder CSQINP2 bzw. intern vom Warteschlangenmanager ausgegeben werden, sind von allen Sicherheitsprüfungen ausgeschlossen, während die Befehle für CSQINPX die Benutzer-ID des Adressraums des Kanalinitiators verwenden. Über

den normalen Datasetschutz müssen Sie kontrollieren, wer zur Aktualisierung dieser Datasets berechtigt ist.

Die Prüfung der Befehlssicherheitsprüfung können Sie entweder für Warteschlangenmanager oder für Gruppen mit gemeinsamer Warteschlange aktivieren bzw. inaktivieren.

## Befehlsressourcensicherheit

Bei einigen WebSphere MQ-Scriptbefehlen, z. B. zum Definieren einer lokalen Warteschlange, werden die IBM MQ-Ressourcen bearbeitet. Bei aktivierter Sicherheit der Befehlsressourcen prüft IBM MQ bei jeder Ausgabe eines Befehls, bei dem Ressourcen bearbeitet werden, ob der Benutzer berechtigt ist, die Definition der betreffenden Ressource zu ändern.

Mithilfe der Sicherheit der Befehlsressourcen können Sie die Umsetzung von Benennungsstandards unterstützen. Ein Gehaltsadministrator ist z. B. unter Umständen berechtigt, nur Warteschlangen, deren Namen mit "GEHALT" beginnen, zu löschen und zu definieren. Bei aktivierter Sicherheit der Befehlsressourcen werden keine Sicherheitsprüfungen für die Ressource durchgeführt, die vom Befehl bearbeitet wird. Verwechseln Sie die Sicherheit der Befehlsressourcen nicht mit der Befehlssicherheit - sie hängen nicht voneinander ab.

Wenn Sie die Sicherheit der Befehlsressourcen inaktivieren, hat dies keine Auswirkung auf die Ressourcenprüfung, die speziell für andere Arten der Verarbeitung ausgeführt wird, die ohne Befehle stattfinden.

Die Prüfung der Befehlsressourcensicherheit können Sie entweder für Warteschlangenmanager oder für Gruppen mit gemeinsamer Warteschlange aktivieren bzw. inaktivieren.

## Kanalsicherheitsaspekte

### Kanalsicherheit

Wenn Sie Kanäle verwenden, richten sich die verfügbaren Sicherheitskomponenten nach dem Kommunikationsprotokoll, das Sie verwenden möchten. Wenn Sie TCP verwenden, stellt das Kommunikationsprotokoll keine Sicherheitskomponenten zur Verfügung, obwohl Sie TLS verwenden können. Wenn Sie APPC verwenden, können Sie Benutzer-ID-Informationen vom sendenden Nachrichtenkanalagent über das Netz zur Überprüfung an den Ziel-Nachrichtenkanalagenten senden.

Für beide Protokolle können Sie angeben, welche und wie viele Benutzer-IDs aus Sicherheitsgründen geprüft werden sollen. Die verfügbaren Auswahlmöglichkeiten richten sich erneut nach dem Protokoll, das Sie verwenden, nach Ihren Angaben beim Definieren des Kanals und nach den RESLEVEL-Einstellungen für den Kanalinitiator.

Weitere Informationen zu den verfügbaren Arten der Kanalsicherheit finden Sie unter Kanalauthentifizierungsdatensätze und Sicherheitsexits - Übersicht

**Zugehörige Verweise**

„API-Ressourcensicherheit in IBM MQ for z/OS” auf Seite 302
Ressourcen werden überprüft, wenn eine Anwendung ein Objekt mit dem Aufruf MQOPEN oder MQPUT1 öffnet. Welche Zugriffsberechtigung zum Öffnen eines Objekts erforderlich ist, hängt von den Öffnungsoptionen ab, die beim Öffnen der Warteschlange angegeben werden.

### z/OS *API-Ressourcensicherheit in IBM MQ for z/OS*

Ressourcen werden überprüft, wenn eine Anwendung ein Objekt mit dem Aufruf MQOPEN oder MQPUT1 öffnet. Welche Zugriffsberechtigung zum Öffnen eines Objekts erforderlich ist, hängt von den Öffnungsoptionen ab, die beim Öffnen der Warteschlange angegeben werden.

Die Sicherheit der API-Ressourcen ist in folgende Sicherheitsprüfungen unterteilt:

- Queue
- Prozess
- Namensliste

- Alternativer Benutzer
- Kontext

Beim Öffnen des Warteschlangenmanager-Objekts oder beim Zugriff auf Speicherklassenobjekte werden keine Sicherheitsprüfungen ausgeführt.

**Warteschlange**

Anhand der Sicherheitsprüfung für Warteschlangen wird gesteuert, welche Benutzer zum Öffnen welcher Warteschlange berechtigt sind und welche Optionen ihnen dabei zur Verfügung stehen. Ein Benutzer kann beispielsweise berechtigt sein, die Warteschlange 'PAYROLL.INCREASE.SALARY' zu öffnen, um die Nachrichten in dieser Warteschlange zu durchsuchen (mit der Option 'MQOO_BROWSE'), jedoch nicht, um Nachrichten aus der Warteschlange zu entfernen (mithilfe einer der Optionen vom Typ 'MQOO_INPUT_*'). Wenn Sie die Sicherheitsprüfung für Warteschlangen inaktivieren, kann jeder Benutzer jede Warteschlange mit einer beliebigen gültigen Option öffnen (d. h. mit einer gültigen Option vom Typ 'MQOO_*' im Aufruf MQOPEN oder MQPUT1).

Die Prüfung der Warteschlangensicherheit können Sie entweder für Warteschlangenmanager oder für Gruppen mit gemeinsamer Warteschlange aktivieren bzw. inaktivieren.

**Prozess**

Die Sicherheitsprüfung für Prozesse wird ausgeführt, wenn ein Benutzer ein Prozessdefinitionsobjekt öffnet. Wenn Sie die Sicherheitsprüfung für Prozesse inaktivieren, kann jeder Benutzer jeden Prozess öffnen.

Die Prüfung der Prozesssicherheit können Sie entweder für Warteschlangenmanager oder für Gruppen mit gemeinsamer Warteschlange aktivieren bzw. inaktivieren.

**Namensliste**

Die Sicherheitsprüfung für Namenslisten wird ausgeführt, wenn ein Benutzer eine Namensliste öffnet. Wenn Sie die Sicherheitsprüfung für Namenslisten inaktivieren, kann jeder Benutzer jede Namensliste öffnen.

Die Prüfung der Namenslistensicherheit können Sie entweder für Warteschlangenmanager oder für Gruppen mit gemeinsamer Warteschlange aktivieren bzw. inaktivieren.

**Alternativbenutzer**

Anhand der Sicherheitsprüfung für alternative Benutzer-IDs wird gesteuert, ob eine Benutzer-ID die Berechtigung einer anderen Benutzer-ID zum Öffnen eines IBM MQ-Objekts verwenden darf.

For example:

- Ein Serverprogramm, das unter der Benutzer-ID 'PAYSERV' ausgeführt wird, ruft aus einer Warteschlange eine Anforderungsnachricht ab, die von der Benutzer-ID 'USER1' in diese Warteschlange eingereiht wurde.
- Wenn das Serverprogramm die Anforderungsnachricht abruft, verarbeitet es die Anforderung und versetzt die Antwort zurück in die Warteschlange für Antwortnachrichten, die mit der Anforderungsnachricht angegeben ist.
- Anstatt der eigenen Benutzer-ID (PAYSERV) die Berechtigung zum Öffnen der Warteschlange für zu beantwortende Nachrichten zu erteilen, kann der Server auch eine andere Benutzer-ID, in diesem Fall 'USER1' dafür angeben. In diesem Beispiel wird anhand der Sicherheitsprüfung für alternative Benutzer-IDs überprüft, ob die Benutzer-ID 'PAYSERV' dazu berechtigt ist, die Benutzer-ID 'USER1' als alternative Benutzer-ID zum Öffnen der Warteschlange für zu beantwortende Nachrichten anzugeben.

Die alternative Benutzer-ID wird im Feld *AlternateUserId* des Objektdeskriptors (MQOD) angegeben.

Alternative Benutzer-IDs können für alle IBM MQ-Objekte verwendet werden, z. B. für Prozesse oder Namenslisten. Dies hat keinen Einfluss auf die Benutzer-ID, die von anderen Ressourcenmanagern verwendet wird, z. B. für die CICS-Sicherheit oder den z/OS-Dateischutz.

Wenn die Sicherheitsprüfung für alternative Benutzer-IDs inaktiviert ist, kann jeder Benutzer jede beliebige andere Benutzer-ID als alternative Benutzer-ID verwenden.

Die Prüfung der Sicherheit der alternativen Benutzer-IDs können Sie entweder für Warteschlangenmanager oder für Gruppen mit gemeinsamer Warteschlange aktivieren bzw. inaktivieren.

**Context**

Als Kontext werden Informationen bezeichnet, die sich auf eine bestimmte Nachricht beziehen und im Nachrichtendeskriptor (MQMD) befinden, der Teil der Nachricht ist. Die Kontextinformationen werden in zwei Abschnitten geliefert:

**Identitätsabschnitt**
Gibt den Benutzer der Anwendung an, von der die Nachricht zuerst in die Warteschlange eingereiht wurde. Der Abschnitt besteht aus folgenden Feldern:

- *UserIdentifier*
- *AccountingToken*
- *ApplIdentityData*

**Ursprungsabschnitt**
Gibt die Anwendung an, von der die Nachricht in die Warteschlange eingereiht wurde, in der sie aktuell gespeichert ist. Der Abschnitt besteht aus folgenden Feldern:

- *PutApplType*
- *PutApplName*
- *PutDate*
- *PutTime*
- *ApplOriginData*

Die Kontextdaten können von Anwendungen angegeben werden, wenn diese einen MQPUT- oder MQPUT1-Aufruf ausgeben. Es ist möglich, dass die Daten von der Anwendung generiert wurden, von einer anderen Nachricht weitergegeben wurden oder vom Warteschlangenmanager standardmäßig generiert wurden. Anhand der Kontextdaten können Serverprogramme beispielsweise die Identität des anfordernden Benutzers überprüfen und so feststellen, ob die Nachricht von der richtigen Anwendung gesendet wurde. Normalerweise wird im Feld *UserIdentifier* die Benutzer-ID eines alternativen Benutzers angegeben.

Anhand der Sicherheitsprüfung für Kontext wird gesteuert, ob der Benutzer im Aufruf MQOPEN oder MQPUT Kontextoptionen angeben kann. Weitere Informationen zu Kontextoptionen finden Sie unter MQOPEN-Optionen in Bezug auf den Nachrichtenkontext. Beschreibungen der Nachrichtendeskriptorfelder in Bezug auf den Kontext finden Sie unter MQMD – Nachrichtendeskriptor.

Wenn Sie die Sicherheitsprüfung für Kontext inaktivieren, kann jeder Benutzer alle im Rahmen der Warteschlangensicherheit zulässigen Kontextoptionen verwenden.

Die Prüfung der Kontextsicherheit können Sie entweder für Warteschlangen, für Warteschlangenmanager oder für Gruppen mit gemeinsamer Warteschlange aktivieren bzw. inaktivieren.

## z/OS Verfügbarkeit unter z/OS

IBM MQ for z/OS verfügt über viele Funktionen für Hochverfügbarkeit. In diesem Abschnitt werden einige Überlegungen zur Verfügbarkeit erläutert.

IBM MQ verfügt über verschiedene Funktionen, mit denen die Systemverfügbarkeit erhöht werden kann, falls der Warteschlangenmanager oder Kanalinitiator fehlschlägt. Weitere Informationen zu diesen Funktionen finden Sie in folgenden Abschnitten:

- Überlegungen zum Sysplex

- Gemeinsam genutzte Warteschlangen
- Gemeinsam genutzte Kanäle
- IBM MQ-Netzverfügbarkeit
- z/OS Automatic Restart Manager (ARM) verwenden
- z/OS Extended Recovery Facility (XRF) verwenden
- z/OS-Attribut GROUPUR zur Wiederherstellung in einer Gruppe mit gemeinsamer Warteschlange verwenden
- Weitere Informationen zur Verfügbarkeit

## Überlegungen zum Sysplex

In einem *Sysplex* arbeiten mehrere z/OS-Betriebssystemimages in einem einzelnen Systemimage zusammen und kommunizieren dabei mithilfe einer Coupling-Facility. IBM MQ kann die Funktionen der Sysplex-Umgebung zur Erhöhung der Verfügbarkeit verwenden.

Das Löschen der Affinitäten zwischen einem Warteschlangenmanager und einem bestimmten z/OS-Image ermöglicht den Neustart des Warteschlangenmanagers in einem anderen z/OS-Image, falls das ursprüngliche Image fehlschlägt. Der Neustart kann manuell, mithilfe von Automatic Restart Manager (ARM) oder mithilfe der Systemautomatisierung ausgeführt werden, wenn Sie Folgendes sicherstellen:

- Alle Seitengruppen, Protokolle, Bootstrap-Datasets, Code-Bibliotheken und Warteschlangenmanager-Konfigurationsdateien sind auf gemeinsam genutzten Datenträgern definiert.
- Die Subsystemdefinition gilt für das gesamte Sysplex und hat innerhalb des Sysplex einen eindeutigen Namen.
- Die Version des *Vorabcodes* , der auf jedem z/OS -Image zum Zeitpunkt des einleitenden Programmladens installiert ist, hat dieselbe Version.
- Virtuelle TCP/IP-Adressen (VIPA) sind in jedem TCP-Stapel im Sysplex verfügbar und Sie haben die TCP-Empfangsprogramme für IBM MQ sowie die eingehenden Verbindungen so konfiguriert, dass sie VIPAs anstatt der standardmäßigen Hostnamen verwenden.

Weitere Informationen zur Verwendung von TCP in einem Sysplex finden Sie im Abschnitt *TCP/IP in einem Sysplex*, SG24-5235, einer IBM Redbooks -Veröffentlichung.

Darüber hinaus können Sie in einem Sysplex mehrere Warteschlangenmanager, die in unterschiedlichen Betriebssystemimages ausgeführt werden, als Gruppe mit gemeinsamer Warteschlange konfigurieren, damit sie die Vorteile von gemeinsam genutzten Warteschlangen und Kanälen zur Erhöhung der Verfügbarkeit und zum Lastausgleich nutzen können.

## Gemeinsam genutzte Warteschlangen

In einer Umgebung, in der Gruppen mit gemeinsamer Warteschlange unterstützt werden, kann eine Anwendung eine Verbindung zu einem beliebigen Warteschlangenmanager herstellen, der zu einer Gruppe mit gemeinsamer Warteschlange gehört. Da alle Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange auf dieselben gemeinsam genutzten Warteschlangen Zugriff haben, ist die Anwendung nicht auf die Verfügbarkeit eines bestimmten Warteschlangenmanagers angewiesen, denn jeder dieser Warteschlangenmanager kann jede Warteschlange bedienen. Dies führt zu einer höheren Verfügbarkeit, weil im Falle eines gestoppten Warteschlangenmanagers alle anderen Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange die Verarbeitung der Warteschlange fortsetzen können. Weitere Informationen zur hohen Verfügbarkeit von gemeinsam genutzten Warteschlangen finden Sie unter „Vorteile von gemeinsam genutzten Warteschlangen” auf Seite 204.

Um die Verfügbarkeit von Nachrichten in einer Gruppe mit gemeinsamer Warteschlange weiter zu erhöhen, ermittelt IBM MQ, ob ein anderer Warteschlangenmanager in der Gruppe seine Verbindung zur Coupling-Facility abnormal beendet hat, und schließt die für diesen Warteschlangenmanager noch an-

stehenden Arbeitseinheiten ab, falls möglich. Dies wird als *Peerwiederherstellung* bezeichnet und unter „Peerwiederherstellung” auf Seite 290 erläutert.

Für Arbeitseinheiten, die zum Zeitpunkt des Fehlers einen unbestätigten Status hatten, ist keine Peerwiederherstellung möglich. Sie können mit Automatic Restart Manager (ARM) alle vom Ausfall betroffenen Systeme (z. B. CICS, Db2 und IBM MQ) neu starten und gleichzeitig sicherstellen, dass sie alle auf demselben neuen Prozessor gestartet werden. Dies bedeutet, dass die Systeme resynchronisiert werden können, und ermöglicht eine rasche Wiederherstellung von unbestätigten Arbeitseinheiten. Weitere Informationen hierzu finden Sie unter „z/OS Automatic Restart Manager (ARM) verwenden” auf Seite 306.

## Gemeinsam genutzte Kanäle

In einer Umgebung, in der Gruppen mit gemeinsamer Warteschlange unterstützt werden, stellt IBM MQ Funktionen zur Erhöhung der Netzverfügbarkeit bereit. Der Kanalinitiator ermöglicht die Verwendung von Produkten für den Netzbetrieb (z. B. VTAM Generic Ressources), die einen Lastausgleich der Netzanforderungen in einer Gruppe zulässiger Server ausführen und Serverfehler im Netz verbergen. IBM MQ verwendet einen generischen Port für eingehende Anforderungen, damit Verbindungsanforderungen an einen beliebigen verfügbaren Kanalinitiator in der Gruppe mit gemeinsamer Warteschlange weitergeleitet werden können. Weitere Informationen hierzu finden Sie unter „Gemeinsam genutzte Kanäle” auf Seite 226.

Gemeinsam genutzte ausgehende Kanäle rufen die Nachrichten, die sie senden, aus einer gemeinsam genutzten Übertragungswarteschlange ab. Informationen zum Status eines gemeinsam genutzten Kanals wird für die gesamte Gruppe mit gemeinsamer Warteschlange an einer einzelnen Position gespeichert. Dies bedeutet, dass ein Kanal automatisch auf einem anderen Kanalinitiator in der Gruppe mit gemeinsamer Warteschlange neu gestartet werden kann, falls der Kanalinitiator, der Warteschlangenmanager oder das Kommunikationssubsystem fehlschlägt. Dies wird als *Peer-Kanalwiederherstellung* bezeichnet und unter Gemeinsam genutzte ausgehende Kanäle erläutert.

## Netzverfügbarkeit von IBM MQ

IBM MQ-Nachrichten werden über Kanäle in einem IBM MQ-Netz von einem Warteschlangenmanager zum nächsten übertragen. Sie können die Konfiguration auf mehreren Ebenen ändern, um die Netzverfügbarkeit eines Warteschlangenmanagers zu erhöhen und die Möglichkeit eines IBM MQ-Kanals, einen Netzfehler zu erkennen und die Verbindung wiederherzustellen, zu verbessern.

Für TCP/IP-Kanäle ist die TCP-*Keepalive*-Funktion verfügbar. Sie sorgt dafür, dass TCP regelmäßig zwischen Sitzungen Pakete sendet, um Netzfehler zu erkennen. Mit dem Kanalattribut 'KAINT' wird die Frequenz der Pakete für einen Kanal festgelegt.

Mit *AdoptMCA* kann ein Kanal, der aufgrund eines Netzausfalls in der Empfangsverarbeitung blockiert ist, beendet und durch eine neue Verbindungsanforderung ersetzt werden. Sie können 'AdoptMCA' mithilfe der Warteschlangenmanager-Eigenschaft 'ADOPTMCA' über das Dienstprogramm 'MQSC' bzw. der Eigenschaft 'AdoptNewMCAType' über die Programmable Command Formats-Schnittstelle steuern.

