8.0

IBM MQ 概述

注

在使用本资料及其支持的产品之前，请阅读第 487 页的『声明』中的信息。

本版本适用于 IBM® MQ V 8 发行版 0 以及所有后续发行版和修订版，直到在新版本中另有声明为止。

当您向 IBM 发送信息时，授予 IBM 以它认为适当的任何方式使用或分发信息的非独占权利，而无需对您承担任何责任。

© Copyright International Business Machines Corporation 2007, 2023.

内容

产品概述 7
IBM MQ 简介 8
IBM MQ 许可证信息 10
IBM MQ 8.0.0 中的新增功能 13
新的系列功能 14
分布式平台的新增信息 18
z/OS 平台的新增功能 19
IBM MQ 8.0.0 中更改的内容 22
API: 更改为 API 出口的操作 26
在 SVRCONN 通道上更改为 STOPPED 状态 26
命令更改 :crtmqenv 命令 26
命令更改: **dmpmqcfg** 输出 26
命令更改: 响应来自 **runmqsc** 的 DISPLAY CONN 命令时显示的结果 26
命令更改 :runmqsc 命令 27
命令更改: 除去 migmbbrk 和 z/OS CSQUMGMB 27
命令级别: 更改 27
不推荐: 与 libmqmcs 和 libmqmzse 库链接 28
不推荐: Managed File Transfer Web 网关不推荐 28
不推荐: SSLv3 协议 28
不推荐: 较弱的密码算法 29
不推荐: IBM MQ 资源管理器服务定义向导 29
不推荐: 针对 SOAP 客户机和 HTTP 网桥的 IBM MQ 传输 29
多实例队列管理器的扩展启动事件 30
IBM MQ 8.0 的 IBM i: 更改 30
Java 和 JMS: 对字符串转换的更改 30
Java 和 JMS: 对 CipherSuite 支持的更改 30
Java 和 JMS: 资源适配器更改 31
Java: Java 7 的更改 31
Java: 对 IBM MQ classes for Java 的更改 31
Java: 对 Java 跟踪行为的更改 32
Java: 对 MQException 和 MQDataException 类日志记录的更改 32
JMS: 对 "字节" 和 "流" 消息状态的更改 32
JMS: 版本 8.0 中的异常侦听器更改 33
JMS: 对 IBM MQ classes for JMS 的更改 33
JMS: 对 JMS 2.0 34
JMS: 对 **PROVIDERVERSION** 属性的更改 34
对集群主题行为的更改 34
发布/预订: 对 DISPLAY TOPIC 和 DISPLAY TSTATUS 输出的更改 35
发布/预订: 对预订选择字符串求值方式的更改 35
发布/预订: 使用 PROXYSUB (FORCE) 对代理预订传播进行的更改 35
发布/预订: 尝试更改时未将代理预订修改为 ADMIN 36
队列管理器: 更改为缺省 TCP 缓冲区大小 36
队列管理器: 为集群中的队列管理器添加的版本属性 36
安全性: 证书撤销检查 36
安全性: 对 CipherSuite 支持的更改 36
安全性: 新的 CONNAUTH CHCKLOCL 参数 37
安全性: UNIX 和 Linux 系统上基于 OAM 用户的许可权 37
安全性: 在客户机上禁用 IBM MQ AMS 38
安全性: 客户机应用程序上的密码保护 39
安全性: 减少 SSL 和 TLS 的通道发送出口缓冲区空间 39
Solaris 客户机: 安装方法中的更改 39

遥测: 对 IBM MQ Telemetry Client Software Development Kit (SDK) 的更改 40
IBM MQ 8.0 的 Windows: 更改 40
z/OS: 更改为可用 CipherSpecs 43
z/OS: 更改为在错误消息插入中显示映射的 IP 地址 43
z/OS: 更改为 MAKECLNT 43
z/OS: 对日志 RBA 和 URID 长度的更改 43
对 z/OS， 版本 8.0 的 IBM MQ Managed File Transfer 的更改 43
z/OS: 通道启动程序 SMF 记录 44
不再支持 z/OS: WLM/DNS 44
z/OS: 对 DISPLAY 命令输出的更改 45
IBM MQ 8.0.0 的新消息，已更改的消息和已除去的消息 45
版本 8.0.0 的新增和已更改的 z/OS 消息 45
IBM MQ 8.0 修订包中新增和更改的内容 46
V 8.0.0FP15 中新增及更改的内容 50
V 8.0.0FP14 中新增及更改的内容 51
V 8.0.0FP12 中新增及更改的内容 52
V 8.0.0FP11 中新增及更改的内容 53
V 8.0.0FP10 中新增及更改的内容 54
V 8.0.0 修订包 9 中新增及更改的内容 54
V 8.0.0 修订包 8 中新增及更改的内容 55
V 8.0.0 修订包 7 中新增及更改的内容 57
V 8.0.0 修订包 6 中新增及更改的内容 58
V 8.0.0 修订包 5 中新增及更改的内容 60
V 8.0.0 修订包 4 中新增及更改的内容 61
V 8.0.0 修订包 3 中新增及更改的内容 64
V 8.0.0 修订包 2 中新增及更改的内容 65
版本 8.0 修订包的新增，已更改和已除去消息 67
先前版本中新增和更改的内容 77
声明 78
IBM MQ 的辅助功能选项 80
产品文档中的发行版和平台图标 80
IBM MQ 8.0 快速入门指南 81
IBM MQ 8.0 信息指南 83
IBM Documentation Offline 应用程序中的 IBM MQ 8.0 85
产品文档和程序目录的 IBM MQ 8.0 PDF 文件 85
术语表 87
A 87
B 90
C 90
D 94
E 96
F 97
G 98
H 99
I 100
J 101
K 102
L 102
M 103
N 107
O 108
P 109
Q 111
R 112
S 115
T 119
U 121
V 122

W....122
X....123
技术概述....125
消息排队简介....125
消息排队的主要功能和优点....126
消息排队术语....128
消息和队列....130
IBM MQ 对象....131
对象类型....132
对 IBM MQ 对象进行命名....146
对象属性....151
队列共享组....151
系统缺省对象....152
分布式排队和集群....152
分布式排队组件....155
集群组件....164
发布/预订消息传递....168
发布/预订组件....169
单个队列管理器发布/预订配置的示例....190
分布式发布/预订网络....190
IBM MQ 多点广播....205
初始多点广播概念....205
IBM MQ Telemetry....206
IBM MQ Telemetry 简介....207
遥测用例....209
将遥测设备连接至队列管理器....216
遥测连接协议....218
遥测 (MQXR) 服务....219
遥测通道....221
MQTT protocol....223
MQTT Client....224
向 MQTT 客户机发送消息....226
从 MQTT 客户机向 IBM MQ 应用程序发送消息....237
在设备的 MQTT 守护程序与 IBM MQ 之间传输消息....239
MQTT 发布/预订应用程序....242
遥测应用程序....243
IBM MQ Telemetry 与队列管理器的集成....245
设备的 MQTT 守护程序....249
MQTT 无状态和有状态会话....251
未连接 MQTT 客户机时....252
MQTT 客户机与 IBM MQ 应用程序之间的松耦合....254
IBM MQ Telemetry 安全性....256
IBM MQ Telemetry 全球化....257
IBM MQ Telemetry 的性能和可伸缩性....259
IBM MQ Telemetry 支持的设备....262
安全性....264
客户机和服务器....264
IBM MQ MQI clients 概述....265
事务管理和支持....270
扩展队列管理器设施....271
IBM MQ Java 语言接口....272
IBM MQ classes for JMS....272
IBM MQ 消息传递提供者方式....279
IBM MQ.NET 受管客户机的 SSL 和 TLS 支持....280
HP Integrity NonStop Server 的 IBM MQ 客户机....281
IBM MQ Client for HP Integrity NonStop Server 技术概述....284
IBM MQ Client for HP Integrity NonStop Server 支持的环境和功能部件....284

IBM MQ for z/OS 概念 285
z/OS 上的队列管理器 286
z/OS 上的通道启动程序 287
术语和任务 289
共享队列和队列共享组 291
组内排队 328
存储管理 339
日志记录 343
定义系统 352
恢复和重新启动 361
z/OS 上的安全概念 374
可用性 379
监控和统计信息 382
恢复单元处置 383
IBM MQ 和其他 z/OS 产品 385
IBM MQ 和 CICS 385
IBM MQ 和 IMS 386
IBM MQ 和 z/OS 批处理， TSO 和 RRS 适配器 389
IBM MQ for z/OS 和 WebSphere Application Server 390
方案 393
IBM MQ 入门 393
规划解决方案 393
实现解决方案 395
后续操作 406
点到点方案 407
规划解决方案 407
实现解决方案 408
保护点到点拓扑 414
发布/预订方案 416
发布/预订集群方案 416
发布/预订层次结构方案 422
业务支持方案 432
介绍工作单元 433
方案 1: 队列管理器执行协调 433
方案 2: 其他软件提供协调 454
即将到期的全球工作单元 460
恢复单元处置 461
安全方案 461
安全方案: z/OS 上的两个队列管理器 461
安全方案: z/OS 上的队列共享组 468
使用 SSL 或 TLS 连接两个队列管理器 473
安全地将客户机连接到队列管理器 479
文件传输方案 485
声明 487
编程接口信息 488
商标 488

关于 IBM MQ

用于帮助您开始使用 IBM MQ 8.0 的介绍性信息，包括产品简介以及本发行版的新增内容和已更改内容的概述。

相关信息

IBM Professional Certification Program
产品支持
系统需求
支持技术说明
IBM 支持门户网站
IBM Redbooks 出版物
IBM Developer
全局 WebSphere 社区
MQSeries.net

IBM MQ

欢迎使用 IBM MQ 8.0 产品文档，您可以在其中找到有关如何完成创建和维护 IBM MQ 环境所需的任务的详细指示信息。 该文档还包含了一些概念信息，有助于您了解该产品以及如何使用该产品来解决业务问题。

IBM MQ 是一款稳定、安全又可靠的消息传递中间件。 它使用消息和队列来支持应用程序、系统、服务和文件之间的信息交换。 它可以简化和加速多个平台中不同应用程序和业务数据的集成。 支持各种 API 和语言，并可以在本地、云和混合云中部署。

相关信息

产品概述
技术概述
方案
规划
迁移和升级
安装
安全性
配置
管理
开发应用程序
监控和性能
故障诊断和支持
参考
IBM MQ Managed File Transfer
IBM MQ 资源管理器
IBM MQ GUI 向导帮助

© Copyright IBM Corp. 2007, 2023

IBM MQ Internet Pass-Thru (SupportPac MS81)
IBM MQ 虚拟系统模式类型
Message Service Client for .NET
IBM MQ for HPE NonStop V8
IBM MQ Appliance
IBM MQ on Cloud

IBM MQ 简介

您可以使用 IBM MQ 使应用程序能够在不同时间和许多不同的计算环境中进行通信。

什么是 IBM MQ?