Mit *ReceiveTimeout* wird verhindert, dass ein Kanal dauerhaft durch einen Netzempfangsaufruf blockiert wird. Die Kanalinitiatorparameter 'RCVTIME' und 'RCVTMIN' bestimmen die Eigenschaften des Empfangszeitlimits für Kanäle, und zwar als Funktion ihres Intervalls der Überwachungssignale. Weitere Informationen finden Sie unter Warteschlangenmanager-Parameter.

## z/OS Automatic Restart Manager (ARM) verwenden

Sie können IBM MQ for z/OS in Verbindung mit z/OS Automatic Restart Manager (ARM) verwenden. Wenn ein Warteschlangenmanager oder Kanalinitiator fehlgeschlagen ist, wird er von ARM in demselben z/OS-Image neu gestartet. Wenn z/OS fehlschlägt, schlägt eine ganze Gruppe zugehöriger Subsysteme und Anwendungen ebenfalls fehl. ARM kann alle fehlgeschlagenen Systeme automatisch und in vordefinierter

Reihenfolge in einem anderen z/OS-Image innerhalb des Sysplex neu starten. Dies wird als systemübergreifender Neustart bezeichnet.

ARM ermöglicht in einer Umgebung mit Unterstützung von Gruppen mit gemeinsamer Warteschlange die schnelle Wiederherstellung von unbestätigten Transaktionen. Mit dieser Funktion wird die Verfügbarkeit auch dann erhöht, wenn Sie keine Gruppen mit gemeinsamer Warteschlange verwenden.

Mit ARM können Sie einen Warteschlangenmanager in einem anderen z/OS-Image innerhalb des Sysplex neu starten, falls z/OS fehlschlägt.

Führen Sie folgende Schritte aus, um den automatischen Neustart zu aktivieren:

1. Konfigurieren Sie eine ARM-Coupling-Datei.
2. Definieren Sie die Aktionen für automatischen Neustart, die z/OS in einer *ARM-Richtlinie*ausführen soll.
3. Starten Sie die ARM-Richtlinie.

Wenn der Neustart von Warteschlangenmanagern automatisch in anderen z/OS-Images ausgeführt werden soll, muss für jeden Warteschlangenmanager in jedem z/OS-Image, in dem der jeweilige Warteschlangenmanager neu gestartet werden kann, ein Subsystemname mit 4 Zeichen definiert werden, der im gesamten Sysplex eindeutig ist.

Informationen zur Verwendung von ARM mit IBM MQ finden Sie im Abschnitt ARM in einem IBM MQ-Netz verwenden.

## z/OS Extended Recovery Facility (XRF) verwenden

Sie können IBM MQ in einer Umgebung mit erweiterter Wiederherstellungsfunktion (Extended Recovery Facility, XRF) ausführen. Alle Dateien, deren Besitzer IBM MQ ist (ausführbarer Code, Bootstrap-Datasets, Protokolle und Seitengruppen), müssen sich in einer DASD-Einheit befinden, die vom aktiven Prozessor und den alternativen Prozessoren für die erweiterte Wiederherstellungsfunktion (XRF-Prozessoren) gemeinsam genutzt wird.

Wenn Sie die erweiterte Wiederherstellungsfunktion verwenden, müssen Sie den Warteschlangenmanager auf dem aktiven Prozessor stoppen und auf dem alternativen Prozessor neu starten. Für CICS kann dies mithilfe der in CICS bereitgestellten Befehlslistentabelle oder manuell durch den Systembediener ausgeführt werden. Für IMS muss die Operation manuell und zwar nach Abschluss der Prozessorumschaltung durch das koordinierende IMS-System ausgeführt werden.

Der Warteschlangenmanager kann erst auf den alternativen Prozessor umgeschaltet werden, wenn alle IBM MQ-Dienstprogramm abgeschlossen oder beendet wurden. Bei der Planung des Ablaufs der erweiterte Wiederherstellungsfunktion sollten Sie die Auswirkungen dieser potenziellen Unterbrechung sorgfältig berücksichtigen.

Stellen Sie sicher, dass der Warteschlangenmanager auf dem alternativen Prozessor nicht gestartet wird, bevor der Warteschlangenmanager auf dem aktiven Prozessor beendet wurde. Ein vorzeitiger Start kann zu schwerwiegenden Integritätsproblemen für die Daten, den Katalog und das Protokoll führen. Wenn Sie Global Resource Serialization (GRS) verwenden, können Integritätsprobleme vermieden werden, indem die gleichzeitige Verwendung von IBM MQ auf den beiden System verhindert wird. Dazu müssen Sie das Bootstrap-Dataset als geschützte Ressource und den aktiven Prozessor und die alternativen Prozessoren für die erweiterte Wiederherstellungsfunktion in den GRS-Ring einfügen.

## z/OS-Attribut GROUPUR zur Wiederherstellung in einer Gruppe mit gemeinsamer Warteschlange verwenden

Gruppen mit gemeinsamer Warteschlange ermöglichen zusätzliche transaktionsorientierte Funktionen, die in diesem Abschnitt beschrieben werden. Mit dem Attribut 'GROUPUR' können für XA-Clientanwendungen alle erforderlichen Wiederherstellungsoperationen für unbestätigte Transaktionen auf jedem beliebigen Mitglied der Gruppe mit gemeinsamer Warteschlange ausgeführt werden.

Wenn eine XA-Clientanwendung über ein Sysplex eine Verbindung zu einer Gruppe mit gemeinsamer Warteschlange herstellt, lässt sich im Vorhinein nicht bestimmen, mit welchem Warteschlangenmanager die Anwendung verbunden wird. Wenn die Warteschlangenmanager innerhalb der Gruppe mit gemeinsamer Warteschlange das Attribut 'GROUPUR' verwenden, können alle eventuell erforderlichen Wiederherstellungsoperationen für unbestätigte Transaktionen auf einem beliebigen Mitglied der Gruppe mit gemeinsamer Warteschlange ausgeführt werden. Selbst wenn der Warteschlangenmanager, mit dem Anwendung anfänglich verbunden wurde, nicht verfügbar ist, kann die Transaktionswiederherstellung dennoch ausgeführt werden.

Auf diese Weise besteht für die XA-Clientanwendung keine Abhängigkeit von der Verfügbarkeit bestimmter Mitglieder der Gruppe mit gemeinsamer Warteschlange, und die Verfügbarkeit des Warteschlangenmanagers wird erhöht. Die Gruppe mit gemeinsamer Warteschlange wird für die Transaktionsanwendung als einzelne Entität dargestellt, die alle IBM MQ-Funktionen bereitstellt, ohne einen einzelnen Warteschlangenmanager als Fehlerquelle abzubilden.

Diese Funktionalität ist für die Transaktionsanwendung nicht erkennbar.

### Weitere Informationen zur Verfügbarkeit

Weitere Informationen zu diesen Themen finden Sie in folgenden Quellen:

*Tabelle 24. Weitere Informationen zur Verfügbarkeit*

| **Thema** | **Quelle** |
|---|---|
| Gruppen mit gemeinsamer Warteschlange | „Gemeinsam genutzte Warteschlangen und Gruppen mit gemeinsamer Warteschlange” auf Seite 181 |
| Systemparameter | Systemparameter konfigurieren |
| Dienstprogramme des Automatic Restart Dienstprogramme für die | ARM in einem IBM MQ-Netz verwenden |
| MQSC-Befehle | MQSC-Befehle |

## z/OS Überwachung und Statistik in IBM MQ for z/OS

IBM MQ for z/OS verfügt über eine Reihe von Funktionen zur Überwachung des Warteschlangenmanagers und zur Erfassung von statistischen Daten.

IBM MQ stellt Funktionen zur Überwachung des System und Erfassung von statistischen Daten bereit. Weitere Informationen zu diesen Funktionen finden Sie in folgenden Abschnitten:

- „Onlineüberwachung” auf Seite 308
- „IBM MQ-Trace” auf Seite 309
- „Ereignisse” auf Seite 309

### Onlineüberwachung

IBM MQ stellte folgende Befehle zur Überwachung des Status von IBM MQ-Objekten bereit:

- 'DISPLAY CHSTATUS' zeigt den Status des angegebenen Kanals an.
- 'DISPLAY QSTATUS' zeigt den Status der angegebenen Warteschlange an.
- 'DISPLAY CONN' zeigt den Status der angegebenen Verbindung an.

Weitere Informationen zu diesen Befehlen finden Sie unter WebSphere MQ-Scriptbefehle.

## IBM MQ-Trace

IBM MQ verfügt über eine Tracefunktion, mit der Sie folgende Informationen erfassen können, solange der Warteschlangenmanager aktiv ist:

**Leistungsstatistik**
Der Statistiktrace erfasst folgende Informationen, die Sie bei der Leistungsüberwachung und Systemoptimierung unterstützen:

- Anzahl der unterschiedlichen MQI-Anforderungen (Warteschlangenmanager-Statistik)
- Anzahl der unterschiedlichen Objektanforderungen (Datenmanagerstatistik)
- Informationen zur Db2-Verwendung (Db2-Managerstatistik)
- Informationen zur Coupling-Facility-Verwendung (Coupling-Facility-Managerstatistik)
- Informationen zur Nutzung von gemeinsam genutzten Nachrichtendateien (SMDS-Statistik, Shared Message Data Set)
- Informationen zur Pufferpoolverwendung (Puffermanagerstatistik)
- Informationen zur Protokollierung (Protokollmanagerstatistik)
- Informationen zur Speicherverwendung (Speichermanagerstatistik)
- Informationen zu Sperrenanforderungen (Sperrenmanagerstatistik)

**Abrechnungsdaten**

- Der Abrechnungstrace erfasst Informationen zur Prozessorzeit, die für die Verarbeitung von MQI-Aufrufen benötigt wird, und zur Anzahl der MQPUT- und MQGET-Anforderungen, die von einem bestimmten Benutzer gestellt werden.
- IBM MQ kann auch Informationen zu jeder von IBM MQ ausgeführten Task erfassen. Diese Daten werden als Abrechnungsdatensatz auf Threadebene erfasst. Außerdem erfasst IBM MQ für jeden Thread Informationen über alle von ihm verwendeten Warteschlangen.

Die vom Trace generierten Daten werden an die Systemverwaltungsfunktion (System Management Facility, SMF) oder die allgemeine Tracefunktion (Generalized Trace Facility, GTF) gesendet.

## Ereignisse

IBM MQ-Ereignisse stellen Informationen zu Fehlern, Warnungen und anderen erheblichen Vorkommnissen auf einem Warteschlangenmanager bereit. Durch Einbindung dieser Ereignisse in Ihre eigene Systemverwaltungsanwendung können Sie die Aktivitäten mehrerer IBM MQ-Anwendungen auf vielen Warteschlangenmanagern überwachen. Insbesondere können Sie die Überwachung aller Warteschlangenmanager in Ihrem System von einem einzelnen Warteschlangenmanager aus durchführen.

Ereignisse können mithilfe eines benutzerdefinierten Berichtserstellungsmechanismus an eine Verwaltungsanwendung übermittelt werden, mit der die Darstellung der Ereignisse für einen Bediener möglich ist. Aufgrund von Ereignissen können Anwendungen auch als Agenten für andere Verwaltungsnetze, z. B. NetView, agieren, indem sie Berichte überwachen und die entsprechenden Alerts erstellen.

**Zugehörige Tasks**
IBM MQ-Trace verwenden
IBM MQ-Ereignisse verwenden

# z/OS Disposition einer Arbeitseinheit mit Wiederherstellung unter z/OS

Bestimmte Transaktionsanwendungen können, wenn sie mit einem Warteschlangenmanager in einer Gruppe mit gemeinsamer Warteschlange (Queue Sharing Group; QSG) verbunden sind, eine Disposition 'GROUP' statt einer Disposition 'QMGR' der Arbeitseinheit mit Wiederherstellung verwenden, indem bei der Verbindungsherstellung der QSG-Name statt des Warteschlangenmanagernamens angegeben wird.

Damit wird die Wiederherstellung von Transaktionen flexibler und leistungsfähiger, da die Anforderung wegfällt, die Verbindung mit demselben Warteschlangenmanager in der QSG wiederherzustellen.

Transaktionen, die von Anwendungen gestartet werden, die eine Verbindung unter Verwendung des Namens der Gruppe mit gemeinsamer Warteschlange hergestellt haben, verfügen auch über eine Disposition 'GROUP' der Arbeitseinheit mit Wiederherstellung.

Wenn eine Transaktionsanwendung eine Verbindung zu einer Disposition 'GROUP' der Arbeitseinheit mit Wiederherstellung herstellt, ist sie logisch mit der Gruppe mit gemeinsamer Warteschlange verbunden und hat keine Affinität zu einem bestimmten Warteschlangenmanager. Alle Commit-Transaktionen mit zwei Phasen, die gestartet wurden und Phase 1 des Commit-Prozesses abgeschlossen haben, d. h., die einen unbestätigten Status aufweisen, können abgefragt und aufgelöst werden, wenn sie mit einem Warteschlangenmanager in der QSG verbunden werden. In einem Wiederherstellungsszenario bedeutet dies, dass der Transaktionskoordinator nicht die Verbindung mit demselben Warteschlangenmanager wiederherstellen muss, der zu diesem Zeitpunkt möglicherweise nicht verfügbar ist.

Anwendungen, die eine Verbindung zu einer Disposition 'QMGR' der Arbeitseinheit mit Wiederherstellung herstellen, haben eine direkte Affinität zu dem Warteschlangenmanager, mit dem sie verbunden sind. In einem Wiederherstellungsszenario muss der Transaktionskoordinator die Verbindung mit demselben Warteschlangenmanager wiederherstellen, um unbestätigte Transaktionen aufzulösen. Dies gilt unabhängig davon, ob der Warteschlangenmanager zu einer Gruppe mit gemeinsamer Warteschlange gehört oder nicht.

Wenn Anwendungen einen Namen einer Gruppe mit gemeinsamer Warteschlange angeben und damit eine Verbindung zu einem Warteschlangenmanager in einer Gruppe mit gemeinsamer Warteschlange mit einer Disposition 'GROUP' der Arbeitseinheit mit Wiederherstellung herstellen, ist die Gruppe mit gemeinsamer Warteschlange logisch ein eigenständiger Ressourcenmanager. Das heißt, dass unbestätigte Transaktionen für eine Anwendung nur dann sichtbar sind, wenn diese mit derselben Disposition der Arbeitseinheit mit Wiederherstellung die Verbindung wiederherstellt. Unbestätigte Transaktionen mit einer Disposition 'QMGR' der Arbeitseinheit mit Wiederherstellung sind für Anwendungen, die eine Verbindung zu einer Disposition 'GROUP' der Arbeitseinheit mit Wiederherstellung hergestellt haben, nicht sichtbar und umgekehrt.

**Zugehörige Konzepte**

„GROUP-Arbeitseinheiten mit Wiederherstellung aktivieren” auf Seite 310
Eine Gruppe mit gemeinsamer Warteschlange kann Unterstützung für GROUP-Arbeitseinheiten mit Wiederherstellung konfigurieren und aktivieren.

„Anwendungsunterstützung” auf Seite 311
Verwenden Sie diese Seite, um zu bestimmen, welche Anwendungen eine Verbindung zu einer Disposition 'GROUP' der Arbeitseinheit mit Wiederherstellung herstellen können.

## z/OS GROUP-Arbeitseinheiten mit Wiederherstellung aktivieren

Eine Gruppe mit gemeinsamer Warteschlange kann Unterstützung für GROUP-Arbeitseinheiten mit Wiederherstellung konfigurieren und aktivieren.

Wenn Sie auf einem Warteschlangenmanager in einer Gruppe mit gemeinsamer Warteschlange GROUP-Arbeitseinheiten mit Wiederherstellung verwenden möchten, müssen Sie das Warteschlangenmanager-Attribut 'GROUPUR' aktivieren. Weitere Informationen zum Konzept finden Sie im Abschnitt „Disposition einer Arbeitseinheit mit Wiederherstellung unter z/OS” auf Seite 309, den Sie zuerst lesen sollten.

Wenn das Warteschlangenmanager-Attribut GROUPUR aktiviert ist, akzeptiert der Warteschlangenmanager neue Verbindungen zu einer Disposition 'GROUP' der Arbeitseinheit mit Wiederherstellung. Wenn Sie dieses Attribut inaktivieren, werden neue Verbindungen zu dieser Disposition nicht akzeptiert. Bereits verbundene Anwendungen sind bis zur Unterbrechung der Verbindung davon nicht betroffen.

Wenn eine Anwendung eine Verbindung zu einer Disposition 'GROUP' der Arbeitseinheit mit Wiederherstellung herstellt und entweder abfragt, welche Transaktionen unbestätigt sind, oder versucht, eine Transaktion aufzulösen, die an einem anderen Ort in der Gruppe mit gemeinsamer Warteschlange (QSG) gestartet wurde, muss der Warteschlangenmanager, mit dem sie jetzt verbunden ist, mit den anderen Mitgliedern der Gruppe mit gemeinsamer Warteschlange kommunizieren können, damit er die Anforde-

rung bearbeiten kann. Zu diesem Zweck verwendet er eine gemeinsam genutzte Warteschlange mit der Bezeichnung SYSTEM.QSG.UR.RESOLUTION.QUEUE. Diese Warteschlange muss sich in einer wiederherstellbaren Anwendungsstruktur 'CSQSYSAPPL' befinden. Die Struktur muss wiederherstellbar sein, da bei der Verarbeitung von Auflösungsanforderungen in dieser Warteschlange persistente Nachrichten gespeichert werden.

Bevor Sie GROUP-Arbeitseinheiten mit Wiederherstellung aktivieren, müssen Sie sicherstellen, dass die Coupling-Facility-Struktur und die gemeinsam genutzte Warteschlange definiert sind. Sie können die Definitionen im CSQ4INSS-Beispiel verwenden. Wenn die Warteschlange definiert oder beim Start erkannt wird, öffnet jeder Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange die Warteschlange, damit eingehende Anforderungen empfangen werden können. Wenn Sie die Warteschlange löschen oder verschieben möchten, da sie nicht ordnungsgemäß definiert wurde, können Sie anfordern, dass die Warteschlangenmanager ihre offenen Kennungen in der Warteschlange abschließen, indem sie das Warteschlangenobjekt aktualisieren, damit MQET-Anforderungen blockiert werden. Wenn Sie die notwendigen Korrekturen vorgenommen haben, führt das Abrufen von Nachrichten aus der Warteschlange durch Anwendungen erneut dazu, dass jeder Warteschlangenmanager die Warteschlange erneut öffnet. Mit dem Befehl DISPLAY QSTATUS können Sie ermitteln, welche Kennungen in einer Warteschlange offen sind.

Wenn Sie diese Konfiguration abgeschlossen haben, können Sie auf jedem Warteschlangenmanager, mit dem Transaktionsanwendungen über eine Disposition 'GROUP' der Arbeitseinheit mit Wiederherstellung eine Verbindung herstellen können sollen, GROUP-Arbeitseinheiten mit Wiederherstellung aktivieren. Dies müssen nicht alle Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange sein. Wenn Sie diese Funktionalität jedoch nur in einer Untergruppe der Gruppe mit gemeinsamer Warteschlange aktivieren, müssen Sie sicherstellen, dass Ihre Anwendungen nur versuchen, eine Verbindung zu Warteschlangenmanagern herzustellen, auf denen Sie die Funktionalität aktiviert haben. Weitere Informationen finden Sie unter „Anwendungsunterstützung” auf Seite 311.