IBM MQ 可将任何类型的数据作为消息进行传递，可使企业构建灵活的可复用体系结构，如面向服务的体系结构 (SOA) 环境。 它与广泛的计算平台，应用程序， Web Service 和通信协议协同工作，以实现高度安全的消息传递。 IBM MQ 提供了一个通信层，用于查看和控制组织内部和外部的消息和数据流。

IBM MQ 提供:

- 从大型机到移动设备的通用消息传递集成，为动态异构环境提供单一，强大的消息传递主干。
- 具有高度安全的功能的消息传递，可生成可审计的结果。
- 高性能的消息传输，以提高速度和可靠性来交付数据。
- 管理功能，可简化消息传递管理并减少使用复杂工具所花费的时间。
- 开放式标准开发工具，支持可扩展性和业务增长。

应用程序可选择编程接口和编程语言以连接到 IBM MQ。

IBM MQ 是 消息传递 和 排队 中间件，具有多种操作方式: 点到点 ; *publish/subscribe* ; *file transfer* 。 应用程序可以通过 多点广播将消息发布到多个订户。

消息传递
程序通过在消息中相互发送数据而不是直接相互调用来进行通信。

排队
将消息放置在队列上，以便程序可以以不同的速度和时间在不同的位置独立运行，并且在它们之间没有直接连接。

点到点
应用程序将消息发送到队列或队列列表。 发送方必须知道目标的名称，但不能知道它在何处。

发布/预订
应用程序发布有关主题的消息，例如团队进行的游戏的结果。 IBM MQ 将消息副本发送到预订结果主题的应用程序。 他们用球队打出的比赛结果来接收消息。 发布者不知道订户的名称或他们的位置。

多点广播
多点广播是一种高效的发布/预订消息传递形式，可扩展至许多订户。 它将向每个订户发送发布的副本的工作从 IBM MQ 转移到网络。 一旦在发布者和订户之间建立了发布的路径， IBM MQ 就不会参与转发发布。

文件传输
文件在消息中传输。 IBM MQ Managed File Transfer 管理文件传输和管理以设置自动传输并记录结果。 您可以将文件传输与其他文件传输系统， IBM MQ 消息传递和 Web 进行集成。

遥测
IBM MQ Telemetry 提供了一个 IBM MQ Telemetry Transport (MQTT) 服务器，用于将 IBM MQ 环境扩展至 “Internet of Things” -即，位于网络边缘的众多移动， Web 和 M2M 设备和应用程序。 通过 IBM MQ Telemetry ，后台系统可以利用并响应来自这些设备和应用程序的消息。 为了最大限度减少对现有消息传递环境的干扰，遥测服务器可以跨其他协议以及 MQTT 进行通信。 例如，可以将服务器配置为在外部对话 MQTT ，在内部对话 JMS 。

它具有什么功能?

IBM MQ 提供了通用消息传递主干，该主干具有强大的连接性，用于应用程序的灵活可靠消息传递以及使用面向服务的体系结构 (SOA) 集成现有 IT 资产。

- IBM MQ 在应用程序之间以及通过网络发送和接收数据。
- 消息传递 有保证 ，并且与应用程序 解耦 。 有保证，因为 IBM MQ 以事务方式交换消息，并将其解耦，因为应用程序不必检查它们发送的消息是否安全传递。
- 您可以使用 SSL/TLS 保护队列管理器之间的消息传递。
- 借助 Advanced Message Security (AMS) ，您可以在由一个应用程序放置和由另一个应用程序检索之间对消息进行加密和签名。
- 应用程序员不需要具备通信编程知识。

如何使用?

有几种不同的方法可以使用 IBM MQ，例如:

- 使用 MQ Explorer GUI 或通过从命令窗口或应用程序运行命令来创建和管理 IBM MQ 。
- 通过调用其中一个编程接口来发送和接收消息的程序应用程序。 针对不同语言提供了编程接口，并包含标准 JMS 编程接口以及用于 Windows 通信基础的类。
- 使用 HTTP 协议从浏览器发送和接收 IBM MQ 消息。

工作原理

以下是 IBM MQ 工作方式的简要概述。

Windows 如果您不熟悉使用 IBM MQ，请参阅本文档的 " 方案 " 部分中的 IBM MQ 入门 。

- 管理员使用命令创建并启动队列管理器。 随后，通常在操作系统引导时自动启动队列管理器。 然后，应用程序和其他队列管理器可以连接到它以发送和接收消息。
- 应用程序或管理员创建队列或主题。 队列和主题是由 队列管理器拥有和存储的对象。
- 当应用程序要将数据传输到另一应用程序时，会将数据置于消息中。 然后将该消息置于队列中，或者将消息发布到某个主题。 可检索消息的主要方法有三种:

  连接到同一队列管理器的点到点应用程序从同一队列中检索消息。
  例如，应用程序将消息作为存储临时或持久数据的方式放在队列上。 第二个示例是与另一个在不同进程中运行的应用程序共享数据的应用程序。

  连接到另一个队列管理器的点到点应用程序从另一个队列中检索同一消息。
  应用程序通过在队列中交换消息来相互通信。 IBM MQ 的主要用途是发送或交换消息。 一个应用程序将消息放在一台计算机上的队列上，而另一个应用程序从另一台计算机上的另一个队列中获取相同的消息。 两台计算机上的队列管理器一起工作，以将消息从第一个队列传输到第二个队列。 应用程序不会相互通信，队列管理器会相互通信。

  连接到任何队列管理器的订户应用程序检索有关公共主题的消息。
  发布程序应用程序创建消息并将其发布到一台计算机上的主题。 任意数量的订户应用程序在不同计算机上预订同一主题。 IBM MQ 将发布传递到属于订户所连接到的队列管理器的队列。 订户从队列检索消息。
- IBM MQ 通道通过网络将一个队列管理器连接到另一个队列管理器。 您可以自行创建 MQ 通道，或者队列管理器集群中的队列管理器在需要时创建 MQ 通道。
- z/OS 在 z/OS 上，可以配置多个队列管理器以共享耦合设施上的队列。 连接到不同队列管理器的应用程序可以在同一队列中获取和放置消息。
- 在一个队列管理器上可以有许多队列和主题。
- 在一台计算机上可以有多个队列管理器。
- 应用程序可以在与队列管理器相同的计算机上运行，也可以在不同的计算机上运行。 如果它在同一计算机上运行，那么它是 IBM MQ 服务器应用程序。 如果它在另一台计算机上运行，那么它是 IBM MQ 客户机应

用程序。 无论是 IBM MQ 客户机还是服务器，与应用程序几乎没有差别。 您可以使用 IBM MQ 客户机或服务器来构建客户机/服务器应用程序。

IBM MQ 随附了哪些工具和资源?

IBM MQ 提供了以下工具和资源:

- 从命令行运行的控制命令。 使用控制命令创建，启动和停止队列管理器。 您还可以使用控制命令运行 IBM MQ 管理和问题确定程序。
- IBM MQ 脚本命令 (MQSC)，由解释器运行。 使用命令创建队列和主题，配置和管理 IBM MQ 。 编辑文件中的命令，并将该文件传递到 **runmqsc** 程序以解释这些命令。 您还可以在一个队列管理器上运行解释器，该队列管理器将命令发送到另一台计算机以管理另一个队列管理器。
- 可编程命令格式 (PCF) 命令，您在自己的应用程序中调用这些命令以管理 IBM MQ。 PCF 命令与脚本命令具有相同的功能，但它们更易于编程。
- 样本程序。
- 在 Windows 和 Linux® x86 以及 x86-64 平台上，可以使用以下实用程序:
  - MQ Explorer。 您可以使用 MQ Explorer 来管理和配置整个网络。 资源管理器执行与脚本命令相同的管理任务，但更易于以交互方式使用。
  - 用于演示消息传递和验证安装的 明信片 应用程序。
  - 教程。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

第 125 页的『IBM MQ 技术概述』
使用 IBM MQ 可连接应用程序并管理整个组织中的信息分发。

相关信息

方案

IBM MQ 许可证信息

您可以使用 IBM MQ 购买的内容以及每次购买授权您安装的内容。

可以使用 IBM MQ 购买的内容

分布式平台

对于分布式平台上的 IBM MQ ，该产品包含以下可独立购买的收费组件:

5724-H72 IBM WebSphere MQ

IBM WebSphere MQ（服务器）

IBM WebSphere MQ 空闲备用

IBM WebSphere MQ 高级

IBM WebSphere MQ 高级空闲备用

IBM WebSphere MQ Advanced for Developers (免费下载，但可选择获取支持。)

对于以下收费组件，可以更新现有升级和支持。 无法购买新的权利。 请参阅 声明函 以获取更多信息。

IBM WebSphere MQ 遥测
IBM WebSphere MQ Advanced Message Security
IBM WebSphere MQ 高级消息安全性空闲备用
IBM WebSphere MQ Managed File Transfer 服务
IBM WebSphere MQ Managed File Transfer 服务空闲备用

IBM WebSphere MQManaged File Transfer 代理程序

z/OS

IBM MQ for z/OS 包含五个可单独购买的收费组件:

5655-W97
IBM MQ for z/OS

5655-MFT
IBM WebSphere MQ z/OS 的受管文件传输

5655-AMS
IBM WebSphere MQ 高级消息安全性 z/OS

5655-ADV
IBM WebSphere MQ 高级 z/OS

5655-VU8
IBM WebSphere MQ for z/OS Value Unit Edition

我的企业有权安装什么?

分布式平台

对于分布式平台上的 IBM MQ，下面的组件直接映射到 IBM MQ 安装程序可以安装的组件，因此对于这些组件，您购买的组件与可以安装的组件之间的映射很简单。

要点: IBM MQ 安装介质包含所有组件，但您只应该安装已购买权利的子集。

5724-H72 IBM MQ
IBM MQ（服务器）

包括:

未作为其他可收费组件的一部分列出的所有组件。
IBM MQ Client 的无限制安装

IBM MQ 遥测

包括:

遥测服务

IBM MQ Advanced Message Security

包括:

Advanced Message Security

IBM WebSphere MQ Managed File Transfer 服务

包括:

Managed File Transfer 记录器
Managed File Transfer 服务
Managed File Transfer 工具

IBM WebSphere MQ Managed File Transfer 代理

包括:

Managed File Transfer 代理程序
Managed File Transfer 记录器
Managed File Transfer 工具

z/OS

对于 z/OS 上的产品，安装介质仅包含您已购买的内容:

5655-W97
IBM MQ for z/OS

5655-MFT
z/OS 的 IBM WebSphere MQ Managed File Transfer

5655-AMS
IBM WebSphere MQ 高级消息安全性 z/OS

5655-VU8
IBM IBM WebSphere MQ for z/OS Value Unit Edition

5655-AV9
IBM MQ Advanced for z/OS

IBM MQ for z/OS Value Unit Edition (VUE) 限制为 zNALC 分区中的 "合格工作负载"，如产品许可证中所定义。

什么是 IBM MQ Advanced?

已引入 IBM MQ Advanced 以简化购买权利的过程。 您的企业支付了一个价格，并获得了对多个 IBM MQ 组件的权利。

IBM MQ Advanced 部件包括:

5724-H72 IBM WebSphere MQ
IBM WebSphere MQ 高级

IBM WebSphere MQ Advanced for Developers

和

5655-ADV
IBM WebSphere MQ 高级 z/OS

分布式平台上的 IBM MQ Advanced 为您的企业提供了安装以下组件的权利，最高可达您的授权容量:

- IBM WebSphere MQ（服务器）
- IBM WebSphere MQ Advanced Message Security
- IBM WebSphere MQ Managed File Transfer 服务
- 无限制安装 IBM WebSphere MQ Telemetry
- IBM WebSphere MQ Client 的无限制安装
- *** 新建 *** IBM WebSphere MQ Managed File Transfer 代理程序的无限制安装，提供代理程序连接到的队列管理器使用 IBM MQ Advanced 容量进行授权

此外，您的企业可以根据需要混用和匹配 IBM MQ 版本。 因此， IBM WebSphere MQ (服务器) 权利的 100 个 PVU 可以拆分为 IBM WebSphere MQ 7.5 版本的 50 个 PVU 和此组件的 IBM MQ 8.0 版本的 50 个 PVU。

IBM MQ Advanced for Developers 授予对 IBM MQ Advanced 随附的同一组组件的权利。

注意: IBM MQ 许可证定义被视为开发目的的内容。

IBM WebSphere MQ Advanced for z/OS 授予安装以下组件的权利，使其达到您的授权容量:

- IBM WebSphere MQ z/OS 的受管文件传输
- IBM WebSphere MQ 高级消息安全性 z/OS
- 在分布式平台上无限制安装 IBM WebSphere MQ Client
- ***New *** 在分布式平台上无限制地安装 IBM WebSphere MQ Managed File Transfer Agent

要点: 您的企业需要单独购买 IBM WebSphere MQ for z/OS 基本队列管理器的权利。

什么是空闲备用部件?

已引入空闲备用部件以迎合高可用性环境，其中被动系统已安装 IBM WebSphere MQ 并且可用，但该系统未执行任何 IBM WebSphere MQ 处理工作或活动，只是与活动队列管理器的配置和活动保持最新。 在这种情况下，可能适用较低的费用。

注意:

1. 使用 IBM WebSphere MQ 多实例队列管理器功能还需要空闲备用权利。
2. IBM WebSphere MQ Telemetry 组件没有空闲备用部件。 需要为主动和被动系统购买相同的 IBM WebSphere MQ Telemetry 部件，除非您具有 IBM WebSphere MQ 空闲备用，在这种情况下将包含该部件。
3. IBM WebSphere MQ Managed File Transfer 代理程序也没有空闲备用部件，因为端点不是服务器环境的一部分。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

第 125 页的『IBM MQ 技术概述』
使用 IBM MQ 可连接应用程序并管理整个组织中的信息分发。

IBM MQ 8.0.0 中的新增功能

IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

此发行版中的新增内容:

- 更简单，更灵活的安全管理
- 能够与现有身份存储库集成，实现解决方案之间的一致性，并减少与许多用户一起使用的解决方案的安全管理
- 更灵活地管理客户机连接信息
- 可定制的发布-在大规模集群中进行路由以提高效率
- 通过在所有受支持的平台上定义多个集群传输队列来分隔工作负载的能力
- z/OS 大量增加了 IBM MQ for z/OS 中缓冲池的容量
- z/OS 增加了 IBM MQ for z/OS 中的最大可寻址日志范围
- 使用 Db2 的 QREP 功能提高了数据库复制流量的性能
- 通过对代码路径的锁定和优化进行优化，提高了效率并实现了更线性的扩展
- Windows 上的完整 64 位队列管理器实现以及受管 .NET 应用程序中对 SSL 的支持
- 支持 JMS 标准的 JMS 2.0 版本，包括延迟消息传递，共享预订和异步发送操作

新的系列功能包括:

- 发布/预订集群的主题主机路由
- 支持 JMS 2.0
- 第 15 页的『安全性: 连接认证』
- 第 16 页的『安全性: 多个可配置证书』
- 安全性 :CHLAUTH 规则中的逆向查找主机名
- 第 16 页的『集成 SupportPac MO03 - IBM MQ 队列装入/卸装实用程序』
- Managed File Transfer: 使用 IBM MQ 安全性
- Managed File Transfer: 支持在协议网桥代理与 FTPS 服务器之间的通信中指定密码
- Managed File Transfer: 资源监视器的新规范
- Managed File Transfer: 用于处理传输的增强功能

分布式平台的新增功能包括:

- 客户机和服务器连接通道的性能改进
- Windows Windows 上的 64 位队列管理器实现
- Windows 第 18 页的『清除 IBM MQ 安装信息』
- IBM i Managed File Transfer: IBM i 上的用户出口样本
- Managed File Transfer: 4690 OS 支持
- 在受管 .NET 应用程序中支持 SSL
- 用于传输本机 IBM MQ 消息的非 SOAP 接口

z/OS 新增功能包括:

- 增加缓冲池数
- 缓冲池可以位于条形上方
- 支持 z/OS 上的消息禁止
- 支持 z/OS 上的多个集群传输队列
- 对客户机连接的更改
- IBM IBM WebSphere MQ for z/OS Value Unit Edition (VUE)
- 利用 zEnterprise Data Compression (zEDC) Express 工具
- IBM MQ Advanced Message Security 的集成
- 支持 64 位应用程序
- 增加最大可寻址日志范围
- 第 21 页的『通道启动程序 SMF 数据』
- Managed File Transfer: z/OS 的新增内容
- 第 22 页的『活动日志数增加』

相关概念

第 22 页的『IBM MQ 8.0.0 中更改的内容』

第 46 页的『IBM MQ 8.0 修订包中新增和更改的内容』
本节描述了 版本 8.0 修订包中的新增功能和已更改的功能。

第 77 页的『先前版本中新增和更改的内容』
指向有关在 IBM MQ 8.0 之前的产品版本中发生的新功能以及对功能和资源的更改 (包括稳定化，废弃和移除) 的信息的链接。

第 80 页的『产品文档中的发行版和平台图标』
IBM Documentation 中的单个信息集中提供了所有 IBM MQ 8.0 修订包和平台的产品文档。 如果特定于给定修订包或平台的信息，那么此信息由矩形图标指示。 适用于所有修订包和平台的信息保持未标记状态。

相关信息

MQ Explorer 中的新增内容和更改内容

IBM MQ 8.0 系统需求

新的系列功能

IBM MQ 8.0 为发布/预订集群提供 (针对所有平台) 主题主机路由，并支持 JMS 2.0。

- 发布/预订集群的主题主机路由
- 对 JMS 2.0 的支持
- 第 15 页的『安全性: 连接认证』
- 第 16 页的『安全性: 多个可配置证书』
- 安全性 :CHLAUTH 规则中的逆向查找主机名

- 第 16 页的『集成 SupportPac MO03 - IBM MQ 队列装入/卸装实用程序』
- Managed File Transfer: 使用 IBM MQ 安全性
- Managed File Transfer: 支持在协议网桥代理与 FTPS 服务器之间的通信中指定密码
- Managed File Transfer: 资源监视器的新规范
- Managed File Transfer: 用于处理传输的增强功能

发布/预订集群的主题主机路由

在先前的版本中，当在队列管理器上配置集群主题时，集群中的所有队列管理器都可识别集群中的所有其他队列管理器。 执行发布和预订操作时，每个队列管理器便会直接连接到所有其他队列管理器。 IBM MQ 8.0 中仍提供了此方法，但将其称为直接路由。

版本 8.0 中还增加了一种备用方法，称为主题主机路由。 利用此方法，集群中的所有队列管理器都可识别用于托管路由主题定义的集群队列管理器。 在执行发布和预订操作时，集群中的队列管理器只会连接到这些主题主机队列管理器，而不会彼此直接连接。 主题主机队列管理器负责将发布从执行发布的队列管理器路由至具有匹配预订的队列管理器。

主题主机路由发布/预订集群提供了以下优势：

- 提升了大型集群的可伸缩性。 只有主题主机队列管理器需要能够连接到集群中的所有其他队列管理器。 因此，队列管理器之间存在较少的通道，队列管理器间的发布/预订管理流量比用于直接路由的少。 在队列管理器上进行预订更改时，只需要通知主题主机队列管理器。
- 对物理配置进行更多控制。 对于直接路由，所有队列管理器采用所有角色，因此全都需要同等能力。 对于主题主机路由，您可显式选择主题主机队列管理器。 因此，您可以确保这些队列管理器在合适的设备上运行，并且可以将功能差一些的系统用于其他队列管理器。

请参阅 设计发布/预订集群， 配置发布/预订集群和 针对发布/预订集群的路由: 行为注释。

JMS 2.0 支持

IBM MQ 8.0 支持 JMS 标准的 JMS 2.0 版本。 此实现提供标准 API 的所有功能，但需要的接口更少，并且更便于使用。 有关更多信息，请参阅 JMS 模型 和 JMS 2.0 规范 (Java™.net)。

JMS 2.0 支持包含以下功能：

延迟交付
现在，您可以通过在发送消息时指定传递延迟来延迟消息传递，以便 JMS 提供者在经过指定的传递延迟之后才会传递消息。 有关更多信息，请参阅 JMS 2.0 交付延迟。

共享的预订
共享预订用于在多个使用者之间共享来自主题预订的消息。 只会将来自预订的每条消息发送给该预订上的一个使用者。 有关更多信息，请参阅 克隆和共享预订。

异步发送操作
现在，应用程序可以异步发送消息。 异步发送消息时，控制权将立即返回到发送应用程序，而不必等待来自服务器的应答，从而使发送应用程序能够执行其他操作，而不是在等待来自服务器的应答时被阻止处理。