Bei der Aktivierung des Warteschlangenmanager-Attributs 'GROUPUR' werden mehrere Konfigurationsprüfungen durchgeführt. Der Warteschlangenmanager überprüft Folgendes:

- Er gehört zu einer Gruppe mit gemeinsamer Warteschlange.
- Die gemeinsam genutzte Warteschlange mit der Bezeichnung SYSTEM.QSG.UR.RESOLUTION.QUEUE wurde in Übereinstimmung mit der Definition in CSQ4INSS definiert.
- Die Warteschlange SYSTEM.QSG.UR.RESOLUTION.QUEUE befindet sich in einer wiederherstellbaren Coupling-Facility-Struktur mit dem Namen CSQSYSAPPL.

Schlägt eine der vorstehend genannten Prüfungen fehl, bleibt das Attribut 'GROUPUR' inaktiviert und es wird ein Nachrichtencode zurückgegeben.

Diese Konfigurationsprüfungen werden auch beim Start des Warteschlangenmanagers durchgeführt, wenn das Warteschlangenmanager-Attribut aktiviert ist. Sollten Prüfungen beim Start fehlschlagen, werden die GROUP-Arbeitseinheiten mit Wiederherstellung inaktiviert und der Warteschlangenmanager gibt eine Nachricht aus, in der angegeben ist, welche Prüfung fehlschlug. Nachdem Sie die notwendige Korrekturmaßnahme durchgeführt haben, müssen Sie das Warteschlangenmanager-Attribut erneut aktivieren.

## z/OS Anwendungsunterstützung

Verwenden Sie diese Seite, um zu bestimmen, welche Anwendungen eine Verbindung zu einer Disposition 'GROUP' der Arbeitseinheit mit Wiederherstellung herstellen können.

Unterstützung für die Disposition 'GROUP' der Arbeitseinheit mit Wiederherstellung ist auf bestimmte Transaktionsanwendungen beschränkt, für die IBM MQ for z/OS ein Ressourcenmanager, aber nicht der Transaktionskoordinator ist. Derzeit werden folgende Transaktionsanwendungen unterstützt:

- Erweiterte transaktionsorientierte IBM MQ-Clientanwendungen
- IBM MQ classes for JMS-Anwendungen in einem Anwendungsserver, z. B. WebSphere Application Server
- CICS -Anwendungen, die in CICS Transaction Server 4.2 oder höher ausgeführt werden, wenn die CICS MQCONN-Ressourcendefinition mit RESYNCMEMBER (GROUPRESYNC) konfiguriert ist.

**Zugehörige Konzepte**

„Erweiterte transaktionsorientierte IBM MQ-Clientanwendungen” auf Seite 312
Bestimmen Sie mithilfe dieser Seite, wie erweiterte transaktionsorientierte IBM MQ-Clientanwendungen die Disposition GROUP der Arbeitseinheit mit Wiederherstellung verwenden kann.

„CICS-Anwendungen” auf Seite 312
Bestimmen Sie mithilfe dieser Seite, wie CICS die Disposition GROUP der Arbeitseinheit mit Wiederherstellung verwenden kann.

## z/OS *Erweiterte transaktionsorientierte IBM MQ-Clientanwendungen*

Bestimmen Sie mithilfe dieser Seite, wie erweiterte transaktionsorientierte IBM MQ-Clientanwendungen die Disposition GROUP der Arbeitseinheit mit Wiederherstellung verwenden kann.

Ein Beispiel für eine Anwendung des erweiterten transaktionsorientierten IBM MQ-Clients ist eine Anwendung, die mit JMS arbeitet und in WebSphere Application Server mit einer IBM MQ-Verbindung über TCP/IP statt über lokale Bindungen ausgeführt wird. Diese Clientanwendungen stellen eine Verbindung mit IBM MQ for z/OS über Netzverbindungen (z. B. über TCP/IP) her. Bei diesen Anwendungen gibt der für den Parameter 'QMNAME' angegebene Wert der Zeichenfolge 'xa_info', die im Aufruf 'xa_open' übergeben wird, an, ob eine Disposition 'QMGR' oder 'GROUP' der Arbeitseinheit mit Wiederherstellung verwendet wird. Sie finden weitere Informationen zu 'xa_open' unter Format einer xa_open-Zeichenfolge und Zusätzliche Fehlerbehandlung für xa_open. Bei JMS-Anwendungen erfolgt dies durch Angabe des Namens der Gruppe mit gemeinsamer Warteschlange (QSG) in der Verbindungsfactory statt des Namens eines bestimmten Warteschlangenmanagers.

Damit XA-Clientanwendungen die Disposition 'GROUP' der Arbeitseinheit mit Wiederherstellung nutzen können, müssen Sie TCP/IP so konfigurieren, dass Ihre Clientanwendungen an die Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange weitergeleitet werden, bei denen das Attribut 'GROUPUR' aktiviert ist, statt an einen bestimmten Warteschlangenmanager. Eine der Technologien für dynamische virtuelle IP-Adressen, die Sie dafür verwenden können, ist der z/OS SysPlex Distributor. Weitere Informationen finden Sie in den Abschnitten z/OS Communications Server und z/OS Basic Skills: Dynamische virtuelle Adressierung. Wenn Sie GROUP-Arbeitseinheiten mit Wiederherstellung für eine Untergruppe der Warteschlangenmanager in Ihrer Gruppe mit gemeinsamer Warteschlange aktivieren möchten, müssen Sie sicherstellen, dass Ihre Clientanwendungen nicht an die Warteschlangenmanager weitergeleitet werden können, auf denen diese nicht aktiviert sind.

Ihre Clientanwendungen müssen keine Verbindung zur Gruppe mit gemeinsamer Warteschlange über gemeinsame Kanäle herstellen.

## z/OS *CICS-Anwendungen*

Bestimmen Sie mithilfe dieser Seite, wie CICS die Disposition GROUP der Arbeitseinheit mit Wiederherstellung verwenden kann.

CICS 4.2 und höher stellt die Gruppenresynchronisationsoption RESYNCMEMBER (GROUPRESYNC) in einer MQCONN-Ressourcendefinition bereit. Eine mit dieser Option konfigurierte CICS-Instanz kann eine Verbindung zu jedem geeigneten Warteschlangenmanager in einer Gruppe mit gemeinsamer Warteschlange herstellen, der in derselben logischen Partition (LPAR) wie die CICS-Region aktiv ist. Um die CICS-Option GROUPRESYNC unterstützen zu können, muss ein Warteschlangenmanager in MQ V7.1 oder höher ausgeführt werden und für die GROUPUR-Unterstützung aktiviert sein.

Transaktionen, die in einer CICS-Region aktiv sind, die über GROUPRESYNC mit MQ verbunden ist, erstellen Arbeitseinheiten mit der Disposition GROUP der Arbeitseinheit mit Wiederherstellung.

Sie können RESYNCMEMBER (GROUPRESYNC) verwenden, um eine schnellere Wiederherstellung nach einem Ausfall des Warteschlangenmanagers zu ermöglichen, da dies der CICS -Region ermöglicht, sofort eine Verbindung zu einem alternativen auswählbaren Warteschlangenmanager herzustellen, der in derselben LPAR ausgeführt wird, und alle unbestätigten Transaktionen nach Bedarf aufzulösen, ohne auf den Neustart des Warteschlangenmanagers zu warten.

RESYNCMEMBER (GROUPRESYNC) ermöglicht auch flexiblere Neustartoptionen für CICS. Eine CICS-Region, deren MQ-Verbindung für die Verwendung von GROUPRESYNC und gemeinsam genutzten MQ-War-

teschlangen konfiguriert ist, kann in jeder logischen Partition (LPAR) erneut gestartet werden, in der ein Warteschlangenmanager als Mitglied derselben Gruppe mit gemeinsamer Warteschlange aktiv ist.

# z/OS IBM MQ und andere z/OS-Produkte

In diesem Abschnitt wird erläutert, wie IBM MQ zusammen mit anderen z/OS-Produkten eingesetzt werden kann.

**Zugehörige Konzepte**

„IBM MQ und CICS" auf Seite 313
Alle CICS -Versionen, die von IBM MQ 9.0.0und höher unterstützt werden, verwenden die von CICS bereitgestellte Version des Adapters und der Bridge.

„IBM MQ for z/OS und WebSphere Application Server" auf Seite 320
In diesem Abschnitt wird die Verwendung von IBM MQ for z/OS durch WebSphere Application Server erläutert.

**Zugehörige Verweise**

„IBM MQ und IMS" auf Seite 314
In diesem Abschnitt wird erläutert, wie IBM MQ zusammen mit IMS eingesetzt werden kann. Mit dem IMS-Adapter können Sie eine Verbindung zwischen Ihrem Warteschlangenmanager und IMS herstellen und er ermöglicht IMS-Anwendungen die Verwendung der Schnittstelle für Warteschlangen (Message Queue Interface, MQI).

„IBM MQ und die z/OS Batch-, TSO- und RRS-Adapter" auf Seite 319
In diesem Abschnitt wird erläutert, wie IBM MQ zusammen mit den Adaptern für z/OS Batch, TSO und RRS eingesetzt werden kann.

## z/OS IBM MQ und CICS

Alle CICS -Versionen, die von IBM MQ 9.0.0und höher unterstützt werden, verwenden die von CICS bereitgestellte Version des Adapters und der Bridge.

Weitere Informationen zum Konfigurieren des IBM MQ CICS -Adapters und der IBM MQ CICS bridge -Komponenten finden Sie im Abschnitt Verbindungen zu IBM MQkonfigurieren der CICS -Dokumentation.

**Zugehörige Tasks**

IBM MQ mit CICS verwenden

### z/OS CICS-Gruppenanschluss

Der CICS-Gruppenanschluss bietet einer CICS-Region die Möglichkeit, eine Verbindung mit einem beliebigen Mitglied einer IBM MQ-Gruppe mit gemeinsamer Warteschlange in derselben logischen Partition (LPAR) herzustellen, statt einen einzelnen Warteschlangenmanager anzugeben. CICS stellt nach wie vor immer nur eine Verbindung mit einem einzelnen Warteschlangenmanager her.

Zur Unterstützung des CICS-Gruppenanschlusses sind mindestens zwei Warteschlangenmanager in der LPAR erforderlich. Der Gruppenanschluss bietet eine höhere Verfügbarkeit, da kein bestimmter Warteschlangenmanager aktiv sein muss. CICS stellt eine Verbindung mit einem beliebigen Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange in der LPAR her.

Weitere Informationen finden Sie in der CICS-Dokumentation in der Ressource MQCONN.

CICS versucht eine Verbindung mit dem übergebenen MQNAME herzustellen, als wäre es ein Warteschlangenmanager:

- Ist der Warteschlangenmanager vorhanden und aktiv, wird die Verbindung hergestellt.
- Schlägt die Verbindung fehl, fragt CICS den Status von Warteschlangenmanagern in der Gruppe ab, um zu ermitteln, welche in derselben LPAR aktiv sind.
- Wenn mehrere Warteschlangenmanager aktiv sind, sucht CICS nach RESYNCMEMBER (YES) und dem UOW-Status, um festzustellen, ob CICS eine Verbindung zu einem bestimmten Member herstellen muss

oder ob eine Verbindung zu einem bestimmten Member hergestellt werden soll oder warten muss, wenn es nicht aktiv ist.

- Muss keine Verbindung mit einem bestimmten Mitglied hergestellt werden, wählt CICS einen Warteschlangenmanager aus (über einen Zufallsalgorithmus).
- CICS versucht, eine Verbindung mit dem ausgewählten Warteschlangenmanager herzustellen.
- Schlägt der Versuch fehl, wählt CICS abhängig vom Rückgabecode das nächste Mitglied aus und durchläuft dann erneut die Auswahlschleife.
- Sind keine Warteschlangenmanager aktiv, gibt CICS mehrere Verbindungen an die Liste der Warteschlangenmanager aus und wartet an ECBLIST, bis der erste Warteschlangenmanager verfügbar wird.

**Zugehörige Konzepte**

„Gruppenarbeitseinheiten mit Wiederherstellung (GROUPUR) für CICS" auf Seite 314
Das IBM MQ-Attribut GROUPUR für CICS stellt Peerwiederherstellung für unbestätigte Arbeitseinheiten in einer Gruppe mit gemeinsamer Warteschlange bereit. Ein IBM MQ-Warteschlangenmanager kann unbestätigte Arbeitseinheiten im Auftrag eines anderen Warteschlangenmanagers in der Gruppe mit gemeinsamer Warteschlange auflösen. Das heißt, dass CICS, wenn es über die Gruppenzuordnung die Verbindung zu einem anderen Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange wiederherstellt, unbestätigte Transaktionen aus einer vorherigen IBM MQ-Verbindung auflösen kann.

**Zugehörige Informationen**

Unterstützung für IBM MQ-Gruppen mit gemeinsamer Warteschlange

## z/OS Gruppenarbeitseinheiten mit Wiederherstellung (GROUPUR) für CICS

Das IBM MQ-Attribut GROUPUR für CICS stellt Peerwiederherstellung für unbestätigte Arbeitseinheiten in einer Gruppe mit gemeinsamer Warteschlange bereit. Ein IBM MQ-Warteschlangenmanager kann unbestätigte Arbeitseinheiten im Auftrag eines anderen Warteschlangenmanagers in der Gruppe mit gemeinsamer Warteschlange auflösen. Das heißt, dass CICS, wenn es über die Gruppenzuordnung die Verbindung zu einem anderen Warteschlangenmanager in der Gruppe mit gemeinsamer Warteschlange wiederherstellt, unbestätigte Transaktionen aus einer vorherigen IBM MQ-Verbindung auflösen kann.

Wenn eine CICS-Region mit einem Warteschlangenmanager arbeitet und der Warteschlangenmanager abnormal beendet wird, werden alle unbestätigten Transaktionen wiederhergestellt. Die CICS-Region muss dann nicht darauf warten, dass der Warteschlangenmanager, mit dem sie arbeitet, erneut gestartet wird und danach alle unbestätigten Arbeitseinheiten auflöst. Das heißt, dass mindestens zwei Warteschlangenmanager in der LPAR erforderlich sind, damit CICS eine Verbindung mit einem anderen Warteschlangenmanager herstellen kann, falls der erste Warteschlangenmanager abnormal beendet wird.

Die neue Einstellung RESYNCMEMBER (GROUPRESYNC) in der CICS -MQCONN-Definition:

- Verwendet die IBM MQ-Gruppenanschlussfunktion und Peerwiederherstellung.
- Erfordert einen Warteschlangenmanager mit aktiviertem GROUPUR-Attribut.
- Unterstützt weiterhin die vorhandenen CICS MQCONN RESYNCMEMBER-Einstellungen (YES und NO):
  - Verwendet die vorhandene CICS-Gruppenanschlussfunktion und keine Peerwiederherstellung.
  - Die Änderung der RESYNCMEMBER-Einstellungen wird wirksam, wenn CICS das nächste Mal eine Verbindung zu IBM MQherstellt.

**Zugehörige Konzepte**

„GROUP-Arbeitseinheiten mit Wiederherstellung aktivieren" auf Seite 310
Eine Gruppe mit gemeinsamer Warteschlange kann Unterstützung für GROUP-Arbeitseinheiten mit Wiederherstellung konfigurieren und aktivieren.

## z/OS IBM MQ und IMS

In diesem Abschnitt wird erläutert, wie IBM MQ zusammen mit IMS eingesetzt werden kann. Mit dem IMS-Adapter können Sie eine Verbindung zwischen Ihrem Warteschlangenmanager und IMS herstellen

und er ermöglicht IMS-Anwendungen die Verwendung der Schnittstelle für Warteschlangen (Message Queue Interface, MQI).

Die optionale zusätzliche IBM MQ - IMS-Bridge ermöglicht Anwendungen, die Ausführung einer IMS-Anwendung ohne Verwendung der MQI-Schnittstelle. Das heißt, Sie können Ihre traditionellen Anwendungen zusammen mit IBM MQ verwenden, ohne sie neu programmieren zu müssen.

Weitere Informationen zu diesen Komponenten finden Sie in folgenden Unterabschnitten:

**Zugehörige Konzepte**

IMS- und IMS-Brückenanwendungen unter IBM MQ for z/OS

**Zugehörige Tasks**

IMS-Adapter einrichten

IMS-Bridge konfigurieren

IMS-Adapter betreiben

**Zugehörige Verweise**

MQIIH - IMS-Informationsheader

## z/OS IMS-Adapter

Der IMS-Adapter ist eine Schnittstelle zwischen IMS-Anwendungsprogrammen und einem IBM MQ-Subsystem.

Die IBM MQ-Adapter machen es für unterschiedliche Anwendungsumgebungen möglich, Nachrichten über ein Netz zur Steuerung von Warteschlangen zu senden und zu empfangen. Der IMS-Adapter ist die Schnittstelle zwischen IMS-Anwendungsprogrammen und einem IBM MQ-Subsystem. Diese Schnittstelle ermöglicht IMS-Anwendungsprogrammen die Verwendung der Schnittstelle für Nachrichtenwarteschlangen (Message Queue Interface, MQI).

Der IMS -Adapter empfängt und interpretiert Anforderungen für den Zugriff auf IBM MQ unter Verwendung der von IMSbereitgestellten External Subsystem Attach Facility (ESAF) . Normalerweise stellt IMS automatisch ohne Bedienereingriff eine Verbindung zu IBM MQ her.

Der IMS-Adapter ermöglicht den Zugriff auf IBM MQ-Ressourcen für Programme, die in folgenden Modi bzw. mit folgenden Status ausgeführt werden:

- Taskmodus (Tasksteuerblock)
- Fehlerstatus
- Im nicht speicherübergreifenden Modus
- Im Modus ohne Registerzugriff

Der Adapter stellt einen Verbindungsthread von einem Anwendungstasksteuerblock zu IBM MQ bereit.

Der Adapter unterstützt ein Protokoll für zweiphasige Festschreibung, das für Änderungen an IBM MQ-Ressourcen mit IMS als Synchronisationspunktkoordinator verwendet wird. Datenaustauschvorgänge, bei denen IMS nicht der Synchronisationspunktkoordinator ist, z. B. ein APPC-geschützter (SYNCLVL=SYNCPT) Datenaustausch, werden vom IMS-Adapter nicht unterstützt.

Der Adapter stellt auch eine Auslösemonitortransaktion (CSQQTRMN) bereit. Weitere Informationen hierzu finden Sie unter „IMS-Auslösemonitor” auf Seite 316.

Sie können IBM MQ mit der erweiterten Wiederherstellungsfunktion (Extended Recovery Facility, XRF) von IMS verwenden, um bei IMS-Fehlern die Wiederherstellung zu erleichtern.

**Anmerkung:** Ab IMS 15.2 wird Extended Recovery Facility (XRF) nicht mehr unterstützt. Weitere Informationen finden Sie in der IMS -Dokumentation.