JMS 2.0 规范引入了某些行为更改。 IBM MQ 8.0 包含属性 com.ibm.mq.jms.SupportMQExtensions，可将其设置为 TRUE 以将这些更改还原回先前版本的行为。 有关更多信息，请参阅 SupportMQExtensions 属性 和 用于配置 JMS 客户机行为的属性。

安全性: 连接认证

在分布式平台上，对于每个队列管理器，您可以选择队列管理器使用本地操作系统或 LDAP 服务器来认证用户标识和密码。 通过在队列管理器的 **CONNAUTH** 属性中命名相应的认证信息对象来指定此选项。

新属性 **CHCKLOCL** 用于开启本地连接的用户标识和密码检查。 在 IBM WebSphere MQ 7.1 与最新版本之间进行迁移时，每个队列管理器上的 **CONNAUTH CHCKLOCL** 属性都将设置为 NONE，从而确保版本与版本之间的连续性，但会关闭连接认证。 对于新的 IBM MQ 8.0 安装， **CONNAUTH CHCKLOCL** 属性设置为

OPTIONAL。 这意味着不需要用户标识和密码，但如果提供了用户标识和密码，那么它们必须是有效的对，否则将被拒绝。

z/OS 在 z/OS 上， IBM MQ 8.0 仅支持操作系统认证，不支持 LDAP。

有关更多信息，请参阅 连接认证。

安全性: 多个可配置证书

客户机和队列管理器不再限于 SSL/TLS 通道的单个证书。 您可以通过将 **CERTLABL** 通道属性设置为您选择的值来配置所使用的证书。 请参阅 数字证书标签: 了解需求。

使用多个证书允许您使用由不同认证中心 (CA) 签署的证书。 但是，请注意以下事项:

- 您可以使用 **CERTLABL** 配置设置来选择 IBM MQ 将哪个证书发送到其远程合作伙伴。 本地配置的证书标签用于从本地密钥存储库中选择证书。 将所选证书发送到远程 IBM MQ 合作伙伴以进行认证。

  在远程 IBM MQ 合作伙伴处，将根据 SSL/TLS 证书验证策略以及 IBM MQ 安全设置来验证证书。

  其中一个安全设置是 **SSLPEER** 通道属性，此属性指定必须与接收到的证书的主体集 DN 匹配的主体集专有名称 (DN) 过滤器字符串。

  注意: 通道 **SSLPEER** 设置仅与证书的主体集 DN 匹配，而与颁发者 DN 不匹配，因此有可能发生虚假匹配。
- 因此，您应该优先使用 **SSLPEERMAP** 通道认证规则而不是通道 **SSLPEER** 过滤器，因为 **SSLPEERMAP** 认证能够同时匹配主体集 DN (**SSLPEER**) 和颁发者 DN (**SSLCERTI**)，因此不太可能匹配错误的证书。
- 本地 **CERTLABL** 配置与远程 **SSLPEERMAP** 配置不同，但它们之间存在重要关系。

  由其标签选择的本地证书包含可由远程 IBM MQ 合作伙伴验证的主体集 DN 和颁发者 DN。 因此，正确配置本地证书标签非常重要，以避免远程 IBM MQ 合作伙伴发生 **SSLPEER** 和 **SSLPEERMAP** 匹配错误。

有关更多信息，请参阅使用 SSL 或 TLS。

安全性: CHLAUTH 规则中的逆向查找主机名

IBM MQ CHLAUTH 规则已增强，因此 CHLAUTH 定义可以使用域名服务器 (DNS) 主机名而不是 IP 地址，并且 IBM MQ 在执行通道初始化时执行逆向 DNS 主机名查找以获取 IP 地址。

IBM MQ 8.0 还引入了队列管理器 REVDNS 属性，该属性控制在通道初始化期间是否应该执行入站通道的 IP 地址的反向 DNS 主机名查找。

如果启用此属性，那么需要此信息时，将逆向查找 DNS 主机名以查找入站通道的 IP 地址。

如果未启用 REVDNS 属性，那么不会反向查找 DNS 主机名以查找入站通道的 IP 地址。 有关 REVDNS 属性的更多信息，请参阅 ALTER QMGR。

请注意， APAR IC96408 引入了针对先前版本的 IBM WebSphere MQ 的逆向 DNS 主机名查找，以便记录一些错误消息。 对于 IBM MQ 8.0，此值为 true。

集成 SupportPac MO03 - IBM MQ 队列装入/卸装实用程序

IBM MQ Supportpac MO03 中提供的 **qload** 实用程序现在作为 **dmpmqmsg** 实用程序集成到 IBM MQ 版本 8.0 中。

Linux UNIX 在 UNIX 和 Linux 平台上，该实用程序在 `<installdir>./bin` 中可用

Windows 在 Windows 平台上，该实用程序在 `<installdir>./bin64` 中作为服务器文件集的一部分提供。

z/OS 在 z/OS 上，该实用程序可用作 SCSQLOAD 库中的可执行模块 CSQUDMSG，其别名为 QLOAD 以实现兼容性。 样本 JCL 也作为 SCSQPROC 中的成员 CSQ4QLOD 提供。

有关更多信息，请参阅 使用 **dmpmqmsg** 实用程序。

Managed File Transfer: 使用 IBM MQ 安全性

IBM MQ Managed File Transfer 版本 8.0 支持 IBM MQ 版本 8.0 的安全功能，缺省方式为 "已禁用"。 如果关联的队列管理器已启用安全性并且需要凭证详细信息 (用户标识和密码)，那么必须先启用此功能，然后才能成功连接到队列管理器。 有关更多信息，请参阅 使用 IBM MQ 安全性 和 连接认证。

Managed File Transfer: 支持在协议网桥代理与 FTPS 服务器之间的通信中指定密码

您可以使用 `ProtocolBridgeProperties.xml` 文件中名为 cipherSuiteList 的新属性为协议网桥代理与 FTPS 协议服务器之间的连接显式指定密码套件列表。 将在代理与 FTPS 服务器之间的协商中使用提供的列表。 有关更多信息，请参阅 协议网桥属性文件格式。

Managed File Transfer: 资源监视器的新规范

已向资源监视器添加以下支持:

- 您可以将用户元数据传递到资源监视器端点
- 您可以在单个传输请求中传输有序的文件列表
- 可以将作业名信息指定为资源监视器请求的用户定义标识

有关更多信息，请参阅 fteCreateMonitor 命令。

Managed File Transfer: 用于处理传输的增强功能

Managed File Transfer 中包含以下增强功能:

- 您可以将一个大型文件提交到一个消息传输，最大文件大小为 100 MB。 要减少大型单个文件到单条消息传输的内存使用量，建议将 **fteCreateTransfer** 命令上的 **-qs** 参数设置为等于要写入的消息大小。 如果有大于 100 MB 的文件，并且也在 **fteCreateTransfer** 命令上指定了 **-qs** 参数，那么该文件将拆分为多条消息。

  在将二进制文件恢复到消息传输的情况下，将从写入最后一个检查点的位置重新启动传输。 在恢复文本传输的情况下，传输将从文件开头重新启动，这可能会导致目标队列上的消息组不完整。 当失败的文本传输重新启动时，将写入一组全新的消息。

  有关更多信息，请参阅 fteCreateTransfer。
- 对于提前失败的传输，将发布传输进度日志消息。 然后，您可以使用已发布的有关失败传输中的传输项的信息来重新提交该传输。
- commandMessagePriority 属性设置 **fteStopAgent**、**fteCancelTransfer**、**ftePingAgent** 和 **fteSetAgentTraceLevel** 命令的内部消息和命令消息的优先级。 您还可以使用 commandMessagePriority 属性设置内部确认和期望确认消息的优先级。 您可以将 commandMessagePriority 设置为某个值，以使内部 Managed File Transfer 消息的优先级高于新的传输请求，从而提高代理性能。 有关更多信息，请参阅 installation.properties 文件。
- 您可以使用 maxInlineFileSize 属性为单个文件到文件或文件到消息传输设置传输请求消息中包含的最大文件大小。 这可能会提高传输性能。 有关更多信息，请参阅 agent.properties 文件。
- 可以使用 enableMemoryAllocationChecking 属性确保代理在启动传输之前执行以下检查：检查是否有足够的内存可用于运行传输。 如果可用内存不足，传输将被置于恢复状态，以防止代理因内存不足错误而失败。 有关更多信息，请参阅 agent.properties 文件。
- "文件到消息" 和 "消息到文件" 传输的传输日志发布包含所有传输请求属性。

相关概念

distributed 分布式平台的新增信息

IBM MQ 8.0 for Distributed 平台为客户机和服务器连接通道， Windows 上的 64 位队列管理器实现，受管 .NET 应用程序中的 SSL 支持以及用于传输本机 IBM MQ 消息的非 SOAP 接口提供了性能改进。

z/OS z/OS 平台的新增功能

IBM MQ 8.0 for z/OS 提供了大量增加的缓冲池，消息抑制，多个集群传输队列，改进的数据压缩工具，新的安全选项和改进的监视。

distributed 分布式平台的新增信息

IBM MQ 8.0 for Distributed 平台为客户机和服务器连接通道， Windows 上的 64 位队列管理器实现，受管 .NET 应用程序中的 SSL 支持以及用于传输本机 IBM MQ 消息的非 SOAP 接口提供了性能改进。

- 客户机和服务器连接通道的性能改进
- Windows Windows 上的 64 位队列管理器实现
- Windows 第 18 页的『清除 IBM MQ 安装信息』
- IBM i Managed File Transfer: IBM i 上的用户出口样本
- Managed File Transfer: 4690 OS 支持
- 在受管 .NET 应用程序中支持 SSL
- 用于传输本机 IBM MQ 消息的非 SOAP 接口

客户机和服务器连接通道的性能改进

通过 IBM MQ 8.0，如果在 SVRCONN (服务器连接) 通道上将 **SHARECNV** (共享对话) 参数设置为 1，那么可以提高 MQGET 等待调用的性能。 对于发出非预读同步获取等待调用 (例如 C 客户机 MQGET 等待调用) 的客户机应用程序，您可以获得此改进。 当连接这些客户机应用程序时，分布式服务器使用的线程更少，内存更少，吞吐量也会增加。 要从性能改进中获益，请尽可能使用 SHARECNV(1) for SVRCONN 通道。 为了与先前发行版保持一致， **SHARECNV** 的缺省值仍为 10，因此需要显式设置 SHARECNV(1)。 请参阅 调整客户机和服务器连接通道 和 MQI 客户机: 客户机连接和服务器连接的缺省行为。

Windows 上的 64 位队列管理器实现

Windows

IBM MQ 8.0 for Windows 队列管理器运行 64 位进程，并支持 32 位和 64 位应用程序。 在先前版本中，Windows 上的队列管理器仅由 32 位进程组成，即使队列管理器在 64 位 Windows 平台上运行也是如此。 在 IBM MQ 8.0 上， Windows 平台上的队列管理器由 64 位进程组成，因此使用 Windows 64 位操作系统的 64 位寻址功能。 命令，二进制文件等的位置在 版本 8.0 中已更改。 所有命令现在都位于 bin64 文件夹中。 bin 文件夹包含 32 位库。 可以从 bin 或 bin64 文件夹运行队列管理器命令。

有关在规划 IBM MQ 8.0 for Windows 的迁移任务之前必须了解的概念的更多信息，请参阅 IBM MQ 8.0 上 Windows 的更改简介。

有关 IBM MQ 8.0 支持的 Windows 版本的更多信息，请参阅 IBM MQ V8.0。

要点: 从 IBM MQ 8.0 开始，不再支持 Windows XP。

清除 IBM MQ 安装信息

Windows

从 IBM MQ 8.0 开始，在名为 **ResetMQ.cmd** 的安装介质的根目录上提供 Windows 命令文件，使您能够在特定情况下除去整个 IBM MQWindows 安装 (如果需要)。

请参阅 清除安装设置 以获取更多详细信息。

Managed File Transfer: IBM i 上的用户出口样本

IBM i

通过 IBM MQ 8.0，在 IBM i 上，您可以将 MFT 中提供的用户出口样本用于以下任务:

- 在 QDLS 文件系统中传输文件。
- 以与 MFT 文件监视器相同的方式从 IBM i 库自动传输物理文件成员。
- 在传输期间删除源文件成员时，请删除空文件对象。

有关更多信息，请参阅 样本 IBM i 用户出口。

Managed File Transfer: 4690 OS 支持

通过 IBM MQ 8.0，可以在运行 4690 OS 的设备上安装 IBM MQ Managed File Transfer 。 有关更多信息，请参阅 在零售环境中使用 MFT。

在受管 .NET 应用程序中支持 SSL

IBM MQ 8.0 在 IBM MQ.NET 完全受管客户机中支持 SSL/TLS。 在先前发行版中，此支持仅存在于非受管客户机中。 此新产品基于 Microsoft.NET SSLStreams 套件，与基于 GSKit 的其他 IBM MQ 客户机不同。 在非受管方式下使用 IBM MQ.NET 客户机是不变的，并且继续受支持。 IBM MQ.NET 客户机的非受管方式和受管方式都支持 SSL/TLS 协议，但基于 GSKit 和 SSLStreams。 对于受管方式，由于对 Microsoft.NET SSLStreams 的依赖关系，存在一些限制，例如，没有用于设置 CipherSpec 的程序化选项，以及启用或禁用 FIPS。 有关更多信息，请参阅 第 280 页的『IBM MQ.NET 受管客户机的 SSL 和 TLS 支持』 和 受管 .NET 客户机的 SSL 和 TLS 支持。

用于传输本机 IBM MQ 消息的非 SOAP 接口

IBM MQ 8.0 提供了通过 Microsoft Windows Communication Foundation (WCF) 通道传输本机 IBM MQ 消息的功能，即，通过 WCF 通道传输没有任何 SOAP 头的消息。 此支持使 WCF 客户机能够发送可供 C , Java， JMS 和 .NET 的 IBM MQ 客户机使用的 IBM MQ 消息。 消息也可以由 WCF 服务应用程序使用。

已升级 WCF 通道上提供 SOAP/JMS 的 WCF 的现有 IBM MQ 通道，以提供 SOAP/ IBM MQ 设施。 对于 IBM MQ 的较早发行版， WCF URI 必须是允许应用程序提供 JMS 对象详细信息以促进 SOAP/JMS 的 JMS: \\ URI。 但是，对于 IBM MQ 8.0，应用程序可以使用同一客户机，但提供 wmq: \\ 样式的 URI 以提供 SOAP/ IBM MQ 样式的支持。 此更改通过 WCF 提供 SOAP (作为 SOAP/JMS 和/或 SOAP/ IBM MQ) 和非 SOAP (本机 IBM MQ 消息) 通道。 有关更多信息，请参阅 IBM MQ 用于 WCF 端点 URI 地址格式的定制通道。

相关概念

新的系列功能
IBM MQ 8.0 为发布/预订集群提供 (针对所有平台) 主题主机路由，并支持 JMS 2.0。

z/OS z/OS 平台的新增功能
IBM MQ 8.0 for z/OS 提供了大量增加的缓冲池，消息抑制，多个集群传输队列，改进的数据压缩工具，新的安全选项和改进的监视。

z/OS z/OS 平台的新增功能

IBM MQ 8.0 for z/OS 提供了大量增加的缓冲池，消息抑制，多个集群传输队列，改进的数据压缩工具，新的安全选项和改进的监视。

- 增加缓冲池数
- 缓冲池可以位于条形上方
- 支持 z/OS 上的消息禁止
- 支持 z/OS 上的多个集群传输队列
- 对客户机连接的更改
- IBM IBM WebSphere MQ for z/OS Value Unit Edition (VUE)
- 利用 zEnterprise Data Compression (zEDC) Express 工具
- IBM MQ Advanced Message Security 的集成
- 支持 64 位应用程序
- 第 21 页的『增大了最大可寻址日志范围』
- 第 21 页的『通道启动程序 SMF 数据』
- Managed File Transfer: z/OS 的新增内容
- 第 22 页的『活动日志数增加』
- 第 22 页的『存储类内存的使用 (SCM)』

缓冲池数增加

IBM MQ 版本 8.0 for z/OS 现在最多支持 100 个缓冲池; 前发行版支持 16 个缓冲池。 拥有更多缓冲池可提供管理灵活性以及应用程序/队列隔离的可能性，因为现在可以有从页集到缓冲池的一对一映射。 有关定义缓冲池的更多信息，请参阅 定义缓冲池 和 DEFINE BUFFPOOL 命令。

缓冲池可以位于条形上方

IBM MQ 版本 8.0 for z/OS 允许缓冲池位于条形上方以及条形下方。 位于条形上方的缓冲池可以具有更多的缓冲区，有可能提高某些应用程序类的性能。

如果缓冲池位于条形上方，那么可以选择由固定的 4 KB 页面 (而不是可分页的 4 KB 页面) 来支持该缓冲池。 对于缓冲池的某些用途，这可以提高性能。

请参阅 地址空间存储器 以了解大型缓冲池对 z/OS 系统的影响。

通过在 DEFINE BUFFPOOL 和 ALTER BUFFPOOL 命令上指定 LOCATION (ABOVE) ，缓冲池可以位于条形上方。

支持 z/OS 上的消息禁止

IBM MQ 版本 8.0 for z/OS 引入了 ZPARM 属性 **EXCLMSG**，用于列出要从发出中排除的控制台消息。 此属性是动态的，也可以使用 **SET SYSTEM** 命令进行配置。 有关更多信息，请参阅 使用 CSQ6SYSP。 此外，还有两条新消息 CSQX511I 和 CSQX512I (针对仅适用于 SVRCONNs 的通道启动/停止事件) ，使您能够在不影响队列管理器通道发出的消息的情况下列出这些消息。 有关更多信息，请参阅 CSQX511I: *csect-name* SRVCONN Channel *channel-name* 已启动，连接 *conn-id* 和 CSQX512I: *csect-name* SRVCONN Channel *channel-name* 不再活动，连接 *conn-id* 。

支持多个集群传输队列

IBM MQ 8.0 for z/OS 支持多个集群传输队列，如 版本 7.5 中针对分布式平台引入的那样。

您可以指定要用于集群发送方通道的传输队列，这将提供更高的可用性，因为一个集群发送方通道的消息可以与其他集群发送方通道隔离。 有关更多信息，请参阅 集群: 规划如何配置集群传输队列。

要由每个集群发送方通道转发的消息将放置在单独的集群传输队列上。 有关更多信息，请参阅 使用集群传输队列和集群发送方通道。

您还可以通过设置新的队列属性 CLCHNAME 来手动配置集群传输队列。

您可以决定哪些集群发送方通道共享哪些传输队列，哪些具有单独的传输队列，哪些使用集群传输队列或队列。 有关更多信息，请参阅 规划如何使用多个集群传输队列。

对客户机连接的更改

先前想要将 IBM MQ 客户机从非 z/OS 平台直接连接到 z/OS 上的 IBM MQ 服务器的用户需要使用单独收费的客户机附件功能部件 (CAF) 来许可此功能。

在 IBM MQ for z/OS 版本 8.0 中除去了客户机附件功能部件，这意味着不需要额外的许可证费用或安装步骤来支持 IBM MQ 客户机连接。

在先前发行版中，最多允许 5 个服务器连接通道实例从 IBM MQ Explorer 管理队列管理器。 查看 SYSTEM.ADMIN.SVRCONN 和 SYSTEM.BKR.CONFIG 通道定义，因为现在可以根据需要增加这些设置。

如果您的站点先前未启用客户机连接，请考虑实施 CHLAUTH 规则以通过其他服务器连接通道定义 (尤其是 SYSTEM.DEF.SVRCONN。

IBM IBM WebSphere MQ for z/OS Value Unit Edition (VUE)

IBM IBM MQ for z/OS Value Unit Edition (VUE) (VUE) 提供基本 IBM MQ for z/OS 的所有功能和能力，采用的格式可为合格 IBM Z Systems New Application License Charge (zNALC) 逻辑分区 (LPAR) 中部署的符合条件的工作负载提供一次性付费 (OTC) 价格指标。

术语“符合条件的工作负载”定义为使用 IBM MQ for z/OS VUE 服务器环境执行的新工作负载，条件是该工作负载通过 zNALC 资格流程经鉴定合格并获批准。

有关 zNALC 的进一步信息，请参阅 IBM Z Systems 软件定价

OTC 价格指标为新的连接到 IBM MQ for z/OS 的应用程序和新的 IBM MQ for z/OS VUE 服务支持工作负载提供备用定价模型。

支持 zNALC 指标将为运行合格应用程序的 LPAR 上的 z/OS 操作系统提供更低的价格。

IBM MQ for z/OS VUE 可以连接到 IBM MQ for z/OS 的其他受支持版本（不管是在 zNALC 环境还是非 zNALC 环境中）以进行工作负载联合和系统管理。