### Adapter verwenden

Die Anwendungsprogramme und der IMS-Adapter werden in demselben Adressraum ausgeführt. Der Warteschlangenmanager befindet sich in einem separaten, eigenen Adressraum.

Für jedes Programm, dass mindestens einen MQI-Aufruf ausgibt, müssen Sie eine Programmverknüpfung zu einem geeigneten IMS-Sprachschnittstellenmodul und, sofern es sich nicht um dynamische MQI-Aufrufe handelt, auch zu dem von IBM MQ bereitgestellten API-Stubprogramm CSQQSTUB herstellen. Wenn die Anwendung einen MQI-Aufruf ausgibt, überträgt der Stub die Steuerung mithilfe der externen IMS-Subsystemschnittstelle, die die Verarbeitung der Anforderung durch den Nachrichtenwarteschlangenmanager verwaltet, an den Adapter.

## Systemverwaltung und -bedienung mit IMS

Ein entsprechend berechtigter IMS-Terminalbediener kann IMS-Befehle ausgeben, um die Verbindung zu IBM MQ zu steuern und zu überwachen. Der IMS-Terminalbediener hat jedoch keinen Einfluss auf den IBM MQ-Adressraum. Beispielsweise kann der Bediener IBM MQ nicht von einem IMS-Adressraum aus herunterfahren.

## Einschränkungen

Folgende IBM MQ-API-Aufrufe werden nicht in Anwendungen unterstützt, die den IMS-Adapter verwenden:

- MQCB
- MQCB_FUNCTION
- MQCTL

## IMS-Auslösemonitor

Der IMS-Auslösemonitor (**CSQQTRMN**) ist eine von IBM MQ bereitgestellte IMS-Anwendung, die eine IMS-Transaktion startet, wenn ein IBM MQ--Ereignis auftritt, z. B. wenn eine Nachricht in eine bestimmte Warteschlange eingereiht wird.

### Funktionsweise

Bei Einreihung einer Nachricht in eine Warteschlange für Anwendungsnachrichten wird ein Auslöser generiert, sofern die Auslöserbedingungen erfüllt werden. Dann sendet der Warteschlangenmanager eine Nachricht die als *Auslösenachricht* bezeichnet wird und benutzerdefinierte Daten enthält, an die Initialisierungswarteschlange, die für diese Nachrichtenwarteschlangen angegeben wurde. In einer IMS-Umgebung können Sie eine Instanz von 'CSQQTRMN' starten, um eine Initialisierungswarteschlange zu überwachen und die Auslösenachrichten unmittelbar nach ihrem Eintreffen abzurufen. Normalerweise plant 'CSQQTRMN' eine weitere IMS-Transaktion mithilfe des Befehls 'INSERT' (ISRT) für die IMS-Nachrichtenwarteschlange. Die gestartete IMS-Anwendung ruft die Nachricht aus der Anwendungsnachrichtenwarteschlange ab und verarbeitet sie anschließend. 'CSQQTRMN' muss als nicht auf Nachrichten basierende Bean-managed Persistence (BMP) ausgeführt werden.

Jede Kopie von 'CSQQTRMN' bedient eine einzelne Initialisierungswarteschlange. Der Auslösemonitors wird nach seinem Start bis zu Beendigung von IBM MQ oder IMS ausgeführt.

Im Makro 'APPLCTN' für 'CSQQTRMN' muss der Parameter 'SCHDTYP=PARALLEL' angegeben werden.

Da der Auslösemonitor eine stapelorientierte Bean-managed Persistence (BMP) ist, enthalten die vom Auslösemonitor gestarteten IMS-Transaktionen Folgendes:

- Leerzeichen im Feld 'LTERM' des Eingabe-/Ausgabe-PCB
- Den Programmspezifikationsblocknamen der Auslösemonitor-BMP im Feld mit der Benutzer-ID des Eingabe-/Ausgabe-PCB

Wenn die IMS-Zieltransaktion durch Security Server (zuvor unter dem Namen RACF bekannt) geschützt wird, müssen Sie möglicherweise 'CSQQTRMN' als Benutzer-ID für Security Server definieren.

## z/OS IBM MQ - IMS-Bridge

Die IBM MQ-IMS-Bridge ist die Komponente von IBM MQ for z/OS, die den direkten Zugriff von IBM MQ-Anwendungen auf das IMS-System ermöglicht.

Die IBM MQ-IMS-Bridge ermöglicht *implizite MQI-Unterstützung*. Dies bedeutet, dass traditionelle Anwendungen, die über Terminals mit 3270-Verbindung gesteuert wurden, so umstrukturiert werden können, dass sie nun durch IBM MQ-Nachrichten gesteuert werden, ohne dass diese Anwendungen neu programmiert, neu kompiliert oder neu verknüpft werden müssen. Die Bridge ist ein IMS *Open Transaction Manager Access*-(OTMA-)Client.

In Bridge-Anwendungen gibt es keine IBM MQ-Aufrufe innerhalb der IMS-Anwendung. Die Anwendung ruft ihre Eingabe mit dem an den Eingabe-/Ausgabe-PCB gerichteten Befehl 'GET UNIQUE' (GU) ab und sendet ihre Ausgabe mit dem Befehl 'ISRT' ebenfalls an den Eingabe-/Ausgabe-PCB. IBM MQ-Anwendungen stellen anhand des IMS-Headers (d. h. der MQIIH-Struktur) in den Nachrichtendaten sicher, dass die Anwendungen weiterhin so ausgeführt werden können, wie es der Fall war, als sie durch nicht programmierbare Terminals gesteuert wurden. Wenn Sie eine IMS-Anwendung verwenden, die Nachrichten mit mehreren Segmenten verarbeitet, ist zu beachten, dass alle Segmente in einer einzelnen IBM MQ-Nachricht enthalten sein müssen.

Die IMS-Bridge ist in Abbildung 78 auf Seite 317 dargestellt.

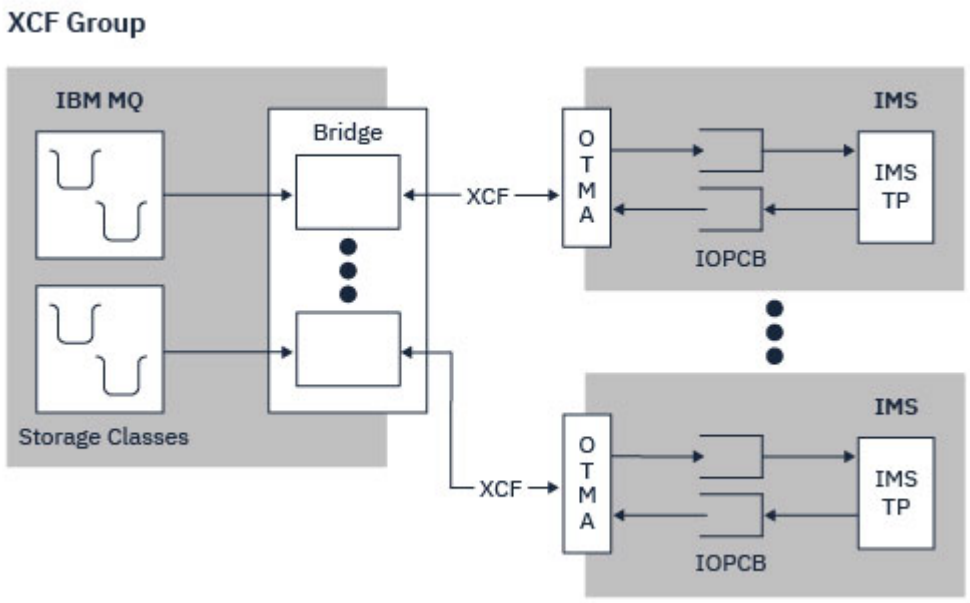



*Abbildung 78. IBM MQ - IMS-Bridge*

Ein Warteschlangenmanager kann mit einem oder mehreren IMS-Systemen verbunden sein und es können auch mehrere Warteschlangenmanager mit einem IMS-System verbunden sein. Die einzige Einschränkung besteht darin, dass alle zu derselben XCF-Gruppe gehören und sich alle in demselben Sysplex befinden müssen.

Im Abschnitt IMS -Bridge einrichten finden Sie Informationen zum Einrichten einer IMS -Bridge und zum Hinzufügen einer zusätzlichen IMS -Verbindung zu demselben Warteschlangenmanager.

### Was ist OTMA (Open Transaction Manager Access)?

Die IMS OTMA-Funktion ist ein transaktionsbasiertes verbindungsunabhängiges Client/Server-Protokoll, das unter IMSausgeführt wird. Es fungiert als Schnittstelle für hostbasierte Kommunikationsserver, die über die z/OS Cross Systems Coupling Facility (XCF)auf IMS TM-Anwendungen zugreifen.

Mit der OTMA-Funktion können Clients eine Verbindung zu IMS herstellen, um Hochleistungsinteraktionen zwischen Clients und IMS für ein großes Netz oder eine große Anzahl von Sitzungen zu ermöglichen. OTMA wird in einer z/OS-Sysplex-Umgebung implementiert. Deshalb ist die OTMA-Domäne auf die Domäne von XCF beschränkt.

## OTMA-Ressourcenüberwachung

Unterstützung für die OTMA-Protokollnachrichten x '3C', die in IMS v10 oder höher verfügbar sind, ist in der Brücke IBM MQ - IMS in IBM MQ for z/OSenthalten. In diesen Nachrichten gibt IMS den OTMA-Clients Aufschluss über seinen Allgemeinzustand.

Falls IMS-Partner die Menge der gesendeten Transaktionsanforderungen nicht mehr verarbeiten kann, meldet er IBM MQ, dass eine Überflutungswarnung ausgegeben wurde. Als Antwort darauf verringert IBM MQ die Geschwindigkeit, mit der die Anforderungen über die Bridge übertragen werden.

Falls IMS nach wie vor nicht in der Lage ist, mit den Transaktionsanforderungen Schritt zu halten und eine tatsächliche Überflutungsbedingung eintritt, werden alle TPIPEs an den IMS-Partner ausgesetzt. Erst wenn der IMS-Partner meldet, dass die Überflutungsbedingung bzw. die Überflutungswarnung aufgehoben wurde, nimmt IBM MQ die ausgesetzten TPIPEs wieder auf und erhöht allmählich die Geschwindigkeit, mit der die Transaktionsanforderungen gesendet werden, bis hin zur Maximalgeschwindigkeit. Konsolennachrichten werden von IBM MQ als Antwort auf eine Statusänderung von IMS -Partnern ausgegeben.

Bei Verwendung von IMS v10-Partnern sollten Sie sicherstellen, dass PTF UK45082 angewendet wurde.

## IMS-Transaktionen von IBM MQ übergeben

Um eine IMS-Transaktion, die die Bridge verwendet, zu übergeben, reihen Anwendungen wie gewöhnlich Nachrichten in eine IBM MQ-Warteschlange ein. Die Nachrichten enthalten IMS-Transaktionsdaten. Sie können einen IMS-Header (d. h. die MQIIH-Struktur) haben oder zulassen, dass die IBM MQ-IMS-Bridge Vorgaben für die Daten in der Nachricht macht.

Danach reiht IBM MQ die Nachricht in eine IMS-Warteschlange ein (zunächst wird die Nachricht jedoch in IBM MQ in eine Warteschlange eingereiht, um die Verwendung von Synchronisationspunkten zur Absicherung der Datenintegrität zu ermöglichen). Die Speicherklasse der IBM MQ-Warteschlange bestimmt, ob es sich um eine *OTMA-Warteschlange* handelt (d. h. eine Warteschlange, die für die Übertragung von Nachrichten an die IBM MQ-IMS-Bridge dient), und legt den bestimmten IMS-Partner fest, an den die Nachrichtendaten gesendet werden.

Auch ferne Warteschlangenmanager können IMS-Transaktionen starten, indem sie Daten in diese OTMA-Warteschlangen in IBM MQ for z/OS schreiben.

Die vom IMS-System zurückgegebenen Daten werden direkt in die IBM MQ-Warteschlange für zu beantwortende Nachrichten geschrieben, die in der Nachrichtendeskriptorstruktur (MQMD) angegeben ist. (Dies kann eine Übertragungswarteschlange für den Warteschlangenmanager sein, der im Feld **ReplyToQMgr** der MQMD-Struktur angegeben ist.)

**Zugehörige Konzepte**

IMS- und IMS-Brückenanwendungen unter IBM MQ for z/OS

**Zugehörige Tasks**

IMS-Bridge anpassen

**Zugehörige Verweise**

„IBM MQ und IMS” auf Seite 314

In diesem Abschnitt wird erläutert, wie IBM MQ zusammen mit IMS eingesetzt werden kann. Mit dem IMS-Adapter können Sie eine Verbindung zwischen Ihrem Warteschlangenmanager und IMS herstellen

und er ermöglicht IMS-Anwendungen die Verwendung der Schnittstelle für Warteschlangen (Message Queue Interface, MQI).

# z/OS IBM MQ und die z/OS Batch-, TSO- und RRS-Adapter

In diesem Abschnitt wird erläutert, wie IBM MQ zusammen mit den Adaptern für z/OS Batch, TSO und RRS eingesetzt werden kann.

## Einführung zu den Batch-Adaptern

Die Batch-/TSO-Adapter sind die Schnittstelle zwischen IBM MQ und z/OS-Anwendungsprogrammen, die unter JES (Job Entry Subsystem), TSO (Time Sharing Option) oder z/OS UNIX System Services ausgeführt werden. Diese Adapter ermöglichen z/OS-Anwendungsprogrammen die Verwendung der Schnittstelle für Nachrichtenwarteschlangen (Message Queue Interface, MQI).

Die Adapter ermöglichen den Zugriff auf IBM MQ-Ressourcen für Programme, die in folgenden Modi bzw. mit folgenden Status ausgeführt werden:

- Taskmodus (Tasksteuerblock)
- Fehlerstatus oder Supervisorstatus
- Im nicht speicherübergreifenden Modus
- Im Modus ohne Registerzugriff

Die Verbindungen zwischen Anwendungsprogrammen und IBM MQ finden auf Taskebene statt. Die Adapter stellen einen Verbindungsthread von einem Anwendungstasksteuerblock zu IBM MQ bereit.

Der Batch-/TSO-Adapter unterstützt ein Protokoll für einphasige Festschreibung, das für Änderungen an IBM MQ-Ressourcen verwendet wird. Er unterstützt keine Protokolle für mehrphasige Festschreibung. Der RRS-Adapter ermöglicht IBM MQ-Anwendungen zusammen mit anderen RRS-fähigen Produkten die Teilnahme an Protokollen für zweiphasige Festschreibung, die durch z/OS RRS (Resource Recovery Services) koordiniert wird.

Mithilfe des Service z/OS STIMERM planen die Adapter für jede Sekunde ein asynchrones Ereignis. Dieses Ereignis führt einen Interruptanforderungsblock (Interrupt Request Block, IRB) aus, wodurch jede Wartezeit für die Batch-Anwendungstask vermieden wird. Der Interruptanforderungsblock überprüft, ob das Ereignissteuerbit für die Beendigung von IBM MQ übertragen wurde. Wenn dies der Fall ist, überträgt der Interruptanforderungsblock alle Anwendungsereignissteuerbits, die auf ein Ereignis in IBM MQ warten (z. B. auf ein Signal oder einen Wartestatus).

## Batch-/TSO-Adapter

Der Batch-/TSO-Adapter von IBM MQ stellt IBM MQ Unterstützung für z/OS Batch- und TSO-Anwendungen bereit. Für alle Anwendungsprogramme, die unter z/OS Batch oder TSO ausgeführt werden, muss eine Programmverbindung zum API-Stubprogramm 'CSQBSTUB' hergestellt werden. Der Stub bietet der Anwendung Zugriff auf alle MQI-Aufrufe. Sie können die einphasige Festschreibung und Zurücksetzung für Anwendungen verwenden, indem Sie die MQI-Aufrufe **MQCMIT** und **MQBACK** ausgeben.

## RRS-Adapter

Resource Recovery Services (RRS) ist eine Unterkomponente von z/OS, die einen systemweiten Service für die Koordination der zweiphasigen Festschreibung in mehreren z/OS-Produkten bereitstellt. Der IBM MQ-Batch/TSO-RRS-Adapter (kurz RRS-Adapter genannt) bietet IBM MQ-Unterstützung für z/OS Batch- und TSO-Anwendungen, die diese Services nutzen möchten. Der RRS-Adapter ermöglicht IBM MQ die vollständige Teilnahme an der RRS-Koordination. Anwendungen können zusammen mit anderen Produkten, die RRS unterstützen (z. B. Db2), an der Verarbeitung der zweiphasigen Festschreibung teilnehmen.

Der RRS-Adapter stellt zwei Stubs bereit. Für Anwendungsprogramme, die RRS verwenden möchten, muss eine Programmverbindung zu einem dieser Stubs hergestellt werden.

**CSQBRSTB**
Mithilfe dieses Stubs können Sie die zweiphasige Festschreibung und Zurücksetzung für Anwendungen verwenden, indem Sie die aufrufbaren RRS-Services anstatt der MQI-Aufrufe **MQCMIT** und **MQBACK** verwenden.

Darüber hinaus müssen Sie für das Modul 'ATRSCSS' in der Bibliothek 'SYS1.CSSLIB' eine Programmverbindung zu Ihrer Anwendung herstellen. Wenn Sie die MQI-Aufrufe **MQCMIT** und **MQBACK** verwenden, erhalten Sie den Rückkehrcode `MQRC_ENVIRONMENT_ERROR`.

**CSQBRRSI**
Mithilfe dieses Stubs können Sie die MQI-Aufrufe **MQCMIT** und **MQBACK** verwenden. Tatsächlich werden diese Aufrufe von IBM MQ als die RSS-Aufrufe **SRRCMIT** und **SRRBACK** implementiert.

Weitere Informationen zum Erstellen von Anwendungsprogrammen, die den RRS-Adapter verwenden, finden Sie unter RRS-Batchadapter.

### Weitere Informationsquellen zu den Adaptern für z/OS Batch, TSO und RRS

Weitere Informationen zu den Themen in diesem Abschnitt finden Sie in folgenden Quellen:

*Tabelle 25. Weitere Informationsquellen zur Verwendung von z/OS Batch mit IBM MQ*

| Thema | Quelle |
|---|---|
| Batch-Adapter konfigurieren | Aufgabe 19: Stapel-, TSO- und RRS-Adapter konfigurieren |
| Aufrufbare RRS-Services | *MVS-Programmierung: Callable Services for High Level Languages* |

## z/OS IBM MQ for z/OS und WebSphere Application Server

In diesem Abschnitt wird die Verwendung von IBM MQ for z/OS durch WebSphere Application Server erläutert.

Anwendungen, die in Java geschrieben wurden und unter WebSphere Application Server ausgeführt werden, können die Spezifikation Java Message Service (JMS) für Messaging verwenden. Das Punkt-zu-Punkt-Messaging kann in dieser Umgebung durch einen IBM MQ for z/OS-Warteschlangenmanager bereitgestellt werden.

Ein Vorteil der Verwendung eines IBM MQ for z/OS-Warteschlangenmanagers zur Bereitstellung der Nachrichtenübermittlung ist, dass JMS-Anwendungen die Funktionalität eines IBM MQ-Netzes vollständig nutzen können. Beispielsweise können sie die IMS-Bridge verwenden oder Nachrichten mit Warteschlangenmanagern auf anderen Plattformen austauschen.