IBM MQ for z/OS VUE 允许来自在其他平台上运行的 IBM MQ 客户机的连接。

利用 zEnterprise 数据压缩 (zEDC) Express 工具

对于安装并启用了 zEDC Express 设施的系统，当指定 COMPMSG (ZLIBFAST) 并且未压缩消息的大小大于 4 KB 时， IBM MQ 使用 zEDC 来执行通道消息压缩。

通道消息压缩可帮助减少 IBM MQ 系统之间的链路带宽需求。 此外，当为通道配置了 SSL CipherSpecs 时，在加密之前压缩数据可以降低加密的中央处理单元成本，并增加通道上的消息吞吐量。

有关更多信息，请参阅 准备 IBM MQ for z/OS 以使用 zEnterprise Data Compression Express 工具。

IBM MQ Advanced Message Security 的集成

现在， Advanced Message Security (AMS) 已与 IBM MQ for z/OS 更集成。 AMS 通过安装单独定价的支持产品来启用。 有关安装和启用 AMS 的更多信息，请参阅 IBM MQ Advanced Message Security for z/OS 。

支持 64 位应用程序

支持使用以 C 语言编写的 IBM MQ API 的应用程序，该 API 是使用 LP64 编译选项构建的，并与相应的 64 位侧板绑定。 有关更多信息，请参阅 构建 64 位 C 应用程序。

增大了最大可寻址日志范围

恢复数据将写入日志数据集，因此在重新启动队列管理器时，持久消息可用。 术语 "日志相对字节地址" (log RBA) 用于标识日志数据集中数据的位置。

对于 IBM MQ 的较早发行版，日志 RBA 最多可以处理 256 太字节的数据。 在队列管理器达到最大数据之前，必须停止队列管理器，并运行一些作业以允许日志以较低的值再次启动。 对于高使用率队列管理器，此操作可能每年执行一次。

在 IBM MQ 8.0 中，队列管理器现在可以处理 64,000 倍以上的数据 (16 EB)。 在队列共享组 (QSG) 中使用此功能之前，所有队列管理器都需要运行 IBM MQ 8.0。 使用此功能的影响是写入的日志数据大小稍大。

有关更多信息，请参阅 使日志相对字节地址更大。

通道启动程序 SMF 数据

z/OS 现在支持生成通道启动程序 SMF 统计信息记录，其中包含有关任务 (分派器， 适配器， DNS 和 SSL) 的统计信息以及通道的记帐数据。 您可以使用此信息来确定通道启动程序和关联通道的性能。 有关更多信息，请参阅第 288 页的『通道启动程序的 SMF 记录』。

Managed File Transfer: z/OS 的新增内容

z/OS 上的 IBM MQ Managed File Transfer 包含以下新功能:

- IBM MQ Managed File Transfer on z/OS 产品分别与 IBM MQ 和分布式组件上的 IBM MQ Managed File Transfer 一起安装。
- 您可以使用 PDSE 数据集来配置 z/OS 环境。 使用 PDSE 数据集将无需访问由 UNIX 系统服务文件维护的 MFT 配置属性文件。

使用新的 JCL 命令模板 PDSE 库在单个数据集中指定替换值， JCL 变量和配置属性。 USS 文件系统中的配置属性文件与数据集中定义的配置同步。 有关更多信息，请参阅 z/OS 上的配置选项 。

仍支持使用 **fte** 命令配置 z/OS 环境。

- 您可以使用 z/OS 自动重新启动管理器来配置代理程序和记录器以进行自动重新启动。 有关更多信息，请参阅 配置 MFT for ARM。
- 以文本方式传输到使用 **-dtr** 参数的数据集时，可以截断目标记录。 有关更多信息，请参阅 fteCreateTransfer 命令。

活动日志数增加

现在， IBM MQ for z/OS 在活动日志数据集的每个日志副本环中最多支持 310 个活动日志。 具有更多活动日志允许队列管理器容许归档问题，否则，这些问题可能导致活动日志空间填满， IBM MQ 消息传递延迟，暂挂归档问题的解决。

对于每个最大大小为 4 GB 的活动日志， 310 个活动日志可以提供 1.2 TB 的日志空间。

必须应用 APAR PI46853 才能启用此功能。

有关更多信息，请参阅 日志数据集定义。

存储类内存的使用 (SCM)

z13， zEC12 和 zBC12 机器允许安装 Flash Express 卡。 这些卡包含闪存固态驱动器 (SSD)。 安装后，可将卡中的闪存分配给通常称为 SCM 的一个或多个 LPAR。

当与 IBM MQ for z/OS 共享队列配合使用时，使用存储类内存 (SCM) 可能是有利的。

有关更多信息，请参阅 将存储类内存用于共享队列

相关概念

新的系列功能
IBM MQ 8.0 为发布/预订集群提供 (针对所有平台) 主题主机路由，并支持 JMS 2.0。

distributed 分布式平台的新增信息
IBM MQ 8.0 for Distributed 平台为客户机和服务器连接通道， Windows 上的 64 位队列管理器实现，受管 .NET 应用程序中的 SSL 支持以及用于传输本机 IBM MQ 消息的非 SOAP 接口提供了性能改进。

第 45 页的『版本 8.0.0 的新增和已更改的 z/OS 消息』
已添加大量新消息，并且已更改 IBM MQ for z/OS 版本 8.0.0 的某些现有消息。

IBM MQ 8.0.0 中更改的内容

本节汇总了 IBM MQ 8.0.0 中发生的功能和资源更改 (包括稳定，废弃和移除) ，并提供了指向更多信息的链接。

版本 8.0.0 中的更改

对于此列表的子集 (仅包含那些可能影响现有应用程序或配置的更改) ， 请参阅 影响迁移的更改。

表 *1: V 8.0.0* 更改的摘要

更改内容	**链接到更多信息**
直接进行 MQI 调用的现有 API 出口，即不使用 MQIEP 的那些出口。	第 26 页的『API: 更改为 API 出口的操作』
SVRCONN 通道 STOPPED 通道状态已更改，以允许其在队列管理器重新启动时持久存储	第 26 页的『在 SVRCONN 通道上更改为 STOPPED 状态』
crtmqenv	第 26 页的『命令更改 :crtmqenv 命令』

表 *1: V 8.0.0* 更改的摘要 *(继续)*	
更改内容	**链接到更多信息**
dmpmqcfg 命令已更改，以确保在生成的命令中注释掉密码字段。	第 26 页的『命令更改: dmpmqcfg 输出』
DISPLAY CONN **runmqsc** 命令	第 26 页的『命令更改: 响应来自 runmqsc 的 DISPLAY CONN 命令时显示的结果』
runmqsc 命令具有两种新的客户机方式和一个新的 MQPROMPT 环境变量	第 27 页的『命令更改 :runmqsc 命令』
命令级别	第 27 页的『命令级别: 更改』
多实例队列管理器的启动事件	第 30 页的『多实例队列管理器的扩展启动事件』
IBM i IBM i 对 IBM MQ Advanced Message Security 的支持	第 30 页的『IBM MQ 8.0 的 IBM i: 更改』
IBM i IBM i 支持拆分集群传输队列	第 30 页的『IBM MQ 8.0 的 IBM i: 更改』
使用 Java 和 JMS 客户机进行字符串转换	第 30 页的『Java 和 JMS: 对字符串转换的更改』
IBM MQ classes for Java 和 IBM MQ classes for JMS 中的 SSL/TLS CipherSuites 支持	第 30 页的『Java 和 JMS: 对 CipherSuite 支持的更改』
IBM MQ 针对 WildFly 的资源适配器 IVT 应用程序支持 V10	第 31 页的『Java 和 JMS: 资源适配器更改』
Java 7 支持	第 31 页的『Java: Java 7 的更改』
公共类的 IBM MQ classes for Java `connector.jar` 依赖关系和 API。	第 31 页的『Java: 对 IBM MQ classes for Java 的更改』
控制 Java 跟踪行为	第 32 页的『Java: 对 Java 跟踪行为的更改』
MQException 和 MQDataException 类日志记录	第 32 页的『Java: 对 MQException 和 MQDataException 类日志记录的更改』
IBM MQ classes for JMS 字节和流消息状态	第 32 页的『Java: 对 MQException 和 MQDataException 类日志记录的更改』
JMS ExceptionListener	第 33 页的『JMS: 版本 8.0 中的异常侦听器更改』
强制实施唯一客户机标识 (JMS 2.0)	第 33 页的『JMS: 对 IBM MQ classes for JMS 的更改』
JMS ObjectMessage 类名允许列表支持	第 33 页的『JMS: 对 IBM MQ classes for JMS 的更改』
`jms.jar` 文件 (JMS 2.0)	第 34 页的『JMS: 对 JMS 2.0』
除了正常方式和迁移方式，您还可以选择有限制的正常方式	JMS: 对 PROVIDERVERSION 属性的更改
Managed File Transfer 新代理属性 addCommandPathTo 沙箱	第 58 页的『V 8.0.0 修订包 6 中新增及更改的内容』
z/OS Managed File Transfer 新的代理属性 adminGroup，用于 z/OS 上的 MFT 代理	第 58 页的『V 8.0.0 修订包 6 中新增及更改的内容』
Managed File Transfer fteMigrate 代理命令	第 58 页的『V 8.0.0 修订包 6 中新增及更改的内容』
发布/预订集群主题行为	第 34 页的『对集群主题行为的更改』

表 *1: V 8.0.0* 更改的摘要 *(继续)*	
更改内容	**链接到更多信息**
发布/预订 DISPLAY TOPIC 和 DISPLAY TPSTATUS 输出	第 35 页的『发布/预订: 对 DISPLAY TOPIC 和 DISPLAY TSTATUS 输出的更改』
发布/预订消息选择	第 35 页的『发布/预订: 对预订选择字符串求值方式的更改』
发布/预订 PROXYSUB (FORCE)	第 35 页的『发布/预订: 使用 PROXYSUB (FORCE) 对代理预订传播进行的更改』
队列管理器缺省 TCP 发送和接收缓冲区大小	第 36 页的『队列管理器: 更改为缺省 TCP 缓冲区大小』
队列管理器版本属性	第 36 页的『队列管理器: 为集群中的队列管理器添加的版本属性』
使用 ClientRevocationChecks 属性的安全证书撤销检查配置	第 36 页的『安全性: 证书撤销检查』
CipherSuite 支持	第 36 页的『安全性: 对 CipherSuite 支持的更改』
V 8.0.0.6 V 8.0.0.6 安全性 :GSKit 版本 (IBM MQ 8.0.0 修订包 6)	第 58 页的『V 8.0.0 修订包 6 中新增及更改的内容』
CONNAUTH CHCKLOCL 参数，用于本地连接的用户标识和密码检查	第 37 页的『安全性: 新的 CONNAUTH CHCKLOCL 参数』
Linux UNIX 对 UNIX 和 Linux 的对象权限管理器 (OAM) 基于用户的许可权	第 37 页的『安全性: UNIX 和 Linux 系统上基于 OAM 用户的许可权』
在客户机上禁用 IBM MQ Advanced Message Security	第 38 页的『安全性: 在客户机上禁用 IBM MQ AMS』
客户机应用程序上的密码保护	第 39 页的『安全性: 客户机应用程序上的密码保护』
SSL 或 TLS 通道发送出口缓冲区空间	第 39 页的『安全性: 减少 SSL 和 TLS 的通道发送出口缓冲区空间』
Solaris 客户端安装	第 39 页的『Solaris 客户机: 安装方法中的更改』
IBM MQ Telemetry SDK	第 40 页的『遥测: 对 IBM MQ Telemetry Client Software Development Kit (SDK) 的更改』
Windows 缺省安装位置	第 40 页的『IBM MQ 8.0 的 Windows: 更改』
Windows 队列管理器	第 40 页的『IBM MQ 8.0 的 Windows: 更改』
Windows Microsoft Visual Studio 编译器	第 40 页的『IBM MQ 8.0 的 Windows: 更改』
z/OS 可用 CipherSpecs	第 43 页的『z/OS: 更改为可用 CipherSpecs』
z/OS 在 CHINIT 地址空间中的错误消息中显示的映射 IP 地址	第 43 页的『z/OS: 更改为在错误消息插入中显示映射的 IP 地址』
z/OS MAKECLNT CSQUTIL 工具已稳定	第 43 页的『z/OS: 更改为 MAKECLNT』
z/OS 日志 RBA 和恢复单元标识 (URID)	第 43 页的『z/OS: 对日志 RBA 和 URID 长度的更改』

表 1: V 8.0.0 更改的摘要 (继续)	
更改内容	**链接到更多信息**
z/OS IBM MQ Managed File Transfer 用于 z/OS	第 43 页的『对 z/OS， 版本 8.0 的 IBM MQ Managed File Transfer 的更改』
z/OS 通道启动程序 (CHINIT) SMF 记录	第 44 页的『z/OS: 通道启动程序 SMF 记录』
z/OS 队列管理器属性 DNSWLM 和 DNSGROUP	第 44 页的『不再支持 z/OS: WLM/DNS』
z/OS 更改为 DISPLAY USAGE 命令	显示使用情况
z/OS 对 DISPLAY 命令输出的更改	第 45 页的『z/OS: 对 DISPLAY 命令输出的更改』

版本 8.0 中的稳定化，废弃和移除

表 2: 版本 8.0 的稳定化，废弃和移除的摘要	
已稳定，不推荐或已移除的功能部件	**链接到更多信息**
z/OS 稳定： MAKECLNT CSQUTIL 工具	第 43 页的『z/OS: 更改为 MAKECLNT』
Solaris 不推荐: 使用 libmqmcs 和 libmqmzse 库进行链接	第 28 页的『不推荐: 与 libmqmcs 和 libmqmzse 库链接』
不推荐: 受管文件传输 Web 网关	第 28 页的『不推荐: Managed File Transfer Web 网关不推荐』
不推荐: SSLv3 协议	第 28 页的『不推荐: SSLv3 协议』
不推荐: CipherSpecs 使用现在被视为较弱的加密算法或协议	不推荐： 较弱的密码算法
不推荐: CipherSpecs (IBM MQ 8.0.0 修订包 6)	第 58 页的『V 8.0.0 修订包 6 中新增及更改的内容』
不推荐: MQ Explorer 服务定义向导	第 29 页的『不推荐: IBM MQ 资源管理器服务定义向导』
不推荐: 针对 SOAP 客户机和 HTTP 网桥的 IBM MQ 传输	第 29 页的『不推荐: 针对 SOAP 客户机和 HTTP 网桥的 IBM MQ 传输』
除去: z/OS **CSQUMGMB** 命令 (z/OS)	第 27 页的『命令更改: 除去 migmbbrk 和 [z/OS]CSQUMGMB』
除去: **migmbbrk** 命令	第 27 页的『命令更改: 除去 migmbbrk 和 [z/OS]CSQUMGMB』
除去: `Direct` 和 `DirectHTTP` 传输支持	第 33 页的『JMS: 对 IBM MQ classes for JMS 的更改』
除去: 支持 Visual Basic **注意:** 仍支持 Visual Basic .NET。	Windows 的软件产品兼容性报告

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』

IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 46 页的『IBM MQ 8.0 修订包中新增和更改的内容』
本节描述了 版本 8.0 修订包中的新增功能和已更改的功能。

第 77 页的『先前版本中新增和更改的内容』
指向有关在 IBM MQ 8.0 之前的产品版本中发生的新功能以及对功能和资源的更改 (包括稳定化，废弃和移除) 的信息的链接。

相关信息
MQ Explorer 中的新增内容和更改内容
IBM MQ 8.0 系统需求

API: 更改为 API 出口的操作

本身进行 API 调用 (而不是使用 **MQIEP** 结构) 的 API 出口应该与 IBM MQ API 库链接，否则它们将无法在 **runmqsc** 中装入。

从 IBM MQ 8.0 开始，如果要通过 API 出口进行直接 MQI 调用，那么需要与提供这些调用的对象进行链接。有关更多信息，请参阅 未与 MQI 库链接的 API 出口 。

在 SVRCONN 通道上更改为 STOPPED 状态

IBM MQ 8.0 允许 STOPPED 通道状态持久存储，以便队列管理器重新启动。 此行为适用于除 z/OS 以外的所有平台。

请参阅 重新启动已停止的通道 以获取更多信息。

命令更改 :crtmqenv 命令

crtmqenv 命令的行为发生了更改，以允许更改 PATH 环境变量，从而向 32 位或 64 位可执行文件的二进制路径添加前缀。

对于 IBM MQ 8.0，将对 **crtmqenv** 命令进行以下更改，您可以使用此命令为 IBM MQ 安装创建具有相应值的环境变量列表:

- 如果指定 -x 32 ，那么将更改 PATH 环境变量以向 32 位可执行文件的二进制路径添加前缀。
- 如果指定 -x 64 ，那么将更改 PATH 环境变量以向 64 位可执行文件的二进制路径添加前缀。

相关信息
克特蒙琴夫

命令更改: dmpmqcfg 输出

在 IBM MQ 8.0 中，已更改 **dmpmqcfg** 命令的输出，以确保在生成的命令中注释掉密码字段。

此更改使 **dmpmqcfg** 命令与 DISPLAY 命令保持一致，这些命令将密码字段显示为 PASSWORD (********)。

相关信息
dmpmqcfg (转储队列管理器配置)

命令更改: 响应来自 runmqsc 的 DISPLAY CONN 命令时显示的结果

当没有与连接关联的 XA 事务时，对 DISPLAY CONN **runmqsc** 命令显示的结果的 EXTURID 字段中返回的数据已更改。

在 IBM MQ 8.0 之前，如果没有与连接关联的 XA 事务，那么在 EXTURID 属性中， XA_FORMATID 字段将显示为 [00000000]。 从 IBM MQ 8.0 开始，如果没有与连接关联的 XA 事务，那么 XA_FORMATID 值将显示为空字符串 []。

相关信息
DISPLAY CONN
runmqsc

命令更改 :runmqsc 命令

runmqsc 命令具有两种新的客户机方式，并且提供了新的环境变量以允许您设置您选择的提示。

MQSC 的新客户机方式

MO72 SupportPac *MQSC Client for IBM MQ* 提供了具有两种不同操作方式的 MQSC 门户网站:

- 通过本地或客户机连接向队列管理器发出 MQSC 命令。
- 针对客户机通道表发出 MQSC 命令，而无需连接到队列管理器。

现在，通过为 **runmqsc** 提供两种新的客户机方式，将此功能整合到所有平台的 IBM MQ 中:

客户机-本地方式 (-n)
runmqsc 提供了一组有限的 MQSC 命令来管理客户机通道定义文件。

客户机连接方式 (-c)
runmqsc 通过客户机通道连接到远程队列管理器，并发出转义的 PCF 命令。

MO72 SupportPac 仍可供下载，以用于较旧版本的 IBM MQ。

新建 MQPROMPT 环境变量

从 版本 8.0 开始，通过使用 MQPROMPT 环境变量设置您选择的提示，可以更轻松地看到您处于 MQSC 环境中，并查看当前环境的一些详细信息。 有关更多信息，请参阅 使用 MQSC 命令执行本地管理任务。

相关信息
runmqsc

命令更改: 除去 migmbbrk 和 z/OS CSQUMGMB

IBM MQ 8.0 将除去 **migmbbrk** 和 **CSQUMGMB** 命令。 引入这些命令是为了将发布/预订配置迁移至主题和预订的 IBM WebSphere MQ 7.0 构造。

如果需要将现有发布/预订配置数据从 WebSphere Message Broker 版本 6.0 或 版本 6.1 迁移到 IBM MQ 8.0 ，那么必须首先升级到 IBM MQ 的 版本 7.0 发行版 (即 IBM MQ 版本 7.0.1， 版本 7.1 或 版本 7.5)，并遵循相关发行版的产品文档中的 **migmbbrk** 或 **CSQUMGMB** 迁移指示信息，然后再次升级到 IBM MQ 8.0。

IBM WebSphere MQ 7.0.1 的相关信息
米姆布尔克
迁移发布/预订配置实用程序 (CSQUMGMB)
IBM WebSphere MQ 7.1 的相关信息
米姆布尔克
迁移发布/预订配置实用程序 (CSQUMGMB)
IBM WebSphere MQ 7.5 的相关信息
米姆布尔克

命令级别: 更改

在 IBM MQ 版本 8.0 中，所有平台上的命令级别都更改为 800。 在某些平台上，它在 IBM MQ 8.0.0 修订包 2 中更改为 801 ，在 IBM MQ 8.0.0 修订包 3 中更改为 802。 V 8.0.0.4 对于 IBM MQ 8.0.0 修订包 4 中的那些平台，命令级别保持为 802。

命令级别 802

V 8.0.0.3

在以下平台上安装 IBM MQ V 8.0.0 修订包 3 时，此命令级别变为可用:

- UNIX
- IBM i

- Windows

如果要使用可插拔认证方法 (PAM) 授权，必须显式将命令级别更改为 802 ，如 使用 PAM 中所述

命令级别 801

V 8.0.0.2

在以下平台上安装 IBM MQ V 8.0.0 修订包 2 时，此命令级别变为可用:

- UNIX
- IBM i

如果要使用 LDAP 授权，必须显式将命令级别更改为 801 或更高版本，如 LDAP 授权概述中所述。

相关信息

MQCMDL_ * (命令级别)

CommandLevel (MQLONG)

不推荐: 与 libmqmcs 和 libmqmzse 库链接

如果 Solaris 上的 IBM MQ 检测到 版本 8.0 应用程序与库 libmqmcs 和 libmqmzse 链接，那么任何 MQCONN 或 MQCONNX 调用都将失败，并返回错误 MQRC_ENVIRONMENT_ERROR。 您必须重新链接应用程序以避免使用 libmqmcs 和 libmqmzse 库，因为不推荐使用与这些库的链接。

不推荐: Managed File Transfer Web 网关不推荐

对于 IBM MQ 8.0，不推荐使用 IBM MQ Managed File Transfer Web 网关。

在 IBM MQ 和 IBM MQ Managed File Transfer 的未来发行版中， IBM 预计不会对 IBM MQ Managed File Transfer Web 网关功能应用任何其他增强功能。

V 8.0.0.2 不推荐: SSLv3 协议

从 IBM MQ 8.0.0 修订包 2 开始，不推荐使用 SSLv3 协议和部分 IBM MQ CipherSpec。

IBM MQ 8.0.0 修订包 2 不推荐使用 SSLv3 协议并使用以下 IBM MQ CipherSpecs:

- AES_SHA_US
- RC4_SHA_US
- RC4_MD5_US
- TRIPLE_DES_SHA_US
- DES_SHA_EXPORT1024
- RC4_56_SHA_EXPORT1024
- RC4_MD5_EXPORT
- RC2_MD5_EXPORT
- DES_SHA_EXPORT
- NULL_SHA
- NULL_MD5
- FIPS_WITH_DES_CBC_SHA
- FIPS_WITH_3DES_EDE_CBC_SHA

如果要使用这些 CipherSpecs，请更改通道定义以使用使用 TLS 1.0 或 TLS 1.2 协议的备用 IBM MQ CipherSpec 。

如果需要继续使用 SSLv3 CipherSpecs，可以采用两种替代方法:

- 设置环境变量 AMQ_SSL_V3_ENABLE=1。

- 更改 qm.ini 文件中的 SSL 节以重新启用 SSLv3 CipherSpecs:

```
SSL:
AllowSSLV3=Y
```

可以使用 SECPROT 参数来显示通道上正在使用的安全协议。

相关信息
IBM MQ 中的 SSL 和 TLS 安全协议

V 8.0.0.3 不推荐: 较弱的密码算法

从 IBM MQ V 8.0.0 修订包 3 开始，不推荐使用对较弱密码算法的支持。

已对 IBM MQ 队列管理器进行了更改，以禁止使用现在被视为较弱的密码算法或协议配置 CipherSpecs 。

不推荐使用的 CipherSpecs 如下所示:

SSL V3
缺省情况下，不再启用所有 CipherSpecs 。

请参阅第 28 页的『不推荐: SSLv3 协议』以获取更多信息。

TLS 1.0

- TLS_RSA_EXPORT_WITH_RC2_40_MD5
- TLS_RSA_EXPORT_WITH_RC4_40_MD5
- TLS_RSA_WITH_DES_CBC_SHA
- TLS_RSA_WITH_NULL_MD5
- TLS_RSA_WITH_NULL_SHA
- TLS_RSA_WITH_RC4_128_MD5

TLS 1.2

- ECDHE_ECDSA_NULL_SHA256
- ECDHE_ECDSA_RC4_128_SHA256
- ECDHE_RSA_NULL_SHA256
- ECDHE_RSA_RC4_128_SHA256
- TLS_RSA_WITH_NULL_NULL
- TLS_RSA_WITH_NULL_SHA256
- TLS_RSA_WITH_RC4_128_SHA256

请参阅 CipherSpecs 在 IBM MQ 中受支持 ， 以获取有关受支持的 CipherSpecs 以及如何启用一个或多个不推荐的 CipherSpecs 的信息。

相关信息
IBM MQ 中的 SSL 和 TLS 安全协议

不推荐: IBM MQ 资源管理器服务定义向导

对于 IBM MQ 8.0，不推荐使用 IBM WebSphere MQ 7.0 中引入的 IBM MQ Explorer 服务定义向导。

该向导简化了为 IBM MQ 创建服务定义的过程。

不推荐: 针对 SOAP 客户机和 HTTP 网桥的 IBM MQ 传输

不推荐使用 JAX-RPC 的 IBM MQ Transport for SOAP Java 客户机和侦听器， IBM MQ Transport for SOAP .NET 1 和 2 以及 IBM MQ bridge for HTTP 。

IBM MQ Transport for SOAP Java 客户机

不推荐使用 JAX-RPC 的 IBM MQ Transport for SOAP Java 客户机和侦听器。 在 Java Platform , Enterprise Edition 6 中不推荐使用此客户机和侦听器所提供的功能，将其视为后级别的 JAX-RPC 标准。 此外，支持客户机和侦听器的 Apache Axis1 库的状态现在已停止使用。 请使用 IBM MQ transport SOAP Java client for JAX-WS 作为替代方法。

IBM MQ Transport for SOAP .NET 1 和 2

不推荐使用 IBM MQ Transport for SOAP .NET 1 和 2 客户机。 此客户机提供的功能被视为后级功能。 将 Microsoft Windows Communication Foundation (WCF) 的 IBM MQ 定制通道用作备用通道 (请参阅 使用 IBM MQ 开发 Microsoft Windows Communication Foundation (WCF) 应用程序)。

IBM MQ bridge for HTTP

不推荐使用 IBM MQ bridge for HTTP。 应当将 IBM Messaging Client for JavaScript 提供的 WebSockets 支持用作备用方法。

V 8.0.0.3 多实例队列管理器的扩展启动事件

从 IBM MQ V 8.0.0 修订包 3 开始，将扩展多实例队列管理器的启动事件，以允许系统监视应用程序查看多实例队列管理器何时发生故障以及现在在何处运行。

新的 MQRQ 值是允许的故障转移，不允许的故障转移，从备用数据库启动。 有关更多信息，请参阅 队列管理器活动。

IBM MQ 8.0 的 IBM i: 更改

IBM i 在 IBM MQ 8.0 中支持拆分集群传输队列和高级消息安全性。

有关更多信息，请参阅 在 IBM i 上安装 IBM MQ Advanced Message Security。

Java 和 JMS: 对字符串转换的更改

从 IBM MQ 8.0 开始，与 Java 和 JMS 客户机的字符串转换相关的一些缺省行为已更改。

在 版本 8.0 之前，找到任何不可翻译或格式不正确的数据都会导致这些字符被不可映射的字符替换，通常是? 字符结尾。

从 版本 8.0 开始，缺省行为已更改为通过抛出异常来报告不可映射的字符。 通过设置属性来修改行为，仍然可以替换字符。

相关信息

IBM MQ Java 类中的字符串转换

IBM MQ JMS 类中的字符串转换

Java 和 JMS: 对 CipherSuite 支持的更改

IBM MQ classes for Java 和 IBM MQ classes for JMS 中对 SSL/TLS CipherSuites 的支持已更改。

已从 Java 类中除去三个 Java CipherSuites , 并且不得再与基于 SSL 的 CipherSpecs 结合使用。 这三个 CipherSuite 现在只能与其对应的基于 TLS 的 IBM MQ CipherSpec 结合使用。 以下 CipherSuite 将受影响:

- SSL_RSA_WITH_3DES_EDE_CBC_SHA
- SSL_RSA_WITH_DES_CBC_SHA
- SSL_RSA_WITH_RC4_128_SHA

注意: MQ Explorer 将从 Java 类中获取其受支持的密码列表，因此 MQ Explorer 也不再支持这三个特定的密码。

有关更多信息，请参阅“相关链接”中的“SSL/TLS CipherSpec 和 CipherSuite”主题。

相关概念

第 31 页的『Java: 对 IBM MQ classes for Java 的更改』
对于 IBM MQ 8.0，已除去对 connector.jar 的不需要的依赖关系，并且已对公用类的 API 进行了更改。

相关参考

第 33 页的『JMS: 对 IBM MQ classes for JMS 的更改』
对于 IBM MQ 8.0，将对 IBM MQ classes for JMS 进行一些更改。

相关信息

指定 CipherSpec
IBM MQ classes for Java 中的 SSL/TLS CipherSpec 和 CipherSuite
IBM MQ classes for JMS 中的 SSL/TLS CipherSpec 和 CipherSuite

V 8.0.0.6 Java 和 JMS: 资源适配器更改

V 8.0.0.6 从 IBM MQ 8.0.0 修订包 6 开始， IBM MQ 资源适配器 IVT 应用程序将更新为支持 WildFly V10

已更新 IBM MQ 8.0 资源适配器安装验证测试应用程序，以便 `WMQ_IVT_MDB.jar` 文件包含 WildFly V10 使用的文件 `META-INF\jboss-ejb3.xml`。 `WMQ_IVT.war` 文件中的文件 `WEB-INF\jboss-web.xml` 包含 WildFly V10 的正确资源引用。

有关更多信息，请参阅 在 Wildfly 中安装和测试资源适配器。

Java: Java 7 的更改

为了支持 Java 7 ，已对 IBM MQ classes for Java 和 IBM MQ classes for JMS 进行了更改。

IBM MQ classes for Java 和 IBM MQ classes for JMS 现在是使用 Java 7 构建的。

Java 7 运行时环境支持运行较早的类文件版本。

Java: 对 IBM MQ classes for Java 的更改

对于 IBM MQ 8.0，已除去对 connector.jar 的不需要的依赖关系，并且已对公用类的 API 进行了更改。

已除去对 connector.jar 的不需要的依赖关系

使用 JCA 会导致对已除去的 `connector.jar` 文件产生不需要的依赖关系。 此移除操作的影响如下所示:

- `javax.resource.ResourceException` 及其子类不再在 IBM MQ classes for Java 中抛出或捕获。 将抛出 `MQException` 以代替 `ResourceException`。
- IBM MQ classes for Java 中不再使用 `javax.resource.spi.ConnectionRequestInfo` 。 `MQConnectionRequestInfo` 抽象类及其子类在先前使用了 `ConnectionRequestInfo` 的上下文中使用。
- IBM MQ classes for Java 中不再使用 `javax.resource.spi.ConnectionManager` 。 `MQConnectionManager` 接口在先前使用了 `ConnectionRequestInfo` 的上下文中使用。

对公共类的 API