### Verbindung zwischen WebSphere Application Server und einem Warteschlangenmanager

Weitere Informationen finden Sie unter IBM MQ und WebSphere Application Server zusammen verwenden .

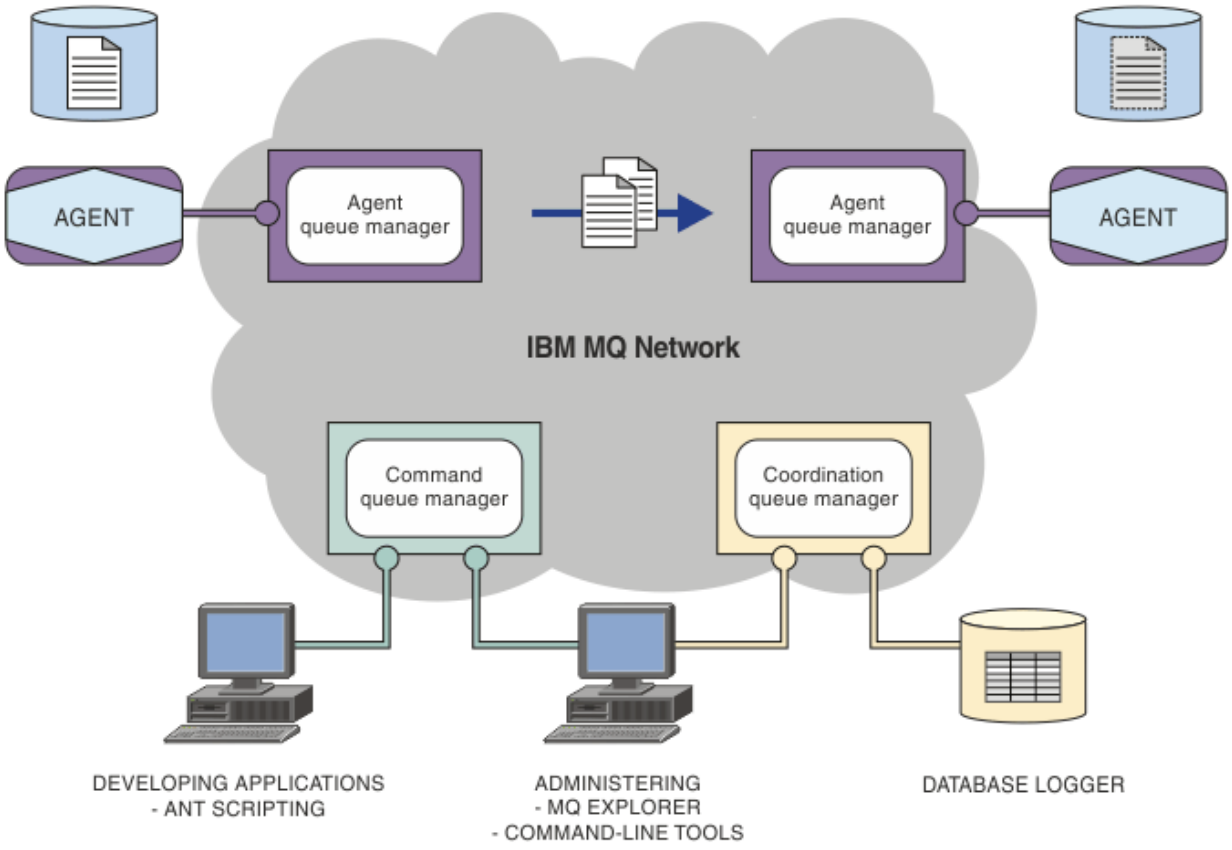
### IBM MQ-Funktionen aus JMS-Anwendungen verwenden

Standardmäßig verwenden JMS -Nachrichten, die in IBM MQ -Warteschlangen gehalten werden, einen MQRFH2 -Header, um einige der JMS -Nachrichtenheaderinformationen aufzunehmen. Viele traditionelle IBM MQ-Anwendungen können Nachrichten mit diesen Headern nicht verarbeiten und erfordern ihre eigenen spezifischen Header, z. B. MQCIH für die CICS-Bridge oder MQWIH für IBM MQ-Workflow-Anwendungen. Weitere Informationen zu diesen besonderen Überlegungen finden Sie unter JMS-Nachrichten und IBM MQ-Nachrichten einander zuordnen.

# Managed File Transfer

Managed File Transfer überträgt Dateien auf kontrollierte und überprüfbare Weise zwischen Systemen - unabhängig von der Dateigröße und den verwendeten Betriebssystemen.

Mit Managed File Transfer können Sie eine angepasste, skalierbare und automatisierte Lösung erstellen, mit der Sie Dateiübertragungen verwalten, sichern und schützen können. Managed File Transfer kommt ganz ohne kostspielige Redundanzen aus. Es verringert die Wartungskosten und maximiert die Ergebnisse aus Ihren IT-Investitionen.




Das Diagramm zeigt eine einfache Managed File Transfer-Topologie. Die Topologie enthält zwei Agenten, von denen jeder innerhalb des IBM MQ-Netzes mit seinem eigenen Agentenwarteschlangenmanager verbunden ist. Die Dateiübertragung findet vom Agenten auf der einen Seite des Diagramms über das IBM MQ-Netz zum Agenten auf der anderen Seite des Diagramms statt. Das IBM MQ-Netz enthält außerdem einen Koordinationswarteschlangenmanager und einen Befehlswarteschlangenmanager. Anwendungen und Tools stellen Verbindungen zu diesen Warteschlangenmanagern her, um Managed File Transfer-Aktivitäten im IBM MQ-Netz zu konfigurieren, zu verwalten, auszuführen und zu protokollieren.

Für Managed File Transfer stehen vier Installationsoptionen zur Auswahl; für welche Option Sie sich entscheiden, hängt von Ihrem Betriebssystem und von Ihrer Konfiguration ab. Diese Optionen sind Managed File Transfer Agent, Managed File Transfer Logger, Managed File Transfer Service oder Managed File Transfer Tools. Weitere Informationen finden Sie im Abschnitt Produktoptionen für Managed File Transfer.

Mit Managed File Transfer können Sie folgende Tasks ausführen:

- Gesteuerte Dateiübertragungen erstellen
  - Windows Linux Neue Dateiübertragungen aus IBM MQ Explorer auf Linux -oder Windows -Plattformen erstellen.
  - Auf allen Plattformen neue Dateiübertragungen über die Befehlszeile erstellen.

  - Dateiübertragungsfunktion in das Tool Apache Ant integrieren.
  - Anwendungen schreiben, die Managed File Transfer durch Übergabe von Nachrichten in die Befehlswarteschlangen der Agenten steuern.
  - Dateiübertragungen planen, sodass sie zu einem späteren Zeitpunkt ausgeführt werden. Außerdem können geplante Dateiübertragungen basierend auf einer Reihe von Dateisystemereignissen ausgelöst werden, z. B. beim Erstellen einer neuen Datei.
  - Eine Ressource (beispielsweise ein Verzeichnis) ständig überwachen; erfüllt der Inhalt dieser Ressource eine vordefinierte Bedingung, wird eine bestimmte Task gestartet. Bei dieser Task kann es sich um eine Dateiübertragung, ein Ant-Script oder einen JCL-Job handeln.
  - Dateien zu und von IBM MQ-Warteschlangen übertragen.
  - Dateien zu und von FTP-, FTPS- oder SFTP-Servern übertragen.
  - Dateien zu und von Connect:Direct-Knoten übertragen.
  - Sowohl Textdateien als auch binäre Dateien übertragen. Bei Textdateien werden Codepages und Zeilenendekonventionen automatisch zwischen den Quellen- und Zielsystemen konvertiert.
  - Übertragungen können gesichert werden. Dazu werden die Industriestandards für auf Secure Socket Layer (SSL) basierende Verbindungen angewendet.
- Laufende Übertragungen anzeigen und Informationen zu allen Übertragungen in Ihrem Netz protokollieren
  - 

Zeigen Sie den Status von Übertragungen in Bearbeitung von IBM MQ Explorer auf Linux -oder Windows -Plattformen an.
  - 

Überprüfen Sie den Status abgeschlossener Übertragungen mithilfe der IBM MQ Explorer auf Linux -oder Windows -Plattformen.
  - Mit der Datenbankprotokollfunktion von Managed File Transfer Protokollnachrichten in einer Db2- oder Oracle-Datenbank speichern.

Managed File Transfer setzt aufIBM MQ auf und stellt somit die gleiche sichere und zuverlässige Zustellung von Nachrichten zwischen Anwendungen bereit. Auch in diesem Tool können Sie die Vorteile der verschiedenen Funktionen von IBM MQ nutzen. Beispielsweise können Sie mit einem Kanalausdruck die zwischen Agenten über IBM MQ-Kanäle übertragenen Daten komprimieren und mittels SSL-Kanälen die zwischen Agenten ausgetauschten Daten schützen. Dateien können zuverlässig übertragen werden, mit Fehlertoleranz gegenüber Ausfällen der Infrastruktur, über die die Dateiübertragung ausgeführt wird. Bei einem Netzausfall wird die Dateiübertragung am Abbruchpunkt neu gestartet, sobald die Verbindung wiederhergestellt ist.

Durch die Konsolidierung der Dateiübertragungsfunktion mit Ihrem bestehenden IBM MQ-Netz sind keine weiteren Ressourcen mehr zur Verwaltung einer zweiten Infrastruktur erforderlich. Wenn Sie noch kein IBM MQ-Kunde sind, errichten Sie durch die Einrichtung eines IBM MQ-Netzes zur Unterstützung von Managed File Transfer den Backbone für eine zukünftige SOA-Implementierung. Wenn Sie IBM MQ bereits verwenden, können Sie mit Managed File Transfer auch die Vorteile Ihrer bestehenden IBM MQ-Infrastruktur einschließlich IBM MQ Internet Pass-Thru und IBM Integration Bus nutzen.

Sie können IBM MQ -Hochverfügbarkeitslösungen nutzen, um die Ausfallsicherheit Ihrer Managed File Transfer -Konfiguration zu verbessern. Wenn Ihre Agenten replizierte Datenwarteschlangenmanager (RDQMs) verwenden, müssen Sie sie für die Verwendung der Funktion für variable IP-Adressen konfigurieren. Dies bedeutet, dass Agenten dieselbe IP-Adresse für die Kommunikation mit jeder der drei RDQM-Instanzen verwenden, die derzeit ausgeführt wird, und die Verbindung bei einem Failover automatisch wiederherstellen (siehe RDQM-Hochverfügbarkeit und Variable IP-Adresse erstellen und löschen). Wenn Sie die Multi-Instanz-Warteschlangenmanager-Lösung verwenden, verwenden Anwendungen eine andere IP-Adresse für die Kommunikation mit jeder Instanz, die von der Wiederherstellung der Clientverbindung bei der Übernahme verarbeitet wird (siehe Multi-Instanz-Warteschlangenmanager und Kanal und Clientverbindung).

Managed File Transfer lässt sich mit verschiedenen IBM-Produkten integrieren:

**IBM Integration Bus**

Mit diesem Produkt können Sie Dateien verarbeiten, die von Managed File Transfer im Zuge eines IBM Integration Bus Nachrichtenflusses übertragen wurden. Weitere Informationen finden Sie im Abschnitt Mit MFT aus IBM Integration Bus arbeiten.

**IBM Sterling Connect:Direct**

Übertragen Sie Dateien zu und von einem vorhandenen Connect:Direct-Netz unter Verwendung der Managed File Transfer Connect:Direct-Bridge. Weitere Informationen finden Sie im Abschnitt Die Connect:Direct-Bridge.

**IBM Tivoli Composite Application Manager**

IBM Tivoli Composite Application Manager stellt einen Agenten bereit, mit dem Sie die auf dem Koordinationswarteschlangenmanager veröffentlichten Informationen überwachen können.

**Zugehörige Konzepte**

Produktoptionen für Managed File Transfer

„Übersicht über die MFT-Topologie" auf Seite 324
Dieser Abschnitt gibt eine Übersicht darüber, wie Managed File Transfer-Agenten mit dem Koordinationswarteschlangenmanager in einem IBM MQ-Netz verbunden sind.

„Funktionsweise von MFT mit IBM MQ" auf Seite 323
Managed File Transfer interagiert auf verschiedene Weise mit IBM MQ.

# Funktionsweise von MFT mit IBM MQ

Managed File Transfer interagiert auf verschiedene Weise mit IBM MQ.

- Managed File Transfer überträgt Dateien zwischen Agentenprozessen, indem die einzelnen Dateien in eine oder mehrere Nachrichten aufgeteilt und diese Nachrichten über das IBM MQ-Netz übertragen werden.
- Die Agentenprozesse verschieben Dateidaten unter Verwendung nicht persistenter Nachrichten, um die Auswirkungen auf Ihre IBM MQ-Protokolle zu minimieren. Durch Kommunikation untereinander steuern die Agentenprozesse den Fluss von Nachrichten mit Dateidaten. Dadurch wird die Ansammlung von Nachrichten mit Dateidaten in IBM MQ-Übertragungswarteschlangen verhindert und sichergestellt, dass im Fall nicht zugestellter nicht persistenter Nachrichten die Dateidaten erneut gesendet werden.
- Managed File Transfer-Agenten verwenden verschiedene IBM MQ-Warteschlangen. Weitere Informationen finden Sie im Abschnitt MFT-Systemwarteschlangen und das System.
- Zwar sind einige dieser Warteschlangen ausschließlich zur internen Verwendung vorgesehen, ein Agent kann jedoch Anforderungen in Form speziell formatierter Befehlsnachrichten akzeptieren, die an eine bestimmte vom Agenten überwachte Warteschlange gesendet werden. Sowohl die Befehlszeilenbefehle als auch das IBM MQ Explorer-Plug-in senden IBM MQ-Nachrichten an den Agenten, um diesen zur Ausführung der gewünschten Aktion anzuleiten. Auch Sie können IBM MQ-Anwendungen schreiben, die in dieser Weise mit dem Agenten interagieren. Weitere Informationen finden Sie im Abschnitt MFT durch Einreihen von Nachrichten in die Befehlswarteschlange des Agenten steuern.
- Managed File Transfer-Agenten senden Informationen zum eigenen Status sowie zum Fortschritt und Ergebnis der Übertragungen an einen MQ-Warteschlangenmanager, der als Koordinationswarteschlangenmanager fungiert. Diese Informationen werden vom Koordinationswarteschlangenmanager publiziert und können von Anwendungen subskribiert werden, um den Übertragungsfortschritt zu überwachen oder die ausgeführten Übertragungen zu protokollieren. Die publizierten Informationen können sowohl von den Befehlszeilenbefehlen als auch vom IBM MQ Explorer-Plug-in verwendet werden. Sie können IBM MQ-Anwendungen schreiben, die diese Informationen nutzen. Weitere Informationen zu dem Thema, in dem die Informationen veröffentlicht werden, finden Sie unter SYSTEM.FTE Thema.
- Die Schlüsselkomponenten von Managed File Transfer nutzen die Fähigkeit der IBM MQ-Warteschlangenmanager, Nachrichten speichern und weiterleiten zu können. Wenn es also zu einem Ausfall kommt, können unbeeinflusste Teile Ihrer Infrastruktur weiterhin Dateien übertragen. Dies erstreckt sich auch auf den Koordinationswarteschlangenmanager, dem eine Kombination aus Speichern, Weiterleiten und

permanenter Subskription eine Ausfalltoleranz ermöglicht, ohne dass wichtige Informationen zu ausgeführten Dateiübertragungen verloren gehen.

## Übersicht über die MFT-Topologie

Dieser Abschnitt gibt eine Übersicht darüber, wie Managed File Transfer-Agenten mit dem Koordinationswarteschlangenmanager in einem IBM MQ-Netz verbunden sind.

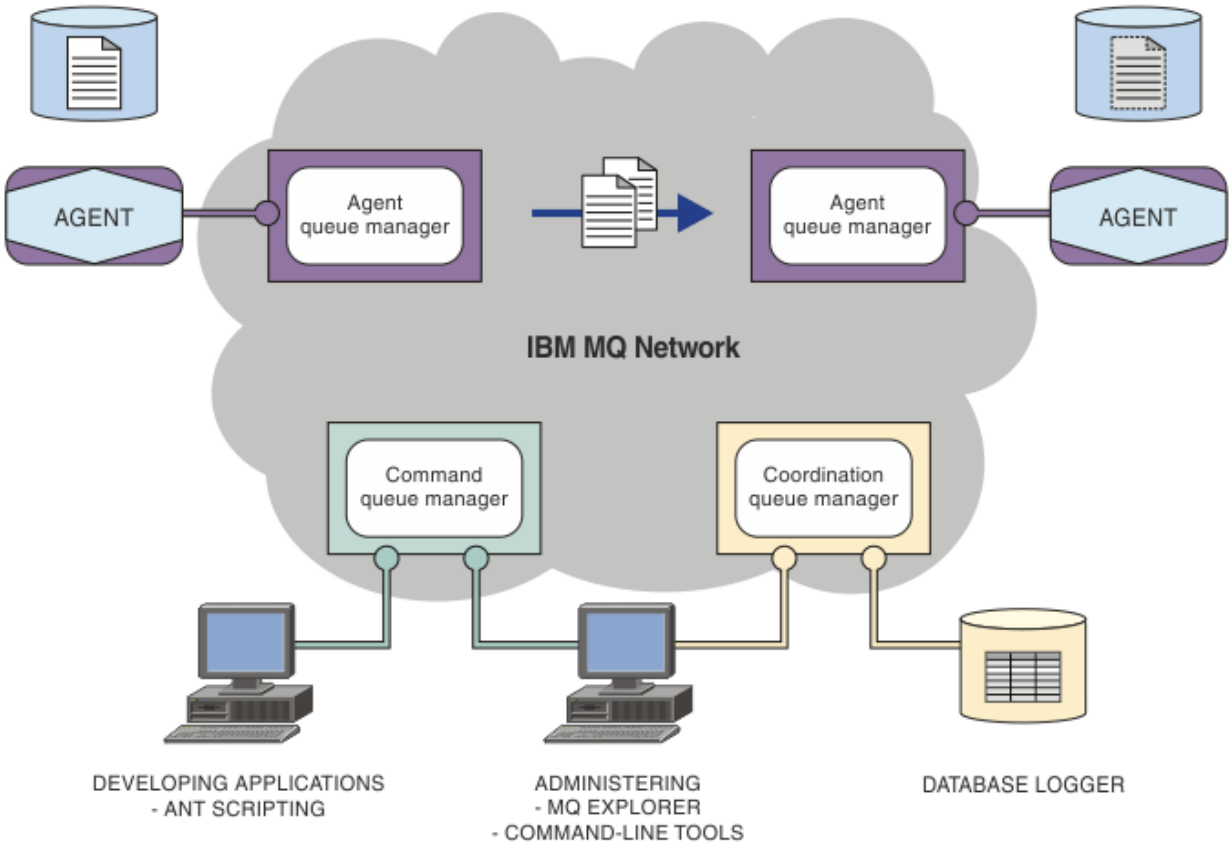
Managed File Transfer-Agenten senden und empfangen die Dateien, die übertragen werden. Jeder Agent verfügt über eine eigene Warteschlangengruppe im zugeordneten Warteschlangenmanager. Der Agent ist mit seinem Warteschlangenmanager entweder in Bindungsmodus oder Clientmodus verbunden. Ein Agent kann auch den Koordinationswarteschlangenmanager als seinen Warteschlangenmanager verwenden.

Der Koordinationswarteschlangenmanager sendet Informationen betreffs Prüfung und Dateiübertragung. Der Koordinationswarteschlangenmanager ist die zentrale Stelle für die Erfassung von Prüfinformationen zu Agenten, zum Übertragungsstatus und zu den Übertragungen. Der Koordinationswarteschlangenmanager muss nicht verfügbar sein, damit eine Übertragung ausgeführt werden kann. Die Übertragungen werden normal fortgesetzt, auch wenn der Koordinationswarteschlangenmanager vorübergehend nicht verfügbar sein sollte. Prüf- und Statusnachrichten werden in diesem Fall in den Agentenwarteschlangenmanagern gespeichert, bis der Koordinationswarteschlangenmanager wieder verfügbar ist, und anschließend wie gewohnt verarbeitet.

Agenten werden beim Koordinationswarteschlangenmanager registriert und publizieren ihre Details auf dem Warteschlangenmanager. Diese Agenteninformationen werden vom Managed File Transfer-Plug-in dazu verwendet, Übertragungen aus IBM MQ Explorer zu starten. Die im Koordinationswarteschlangenmanager erfassten Informationen zum Agenten werden auch von den Befehlen verwendet, um Agenteninformationen und den Agentenstatus anzuzeigen.

Prüfinformationen zum Übertragungsstatus und zur Übertragung werden im Koordinationswarteschlangenmanager veröffentlicht. Mithilfe des Übertragungsstatus und der Prüfinformationen zur Übertragung kann das Managed File Transfer-Plug-in den Fortschritt der Übertragung aus IBM MQ Explorer überwachen. Die im Koordinationswarteschlangenmanager gespeicherten Prüfinformationen zur Übertragung können später zur Überprüfbarkeit herangezogen werden.

Mit dem Befehlswarteschlangenmanager wird eine Verbindung zum IBM MQ-Netz hergestellt. Außerdem wird zu diesem Warteschlangenmanager eine Verbindung hergestellt, wenn Managed File Transfer-Befehle ausgegeben werden.




**Zugehörige Konzepte**

„Managed File Transfer” auf Seite 321
Managed File Transfer überträgt Dateien auf kontrollierte und überprüfbare Weise zwischen Systemen - unabhängig von der Dateigröße und den verwendeten Betriebssystemen.

„Funktionsweise von MFT mit IBM MQ” auf Seite 323
Managed File Transfer interagiert auf verschiedene Weise mit IBM MQ.

Managed File Transfer-Szenario

## MFT REST API - Übersicht

Die REST API unterstützt bestimmte Managed File Transfer-Befehle, einschließlich Auflistung von Übertragungen und Details zu Dateiübertragungsagenten.

Ab IBM MQ 9.1.0 enthält die REST API Optionen zum Auflisten aller aktuellen Managed File Transfer-Übertragungen und zum Abfragen des Status von Managed File Transfer-Agenten. Weitere Informationen finden Sie im Abschnitt Erste Schritte mit der REST API MFT.

# IBM MQ Internet Pass-Thru

IBM MQ Internet Pass-Thru (MQIPT) ist eine optionale Komponente von IBM MQ, mit der Messaging-Lösungen zwischen fernen Sites über das Internet implementiert werden können.

Ab IBM MQ 9.2.0 ist MQIPT eine optionale Komponente von IBM MQ. Wenn Sie die Installationsdateien von MQIPT für IBM MQ 9.3.x abrufen möchten, wechseln Sie zu IBM Fix Central für IBM MQ. Vor IBM MQ 9.2.0 war MQIPT als Support-Pack verfügbar.

Sie müssen IBM MQ 9.3 nicht ausführen, um MQIPT in IBM MQ 9.3verwenden zu können. Mithilfe von MQIPT können Sie jede unterstützte Version von IBM MQ verbinden, und Sie müssen keine anderen IBM MQ-Komponenten mit derselben Version wie MQIPT installieren.

Wenn Sie IBM MQ-Berechtigung erworben haben, können Sie so viele Kopien von MQIPT installieren wie erforderlich. MQIPT-Installationen werden nicht auf Ihre erworbene IBM MQ-Berechtigung angerechnet. Weitere Informationen zur Lizenzierung von IBM MQ finden Sie im Abschnitt IBM MQ - Lizenzinformationen.

**Anmerkung:** Diese Dokumentation bezieht sich auf MQIPT in IBM MQ 9.3. Informationen zur Dokumentation zum MQIPT-Support-Pack (Version 2.1) im IBM Documentation finden Sie unter MQIPT (SupportPac MS81) in der Dokumentation zu IBM MQ 9.0.

**Anmerkung:** Wenn Sie MQIPT 2.1 oder eine frühere Version verwenden, wird Ihnen empfohlen, ein Upgrade auf MQIPT for IBM MQ 9.3durchzuführen, da das Ende des Unterstützungszeitraums für das MQIPT Support Pack 30th September 2020 war.

IBM MQ Internet Pass-Thru wird als eigenständiger Service ausgeführt, der IBM MQ-Nachrichtenflüsse zwischen zwei IBM MQ-Warteschlangenmanagern oder zwischen einem IBM MQ-Client und einem IBM MQ-Warteschlangenmanager empfangen und weiterleiten kann.

Mit MQIPT wird diese Verbindung aktiviert, wenn der Client und der Server sich nicht im gleichen physischen Netz befinden.

Eine oder mehrere Instanzen von MQIPT können im Kommunikationspfad zwischen zwei IBM MQ-Warteschlangenmanagern oder zwischen einem IBM MQ-Client und einem IBM MQ-Warteschlangenmanager angeordnet werden. Mit den Instanzen von MQIPT können die beiden IBM MQ-Systeme Nachrichten austauschen, ohne dass eine direkte TCP/IP-Verbindung zwischen den beiden Systemen erforderlich ist. Dies ist nützlich, wenn die Konfiguration der Firewall eine direkte TCP/IP-Verbindung zwischen den beiden Systemen verhindert.

MQIPT ist an einem oder mehreren TCP/IP-Ports für eingehende Verbindungen empfangsbereit, die normale IBM MQ-Nachrichten, in HTTP getunnelte IBM MQ-Nachrichten oder mit TLS (Transport Layer Security) oder SSL (Secure Sockets Layer) verschlüsselte Nachrichten übertragen können. MQIPT kann mehrere gleichzeitig bestehende Verbindungen verarbeiten.

Der IBM MQ-Kanal, der die ursprüngliche TCP/IP-Verbindung herstellt, wird als *aufrufender Kanal* bezeichnet, der Kanal, zu dem eine Verbindung herstellt werden soll, als *antwortender Kanal* und der Warteschlangenmanager, der schließlich kontaktiert werden soll, als *Zielwarteschlangenmanager*.

MQIPT speichert Daten beim Weiterleiten von der Quelle an das entsprechende Ziel im Hauptspeicher. Es werden keine Daten auf einer Platte gespeichert (außer Speicher, der vom Betriebssystem auf eine Platte ausgelagert wird). MQIPT greift nur explizit auf die Platte zu, um die Konfigurationsdatei zu lesen und um Verbindungsprotokoll- und Traceeinträge zu schreiben.

Sämtliche IBM MQ-Kanaltypen können über eine oder mehrere Instanzen von MQIPT erstellt werden. Der Einsatz von MQIPT im Kommunikationspfad hat keinerlei Auswirkungen auf die funktionalen Merkmale der verbundenen IBM MQ-Komponenten, es können sich allerdings gewisse Auswirkungen auf die Leistung bei der Nachrichtenübertragung ergeben.

MQIPT kann zusammen mit IBM MQ und IBM Integration Bus verwendet werden, wie im Abschnitt „Mögliche Konfigurationen von MQIPT” auf Seite 330 beschrieben.

Informationen zur Installation von MQIPT finden Sie im Abschnitt MQIPT installieren.

**Zugehörige Tasks**

IBM MQ Internet Pass-Thru konfigurieren

IBM MQ Internet Pass-Thru verwalten und konfigurieren

**Zugehörige Verweise**

IBM MQ Internet Pass-Thru -Konfigurationsreferenz

# Verwendung von MQIPT

Es gibt verschiedene Möglichkeiten der Verwendung von IBM MQ Internet Pass-Thru (MQIPT).

## MQIPT kann als Kanalkonzentrator verwendet werden

Durch diese Verwendung von MQIPT können Kanäle zu bzw. von mehreren separaten Hosts von einer Firewall als einziger Kanal zum bzw. vom MQIPT-Host wahrgenommen werden. Dies erleichtert die Definition und Verwaltung der Filterregeln für die Firewall.

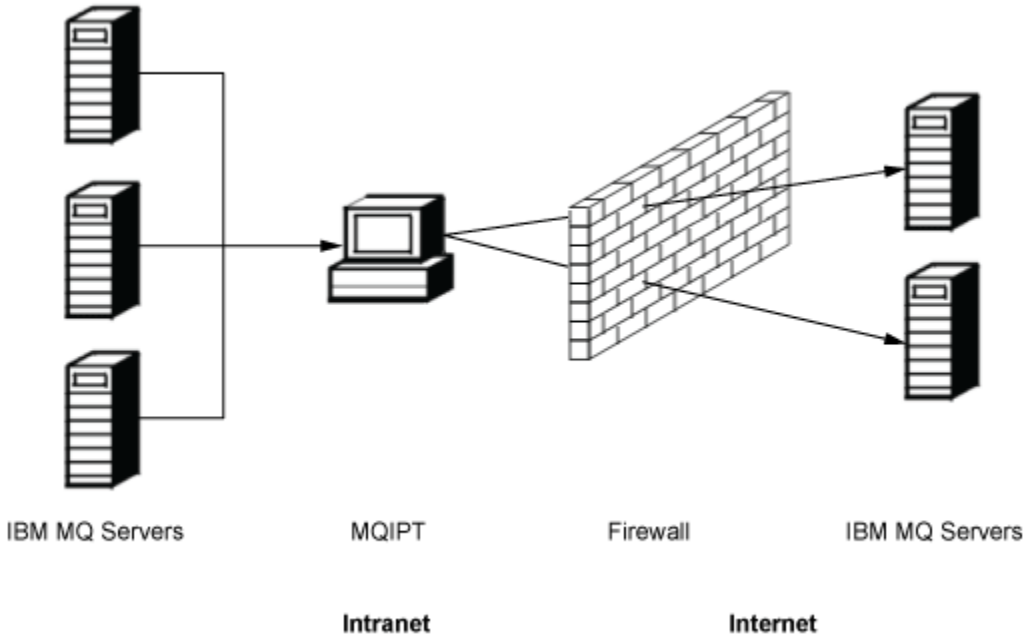



*Abbildung 79. Beispiel für die Verwendung von MQIPT als Kanalkonzentrator*

## MQIPT kann in eine "Demilitarized Zone" (DMZ) gestellt werden, um einen zentralen Zugriffspunkt bereitzustellen

Bei Einsatz von MQIPT auf einem Computer mit einer bekannten und gesicherten IP-Adresse innerhalb einer DMZ-Firewall (eine Firewallkonfiguration für die Sicherung von lokalen Netzen) kann MQIPT zum Überwachen eingehender IBM MQ-Kanalverbindungen verwendet werden, die es dann an das gesicherte Intranet weiterleitet (DMZ = Demilitarized Zone). Die innere Firewall muss diesem vertrauenswürdigen Computer die Herstellung eingehender Verbindungen gestatten. In dieser Konfiguration verhindert MQIPT, dass externe Zugriffsanforderungen die tatsächliche IP-Adresse der Maschinen im gesicherten Intranet erkennen. Auf diese Weise stellt MQIPT einen einzelnen zentraler Zugriffspunkt dar. Bei Bedarf kann MQIPT so konfiguriert werden, dass TLS-Verbindungen akzeptiert werden und Daten über eine separate TLS-Verbindung an das Ziel weitergeleitet werden, wodurch die TLS-Sitzung in der DMZ beendet wird.

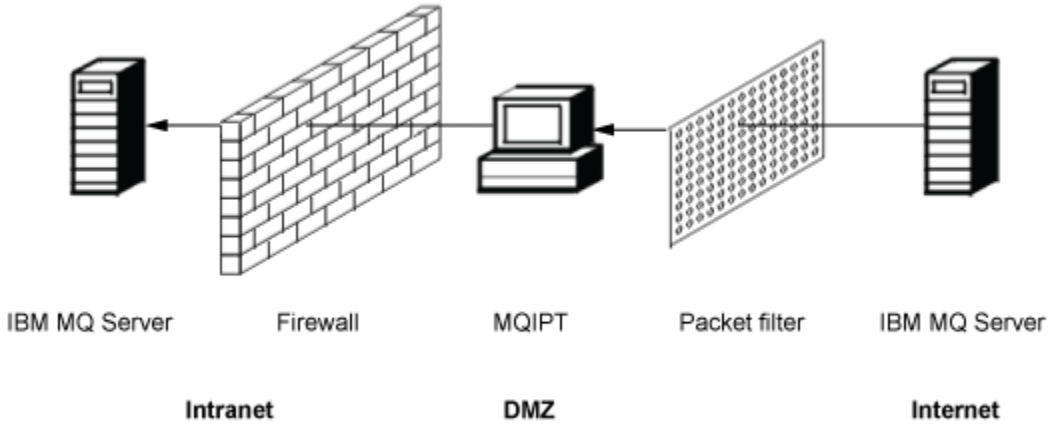



*Abbildung 80. Beispiel für die Verwendung von MQIPT in einer DMZ-Firewall*

## MQIPT kann über die HTTP-Tunnelung kommunizieren

Wenn zwei Instanzen von MQIPT nebeneinander eingesetzt werden, können sie über HTTP miteinander kommunizieren. Durch die HTTP-Tunnelung können Anforderungen unter Verwendung vorhandener HTTP-Proxys durch Firewalls hindurch übergeben werden. Der erste MQIPT fügt das IBM MQ-Protokoll in das HTTP-Protokoll ein und der zweite extrahiert das IBM MQ-Protokoll aus seinem HTTP-Wrapper und leitet es an den Zielwarteschlangenmanager weiter.

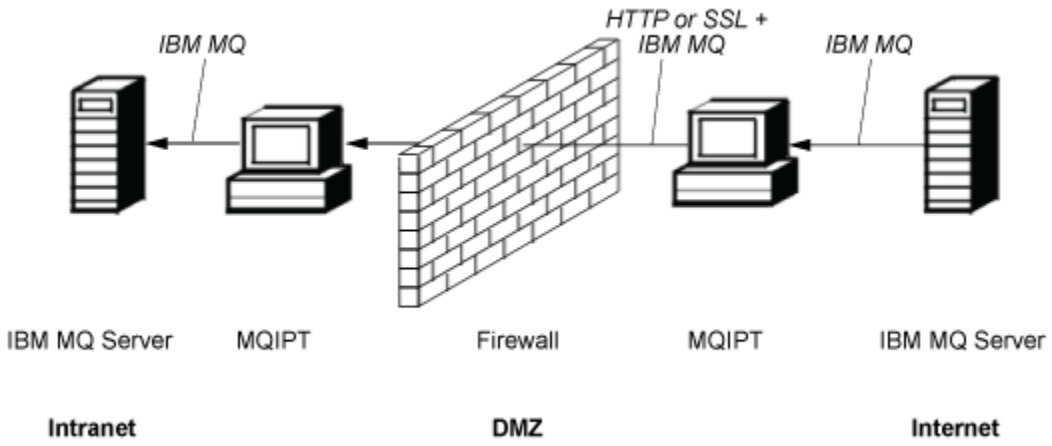



*Abbildung 81. Beispiel für die MQIPT- und HTTP-Tunnelung*

## MQIPT kann Nachrichten verschlüsseln

Wenn MQIPT wie im vorherigen Beispiel konfiguriert ist, können Anforderungen vor der Übertragung durch Firewalls verschlüsselt werden. Die erste Instanz von MQIPT verschlüsselt die Daten, die zweite Instanz entschlüsselt sie mithilfe von SSL/TLS und sendet sie an den Zielwarteschlangenmanager.

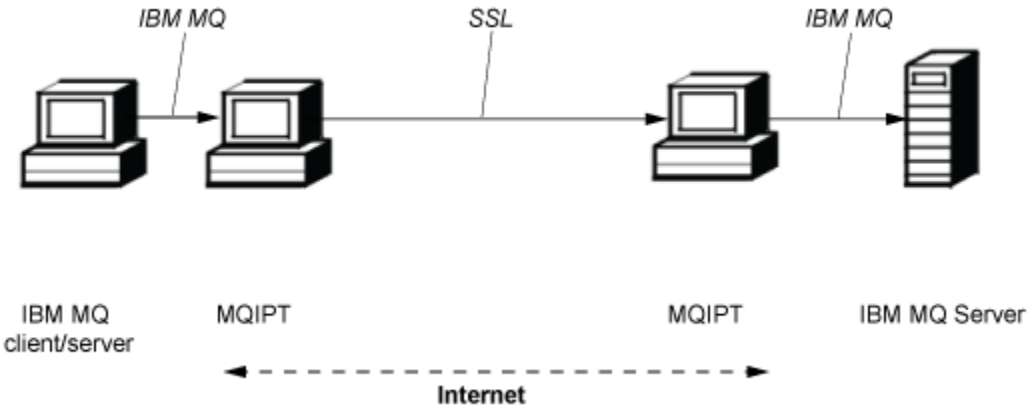



*Abbildung 82. Beispiel für MQIPT und SSL/TLS*

## Funktionsweise von MQIPT

In seiner einfachsten Konfiguration fungiert MQIPT als IBM MQ-Protokollweiterleitungsfunktion. Es ist an einem TCP/IP-Port empfangsbereit und akzeptiert Verbindungsanforderungen von IBM MQ-Kanälen.

Wenn eine korrekt formatierte Anforderung eingeht, stellt MQIPT eine weitere TCP/IP-Verbindung zwischen sich selbst und dem IBM MQ-Zielwarteschlangenmanager her. Anschließend leitet es alle Protokollpakete, die es von seiner eingehenden Verbindung empfängt, an den Zielwarteschlangenmanager weiter und gibt Protokollpakete vom Zielwarteschlangenmanager an die ursprüngliche eingehende Verbindung zurück.

Es liegt keine Änderung am IBM MQ-Protokoll (Client/Server oder Warteschlangenmanager zu Warteschlangenmanager) vor, da keines der beiden Enden direkt über die Anwesenheit des Vermittlers informiert ist. Neue Versionen des IBM MQ-Client- oder -Servercodes sind nicht erforderlich.

Um MQIPT zu verwenden, muss der aufrufende Kanal so konfiguriert sein, dass er den Hostnamen und den Port von MQIPT verwendet und nicht den Hostnamen und Port des Zielwarteschlangenmanagers. Dies wird mit der Eigenschaft **CONNAME** des IBM MQ-Kanals definiert. MQIPT liest die eingehenden Daten und übergibt sie einfach an den Zielwarteschlangenmanager. Andere Konfigurationsfelder, wie z. B. die Benutzer-ID und das Kennwort in einem Client/Server-Kanal, werden ebenfalls an den Zielwarteschlangenmanager übergeben.

### Mehrere Warteschlangenmanager

MQIPT kann verwendet werden, um den Zugriff auf mehr als einen Zielwarteschlangenmanager zu ermöglichen. Damit dies funktioniert, muss ein Mechanismus vorhanden sein, um MQIPT zu informieren, zu welchem Warteschlangenmanager eine Verbindung hergestellt werden soll, sodass MQIPT die eingehende TCP/IP-Portnummer verwendet, um zu ermitteln, zu welchem Warteschlangenmanager eine Verbindung hergestellt werden soll.

Daher können Sie MQIPT so konfigurieren, dass es auf mehreren TCP/IP-Ports empfangsbereit ist. Jeder Empfangsport ist über eine MQIPT-*Route* einem Zielwarteschlangenmanager zugeordnet. Sie können bis zu 100 solche Routen definieren, die einen empfangsbereiten TCP/IP-Port mit dem Hostnamen und dem Port des Zielwarteschlangenmanagers verknüpfen. Somit ist der Hostname (IP-Adresse) des Zielwarteschlangenmanagers für den Ursprungskanal niemals sichtbar. Jede Route kann mehrere Verbindungen zwischen ihrem Empfangsport und dem Ziel verarbeiten, wobei jede Verbindung unabhängig voneinander agiert.

### Konfigurationsdatei MQIPT

MQIPT verwendet eine Konfigurationsdatei mit dem Namen `mqipt.conf`. Diese Datei enthält Definitionen aller Routen und ihrer zugehörigen Eigenschaften. Weitere Informationen zu `mqipt.conf` finden Sie im Abschnitt IBM MQ Internet Pass-Thru verwalten und konfigurieren.

Beim Start von MQIPT wird jede in der Konfigurationsdatei aufgelistete Route gestartet. In die Systemkonsole werden Nachrichten geschrieben, die den Status jeder einzelnen Route anzeigen. Wenn die Nachricht MQCPI078 für eine Route angezeigt wird, ist diese Route bereit, Verbindungsanforderungen zu akzeptieren.

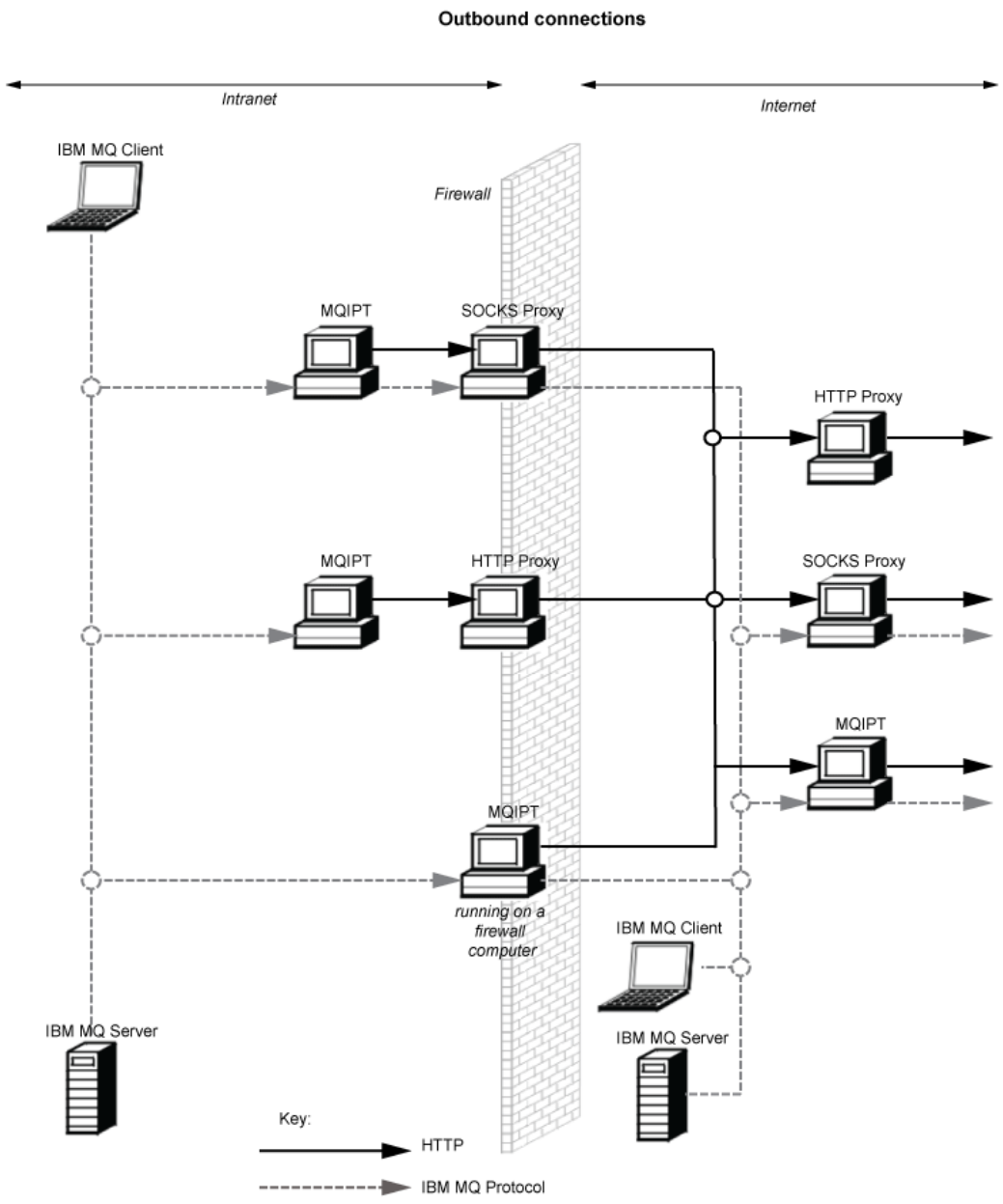
## Mögliche Konfigurationen von MQIPT

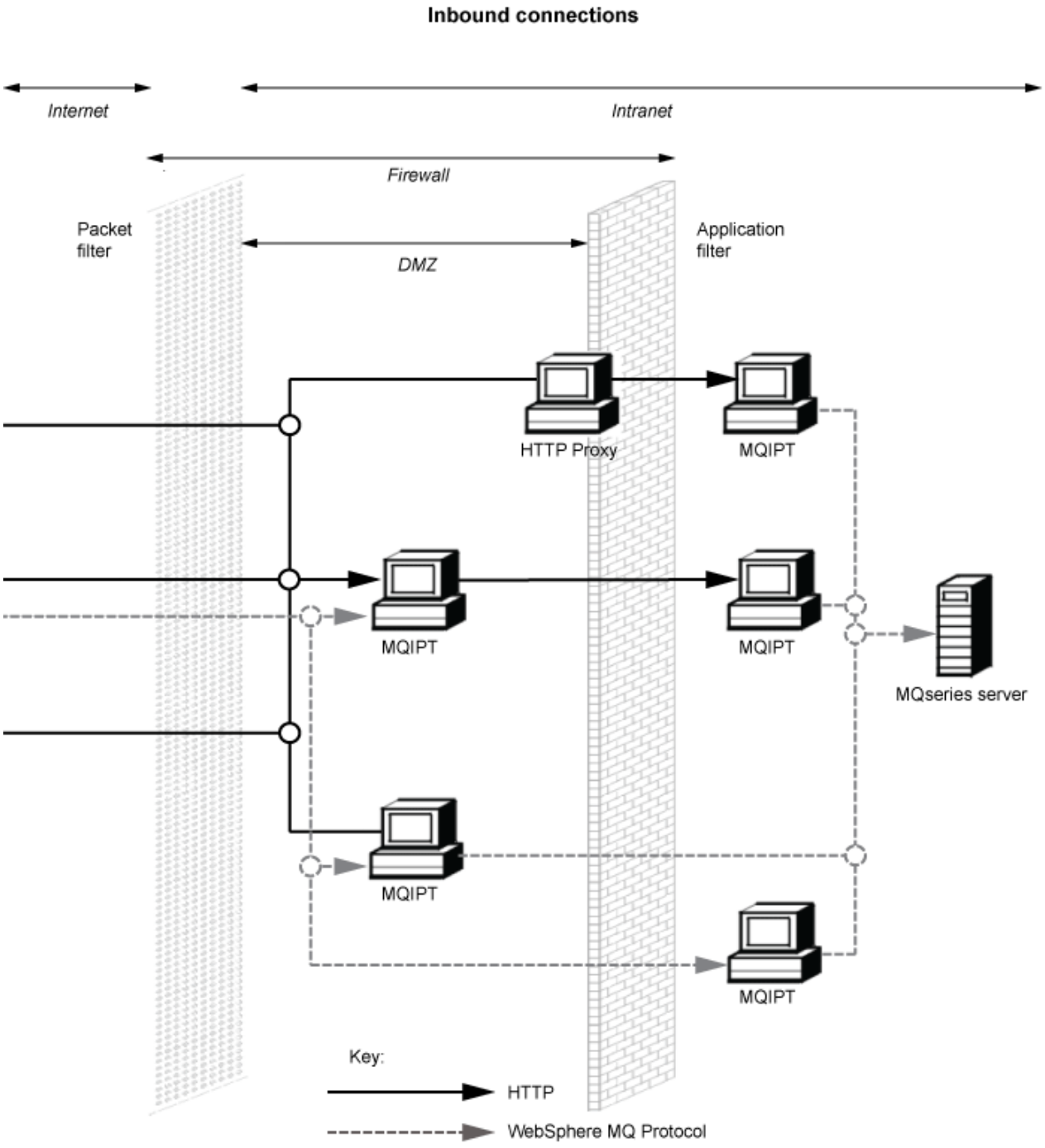
MQIPT kann in Verbindung mit IBM MQ und IBM Integration Busverwendet werden.

Die folgende aus mehreren Teilen bestehende Abbildung zeigt viele mögliche Konfigurationen für MQIPT in einer IBM MQ-Topologie. Sie veranschaulicht verschiedene Möglichkeiten, wie MQIPT Nachrichten senden kann. In der Abbildung sind Clients und Server in einem Intranet innerhalb einer Firewall und im Internet außerhalb der Firewall zu sehen, die Nachrichten an MQIPT, einen HTTP-Proxy oder SOCKS-Proxy übergeben, der diese weiterleitet.

Die Nachrichten werden von einem MQIPT-Proxy oder einem HTTP-Proxy in einer DMZ empfangen, bevor sie über die eingehende Firewall an einen Server übergeben werden.

Beachten Sie, dass der HTTP-Proxy, der SOCKS-Proxy und die MQIPT-Computer auf der Intranetseite der Firewall die Möglichkeit einer Verkettung mehrerer Computer im Internet darstellen. So könnte beispielsweise ein MQIPT-Computer über einen oder mehrere SOCKS- oder HTTP-Proxy-Computer oder weitere MQIPT-Computer kommunizieren, bevor er sein Ziel erreicht.


Outbound connections
Intranet
Internet
IBM MQ Client
Firewall
MQIPT
SOCKS Proxy
HTTP Proxy
MQIPT
HTTP Proxy
SOCKS Proxy
MQIPT
MQIPT
running on a firewall computer
IBM MQ Client
IBM MQ Server
IBM MQ Server
Key:
HTTP
IBM MQ Protocol




## Kompatible Konfigurationen

Kompatible Verbindungsszenarios, bei denen ein IBM MQ-Client oder -Warteschlangenmanager mit MQIPT kommuniziert. Für die Kommunikation mit einem Zielwarteschlangenmanager wird dieselbe oder eine zweite MQIPT-Route verwendet.

## Kompatible Konfigurationen einer einzigen MQIPT-Route

Sie können eine einzelne MQIPT -Route für die Kommunikation mit IBM MQverwenden.

Die Spalten in Tabelle 26 auf Seite 333 enthalten folgende Informationen:

1. Das zwischen IBM MQ und der MQIPT-Route verwendete Protokoll. Die Verbindung kann entweder über einen IBM MQ-Client oder einen Warteschlangenmanager hergestellt werden und kann entweder IBM MQ-Formate und Protokolle (FAP) oder ein SSL/TLS-Protokoll verwenden.
2. Der Modus, in dem die MQIPT-Route ausgeführt wird. Das Format der Kommunikation über das Internet zwischen MQIPT und IBM MQ wird durch die Konfiguration der MQIPT -Route bestimmt. Beachten Sie, dass in den Fällen, in denen die Tabelle SSL erwähnt, auch TLS verwendet werden kann.
3. Das zwischen der MQIPT-Route und dem Zielwarteschlangenmanager verwendete Protokoll.

*Tabelle 26. Gültige Konfigurationen mit einer einzelnen Instanz von MQIPT*

| **1. IBM MQ Quellenprotokoll** | **2. Modus der MQIPT-Route** | **3. IBM MQ Zielprotokoll** |
|---|---|---|
| FAP | FAP-Proxy (Standardwert) | FAP |
| | FAP-Server und SSL-Client | SSL/TLS |
| SSL/TLS | SSL-Proxy | SSL/TLS |
| | SSL-Server und FAP-Client | FAP |
| | SSL-Server und SSL-Client | SSL/TLS |

## Kompatible Konfigurationen mit mehr als einer MQIPT-Route

Möglicherweise wollen Sie auf einer oder mehreren Instanzen von MQIPT mehr als eine Route verwenden, um mit IBM MQ zu kommunizieren.

Die Spalten in Tabelle 27 auf Seite 334 enthalten folgende Informationen:

1. Das zwischen IBM MQ und der ersten MQIPT-Route verwendete Protokoll. Die Verbindung kann entweder über einen IBM MQ-Client oder einen Warteschlangenmanager hergestellt werden und kann entweder IBM MQ-Formate und Protokolle (FAP) oder ein SSL/TLS-Protokoll verwenden.
2. Der Modus, in dem die erste MQIPT-Route ausgeführt wird. Das Format der Kommunikation über das Internet zwischen MQIPT und IBM MQ wird durch die Konfiguration der MQIPT -Route bestimmt. Beachten Sie, dass in den Fällen, in denen die Tabelle SSL erwähnt, auch TLS verwendet werden kann.
3. Der Modus, in dem die zweite MQIPT-Route ausgeführt wird.
4. Das zwischen der zweiten MQIPT-Route und dem Zielwarteschlangenmanager verwendete Protokoll.

| *Tabelle 27. Gültige Konfigurationen mit mehreren Instanzen von MQIPT* | | | |
|---|---|---|---|
| **1. IBM MQ Quellenprotokoll** | **2. Modus der ersten MQIPT-Route** | **3. Modus der zweiten MQIPT-Route** | **4. IBM MQ Zielprotokoll** |
| FAP (Standardwert) | FAP-Proxy (Standardwert) | FAP-Proxy (Standardwert) | FAP |
| | FAP-Server und SSL-Client | SSL-Proxy | SSL/TLS |
| | | SSL-Server und FAP-Client | FAP |
| | | SSL-Server und SSL-Client | SSL/TLS |
| | HTTP-Client | HTTP-Server und SSL-Client | SSL/TLS |
| | HTTPS-Client | HTTPS-Server und SSL-Client | SSL/TLS |
| | HTTP-Client | HTTP-Server | FAP |
| | HTTPS-Client | HTTPS-Server | FAP |
| SSL/TLS | SSL-Proxy | SSL-Proxy | SSL/TLS |
| | | SSL-Server und FAP-Client | FAP |
| | | SSL-Server und SSL-Client | SSL/TLS |
| | HTTP-Client | HTTP-Server | FAP |
| | HTTPS-Client | HTTPS-Server | SSL/TLS |
| | HTTP-Client | HTTP-Server und SSL-Client | FAP |
| | HTTPS-Client | HTTPS-Server und SSL-Client | SSL/TLS |

## Unterstützte Kanalkonfigurationen

Es werden alle IBM MQ-Kanaltypen unterstützt, die Konfiguration ist jedoch auf TCP/IP-Verbindungen beschränkt. Einem IBM MQ-Client oder -Warteschlangenmanager wird MQIPT so angezeigt, als ob es sich um den Zielwarteschlangenmanager handeln würde. Wenn die Kanalkonfiguration einen Zielhost und eine Portnummer erfordert, werden der Hostname und die Listener-Portnummer von MQIPT angegeben.

**Client-/Server-Kanäle**

MQIPT ist für eingehende Clientverbindungsanforderungen empfangsbereit und leitet diese anschließend mithilfe von HTTP-Tunnelung, SSL/TLS oder als IBM MQ-Standardprotokollpakete weiter. Wenn MQIPT HTTP-Tunnelung oder SSL/TLS verwendet, erfolgt die Weiterleitung über eine Verbindung zu einem zweiten MQIPT. Wenn HTTP-Tunnelung nicht verwendet wird, erfolgt die Weiterleitung über eine Verbindung zu dem, was als Zielwarteschlangenmanager angesehen wird (obwohl es sich hierbei wiederum un einen weiteren MQIPT handeln könnte). Wenn der Zielwarteschlangenmanager die Clientverbindung akzeptiert hat, werden Pakete zwischen Client und Server übermittelt.

**Clustersender-/-empfängerkanäle**

Wenn MQIPT eine eingehende Anforderung von einem Clustersenderkanal erhält, wird davon ausgegangen, dass der Warteschlangenmanager SOCKS-fähig ist und die tatsächliche Zieladresse wird während des SOCKS-Handshakeverfahrens ermittelt. Die Anforderung wird genau wie bei Clientver-

bindungskanälen an den nächsten MQIPT oder Zielwarteschlangenmanager weitergeleitet. Hierzu gehören auch automatisch definierte Clustersenderkanäle.

**Sender/Empfänger**
Wenn MQIPT eine eingehende Anforderung von einem Senderkanal erhält, wird diese in genau derselben Weise wie bei Clientverbindungskanälen an den nächsten MQIPT oder Zielwarteschlangenmanager weitergeleitet. Der Zielwarteschlangenmanager überprüft die eingehende Anforderung und startet gegebenenfalls den Empfängerkanal. Die gesamte Kommunikation zwischen Sender-und Empfängerkanal (einschließlich Sicherheitsdatenflüssen) wird übermittelt.

**Requester/Server**
Diese Kombination wird wie die vorherigen Konfigurationen gehandhabt. Die Validierung der Verbindungsanforderung wird vom Serverkanal auf dem Zielwarteschlangenmanager durchgeführt.

**Requester/Sender**
Die "Callback"-Konfiguration könnte hilfreich sein, wenn die beiden Warteschlangenmanager keine direkten Verbindungen zueinander herstellen können, jedoch beide berechtigt sind, eine Verbindung zu MQIPT herzustellen und von dort Verbindungen zu akzeptieren.

**Server/Requester und Server/Empfänger**
Diese werden von MQIPT auf dieselbe Weise behandelt wie die `Sender/Receiver` -Konfiguration.

## Kanalbeendigung und Fehlerbedingungen

Wenn MQIPT das Schließen (entweder normales oder abnormales) eines IBM MQ-Kanals erkennt, leitet es die Kanalschließung weiter. Wenn Sie eine Route mit Hilfe von MQIPT schließen, werden alle über diese Route verlaufenden Kanäle geschlossen.

MQIPT stellt eine optionale Funktion für Inaktivitätszeitlimits bereit. Wenn MQIPT feststellt, dass ein Kanal während eines Zeitraums, der das Zeitlimit überschreitet, inaktiv war, führt er eine sofortige Beendigung der betreffenden beiden Verbindungen durch.

Die IBM MQ-Systeme an jedem Ende des Kanals betrachten diese abnormalen Beendigungsbedingungen entweder als Netzausfälle oder als Beendigung des Kanals durch ihren Partner. Der Kanal kann dann erneut gestartet und wiederhergestellt werden (wenn der Fehler während eines Protokollzeitraums auftritt), als ob MQIPT nicht verwendet würde.

## Sicherheit von Nachrichten

Durch das verteilte Warteschlangenmanagement von IBM MQ wird sichergestellt, dass Nachrichten ordnungsgemäß zugestellt werden. Dies ist auch dann der Fall, wenn MQIPT zwischen den beiden Enden des Kanals vorhanden ist. MQIPT speichert keine Nachrichtendaten und beteiligt sich auch nicht an der Synchronisationspunktprozedur, die eine korrekte Nachrichtenübermittlung sicherstellt.

Bei der Verwendung von schnellen, nicht persistenten IBM MQ-Nachrichten kann eine Nachricht verloren gehen, wenn die MQIPT-Route ausfällt oder erneut gestartet wird, während die IBM MQ-Nachricht unterwegs ist. Bevor Sie die Route erneut starten, müssen Sie sicherstellen, dass alle IBM MQ-Kanäle, die die MQIPT-Route verwenden, inaktiv sind.

Weitere Informationen zur Sicherheit von Nachrichten in IBM MQ finden Sie im Abschnitt Sicherheit von Nachrichten.

## Multi-Instanz-Warteschlangenmanager und Hochverfügbarkeit

MQIPT kann mit Multi-Instanz-Warteschlangenmanagern in Hochverfügbarkeitsumgebungen verwendet werden.

MQIPT hat keinen persistenten Status, daher bietet ein Failover von MQIPT zu einem anderen System keinen Vorteil. Stattdessen sollten mehrere Instanzen von MQIPT mit identischen Konfigurationsdateien `mqipt.conf` auf verschiedenen Systemen ausgeführt werden. Überwachen Sie jede Instanz von MQIPT auf Verfügbarkeit und starten Sie sie gegebenenfalls (auf demselben System) erneut. So wird eine Gruppe identischer MQIPT-Instanzen bereitgestellt, die für die Weiterleitung von Verbindungen verwendet wer-

den können. Sie müssen dann sicherstellen, dass IBM MQ Verbindungen zu MQIPT weiterleiten kann und MQIPT in der Lage ist, diese Verbindungen an den Zielwarteschlangenmanager weiterzuleiten.

IBM MQ-Kanäle für abgehende Nachrichten können auf verschiedene Arten an eine verfügbare MQIPT-Instanz weitergeleitet werden. Beispiele:

- Verwendung einer Lastausgleichsfunktion oder eines Hochverfügbarkeitsrouters wie z. B. des IBM Network Dispatchers aus dem WebSphere Edge Components-Produkt.
- Angabe mehrerer Verbindungsnamen in der IBM MQ-Kanaldefinition in Form einer durch Kommas getrennten Liste. IBM MQ versucht dann, eine Verbindung zu den einzelnen MQIPT-Adressen herzustellen, bis eine verfügbare MQIPT-Instanz gefunden wird.

Sie müssen auch Verbindungen von MQIPT zum Zielwarteschlangenmanager leiten. Wenn die Hochverfügbarkeitskonfiguration für ein Failover der IP-Adresse mit dem Zielwarteschlangenmanager sorgt, ist keine besondere MQIPT-Konfiguration erforderlich: Geben Sie die IP-Zieladresse in der Routeneigenschaft **`Destination`** an und lassen Sie zu, dass die Failover-Operation die IP-Adresse mit dem Warteschlangenmanager verschiebt.

Wenn sich die IP-Adresse des Warteschlangenmanagers jedoch nach einem Failover ändert, müssen Sie dafür sorgen, dass MQIPT die Verbindung an das richtige Ziel weiterleitet. Dies kann auf eine der folgenden Arten erfolgen:

- Schreiben eines Routing-Exits, der prüft, welche IP-Adresse und Portnummer zugänglich ist, und anschließendes Überschreiben des Routenziels für jede Verbindung. Mit MQIPT werden einige Beispielrouting-Exits bereitgestellt; diese können zu diesem Zweck angepasst werden.
- Verwendung einer Hochverfügbarkeits-Lastausgleichsfunktion, um die Verbindung umzuleiten.
- Definition mehrerer MQIPT-Routen (eine für jede IP-Adresse und jeden Port, an der/dem der Warteschlangenmanager ausgeführt werden könnte). Anschließend Leitung der IBM MQ-Verbindungen zu den verschiedenen MQIPT-Routen, beispielsweise durch Auflistung aller Routen-IP-Adressen und Portnummern in Form einer durch Kommas getrennten Liste im Verbindungsnamen des Kanals für abgehende Nachrichten.

Es ist auch wichtig, alle End-to-End-Komponenten im Netzpfad zu optimieren:

1. Verbindungsversuche zu nicht verfügbaren Systemen müssen schnell fehlschlagen, sodass bei Verbindungswiederholungsversuchen zum ersten verfügbaren Ziel übergegangen werden kann.

   Optimieren Sie für MQIPT-SSL-Routen die Routeneigenschaft **`SSLClientConnectTimeout`**, damit bei nicht verfügbaren Zielen umgehend ein Verbindungsfehler ausgegeben wird. Weitere Informationen zu den Optimierungsparametern von IBM MQ finden Sie in der IBM MQ-Dokumentation. Darüber hinaus finden Sie in der Betriebssystemdokumentation Informationen zur TCP/IP-Optimierung für das Betriebssystem. In allen Fällen sollte bei fehlgeschlagenen Verbindungsversuchen schnell ein Netzfehler (z. B. ein TCP-Rücksetzungspaket) zurückgegeben werden oder es sollte ohne unnötige Verzögerung eine Zeitlimitüberschreitung erfolgen.
2. Aktive Verbindungen zu einem fehlgeschlagenen System müssen umgehend getrennt werden, damit neue Verbindungen hergestellt werden können.

   Sie sollten auch bedenken, welche Auswirkungen ein Failover zu einem Zeitpunkt hat, zu dem MQIPT aktiv von Verbindungen genutzt wird. Wahrscheinlich werden Netzverbindungen während eines Failovers getrennt. Für Clientanwendungen können Sie die IBM MQ-Funktion für die automatische Clientverbindungswiederholung verwenden, um getrennte Verbindungen wiederherzustellen. Für Nachrichtenkanäle können Sie ein kurzes Wiederholungsintervall angeben, sodass der Kanal sofort wieder eine Verbindung herstellt. Weitere Informationen zur Konfiguration der automatischen Clientverbindungswiederholung und der Nachrichtenkanalwiederholung finden Sie in der IBM MQ-Dokumentation.

## V 9.3.5 IBM MQ Console und REST API

Sie können die IBM MQ Console und REST API verwenden, um IBM MQzu verwalten und Messaging-Operationen mithilfe von HTTPauszuführen.

- Sie können die IBM MQ Console verwenden, um grundlegende Verwaltungstasks über einen Web-Browser auszuführen. Weitere Informationen hierzu finden Sie im Abschnitt Verwaltung mit dem IBM MQ Console .
- Sie können die administrative REST API verwenden, um IBM MQ-Objekte zu verwalten, wie z. B. Warteschlangenmanager und Warteschlangen sowie Managed File Transfer-Agenten und -Übertragungen. Weitere Informationen finden Sie im Abschnitt Verwaltung mit der REST API.
- Sie können messaging REST API verwenden, um einfaches Punkt-zu-Punkt-und Publish-Messaging durchzuführen. Weitere Informationen finden Sie unter Messaging mit REST API.

## Installationsoptionen

Die IBM MQ Console und REST API werden auf einem WebSphere Liberty -Server mit dem Namen mqweb ausgeführt. Ab IBM MQ 9.3.5können Sie den mqweb-Server als optionale Komponente in einer IBM MQ -Installation oder als eigenständige IBM MQ Web Server -Installation installieren.

V 9.3.5 Linux **Eigenständige Installation von IBM MQ Web Server**

Ab IBM MQ 9.3.5kann der mqweb-Server in einer eigenständigen Installation von IBM MQ Web Serverausgeführt werden. Mit einer eigenständigen IBM MQ Web Server -Installation können Sie den mqweb-Server auf Systemen installieren und ausführen, die von Ihren IBM MQ -Installationen getrennt sind. Die Installation eines eigenständigen IBM MQ Web Server s bietet größere Flexibilität hinsichtlich der Systeme und der Anzahl der Systeme, auf denen Sie Ihre mqweb-Server ausführen möchten. Bei Bedarf können mehrere Instanzen des mqweb-Server auf verschiedenen Maschinen ausgeführt werden, um die erforderliche Skalierbarkeit und Verfügbarkeit bereitzustellen.

Wenn Sie eine IBM MQ -Berechtigung erworben haben, können Sie so viele Kopien installieren, wie für den eigenständigen IBM MQ Web Servererforderlich sind. IBM MQ Web Server-Installationen werden nicht auf Ihre erworbene IBM MQ-Berechtigung angerechnet. Weitere Informationen zur Lizenzierung von IBM MQ finden Sie im Abschnitt IBM MQ - Lizenzinformationen.

In einer eigenständigen IBM MQ Web Server -Installation gelten die folgenden Einschränkungen:

- Mit IBM MQ Console können nur ferne Warteschlangenmanager verwaltet werden.
- messaging REST API kann nur mit fernen Warteschlangenmanagern verwendet werden.
- Die administrative REST API ist nicht verfügbar.

Der eigenständige IBM MQ Web Server wird nur auf Linux -Plattformen unterstützt.

Weitere Informationen zur Installation des eigenständigen IBM MQ Web Servers finden Sie unter Eigenständigen IBM MQ Web Serverinstallieren.

**Optionale Komponente einer IBM MQ -Installation**

Sie können die Komponenten IBM MQ Console und REST API im Rahmen einer IBM MQ -Installation installieren.

Alle IBM MQ Console -und REST API -Funktionen sind verfügbar, wenn der mqweb-Server in einer IBM MQ -Installation ausgeführt wird.

- Mit IBM MQ Console können lokale und ferne Warteschlangenmanager verwaltet werden.
- messaging REST API kann mit lokalen und fernen Warteschlangenmanagern verwendet werden.
- Mit administrative REST API können lokale und ferne Warteschlangenmanager verwaltet werden.

Wenn Sie die Komponenten IBM MQ Console und REST API verwenden möchten, installieren Sie die folgende Komponente als Teil Ihrer IBM MQ -Installation:

- AIX Installieren Sie unter AIX die Dateigruppe `mqm.web.rte`.
- IBM i Installieren Sie unter IBM idie WEB-Komponente.
- Linux Installieren Sie unter Linux die Komponente `MQSeriesWeb`.

- Windows Installieren Sie unter Windows das Feature `Web Administration`.
- z/OS Installieren Sie unter z/OS das Feature `IBM MQ for z/OS UNIX System Services Web Components`.

# Bemerkungen

Die vorliegenden Informationen wurden für Produkte und Services entwickelt, die auf dem deutschen Markt angeboten werden.

Möglicherweise bietet IBM die in diesem Dokument beschriebenen Produkte, Services oder Funktionen in anderen Ländern nicht an. Informationen über die gegenwärtig im jeweiligen Land verfügbaren Produkte und Services sind beim zuständigen IBM Ansprechpartner erhältlich. Hinweise auf Produkte, Programme oder Services von IBM bedeuten nicht, dass nur Produkte, Programme oder Services von IBM verwendet werden können. Anstelle der IBM Produkte, Programme oder Services können auch andere, ihnen äquivalente Produkte, Programme oder Services verwendet werden, solange diese keine gewerblichen oder andere Schutzrechte der IBM verletzen. Die Verantwortung für den Betrieb von Fremdprodukten, Fremdprogrammen und Fremdservices liegt beim Kunden.

Für in diesem Handbuch beschriebene Erzeugnisse und Verfahren kann es IBM Patente oder Patentanmeldungen geben. Mit der Auslieferung dieses Handbuchs ist keine Lizenzierung dieser Patente verbunden. Lizenzanforderungen sind schriftlich an folgende Adresse zu richten (Anfragen an diese Adresse müssen auf Englisch formuliert werden):

IBM Director of Licensing
IBM Europe, Middle East & Africa
Tour Descartes
2, avenue Gambetta
92066 Paris La Défense
U.S.A.

Bei Lizenzanforderungen zu Double-Byte-Information (DBCS) wenden Sie sich bitte an die IBM Abteilung für geistiges Eigentum in Ihrem Land oder senden Sie Anfragen schriftlich an folgende Adresse:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Trotz sorgfältiger Bearbeitung können technische Ungenauigkeiten oder Druckfehler in dieser Veröffentlichung nicht ausgeschlossen werden. Die hier enthaltenen Informationen werden in regelmäßigen Zeitabständen aktualisiert und als Neuausgabe veröffentlicht. IBM kann ohne weitere Mitteilung jederzeit Verbesserungen und/oder Änderungen an den in dieser Veröffentlichung beschriebenen Produkten und/oder Programmen vornehmen.

Verweise in diesen Informationen auf Websites anderer Anbieter werden lediglich als Service für den Kunden bereitgestellt und stellen keinerlei Billigung des Inhalts dieser Websites dar. Das über diese Websites verfügbare Material ist nicht Bestandteil des Materials für dieses IBM Produkt.

Werden an IBM Informationen eingesandt, können diese beliebig verwendet werden, ohne dass eine Verpflichtung gegenüber dem Einsender entsteht.

Lizenznehmer des Programms, die Informationen zu diesem Produkt wünschen mit der Zielsetzung: (i) den Austausch von Informationen zwischen unabhängigen, erstellten Programmen und anderen Programmen (einschließlich des vorliegenden Programms) sowie (ii) die gemeinsame Nutzung der ausgetauschten Informationen zu ermöglichen, wenden sich an folgende Adresse:

IBM Europe, Middle East & Africa
Software Interoperability Coordinator, Department 49XA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

Die Bereitstellung dieser Informationen kann unter Umständen von bestimmten Bedingungen - in einigen Fällen auch von der Zahlung einer Gebühr - abhängig sein.

Die Lieferung des in diesen Informationen beschriebenen Lizenzprogramms sowie des zugehörigen Lizenzmaterials erfolgt auf der Basis der IBM Rahmenvereinbarung bzw. der Allgemeinen Geschäftsbedingungen von IBM, der IBM Internationalen Nutzungsbedingungen für Programmpakete oder einer äquivalenten Vereinbarung.

Alle in diesem Dokument enthaltenen Leistungsdaten stammen aus einer kontrollierten Umgebung. Die Ergebnisse, die in anderen Betriebsumgebungen erzielt werden, können daher erheblich von den hier erzielten Ergebnissen abweichen. Einige Daten stammen möglicherweise von Systemen, deren Entwicklung noch nicht abgeschlossen ist. Eine Gewährleistung, dass diese Daten auch in allgemein verfügbaren Systemen erzielt werden, kann nicht gegeben werden. Darüber hinaus wurden einige Daten unter Umständen durch Extrapolation berechnet. Die tatsächlichen Ergebnisse können davon abweichen. Benutzer dieses Dokuments sollten die entsprechenden Daten in ihrer spezifischen Umgebung prüfen.

Alle Informationen zu Produkten anderer Anbieter stammen von den Anbietern der aufgeführten Produkte, deren veröffentlichten Ankündigungen oder anderen allgemein verfügbaren Quellen. IBM hat diese Produkte nicht getestet und kann daher keine Aussagen zu Leistung, Kompatibilität oder anderen Merkmalen machen. Fragen zu den Leistungsmerkmalen von Produkten anderer Anbieter sind an den jeweiligen Anbieter zu richten.

Aussagen über Pläne und Absichten von IBM unterliegen Änderungen oder können zurückgenommen werden und repräsentieren nur die Ziele von IBM.

Diese Veröffentlichung enthält Beispiele für Daten und Berichte des alltäglichen Geschäftsablaufes. Sie sollen nur die Funktionen des Lizenzprogramms illustrieren und können Namen von Personen, Firmen, Marken oder Produkten enthalten. Sämtliche dieser Namen sind fiktiv. Ähnlichkeiten mit Namen und Adressen tatsächlicher Unternehmen oder Personen sind zufällig.

COPYRIGHTLIZENZ:

Diese Veröffentlichung enthält Beispielanwendungsprogramme, die in Quellensprache geschrieben sind und Programmiertechniken in verschiedenen Betriebsumgebungen veranschaulichen. Sie dürfen diese Beispielprogramme kostenlos ohne Zahlung an IBM in jeder Form kopieren, ändern und verteilen, wenn dies zu dem Zweck geschieht, Anwendungsprogramme zu entwickeln, zu verwenden, zu vermarkten oder zu verteilen, die mit der Anwendungsprogrammierschnittstelle für die Betriebsumgebung konform sind, für die diese Beispielprogramme geschrieben sind. Diese Beispiele wurden nicht unter allen denkbaren Bedingungen getestet. Daher kann IBM die Zuverlässigkeit, Wartungsfreundlichkeit oder Funktion dieser Programme weder zusagen noch gewährleisten.

Wird dieses Buch als Softcopy (Book) angezeigt, erscheinen keine Fotografien oder Farbabbildungen.

## Informationen zu Programmierschnittstellen

Die bereitgestellten Informationen zur Programmierschnittstelle sollen Sie bei der Erstellung von Anwendungssoftware für dieses Programm unterstützen.

Dieses Handbuch enthält Informationen über vorgesehene Programmierschnittstellen, die es dem Kunden ermöglichen, Programme zu schreiben, um die Services von WebSphere MQzu erhalten.

Diese Informationen können jedoch auch Angaben über Diagnose, Bearbeitung und Optimierung enthalten. Die Informationen zu Diagnose, Bearbeitung und Optimierung sollten Ihnen bei der Fehlerbehebung für die Anwendungssoftware helfen.

**Wichtig:** Verwenden Sie diese Diagnose-, Änderungs-und Optimierungsinformationen nicht als Programmierschnittstelle, da sie Änderungen unterliegen.

# Marken

IBM, das IBM Logo, ibm.com, sind Marken der IBM Corporation in den USA und/oder anderen Ländern. Eine aktuelle Liste der IBM Marken finden Sie auf der Webseite "Copyright and trademark information"www.ibm.com/legal/copytrade.shtml. Weitere Produkt- und Servicenamen können Marken von IBM oder anderen Unternehmen sein.

Microsoft und Windows sind eingetragene Marken der Microsoft Corporation in den USA und/oder anderen Ländern.

UNIX ist eine eingetragene Marke von The Open Group in den USA und anderen Ländern.

Linux ist eine eingetragene Marke von Linus Torvalds in den USA und/oder anderen Ländern.

Dieses Produkt enthält Software, die von Eclipse Project (https://www.eclipse.org/) entwickelt wurde.

Java und alle auf Java basierenden Marken und Logos sind Marken oder eingetragene Marken der Oracle Corporation und/oder ihrer verbundenen Unternehmen.

Teilenummer:

(1P) P/N: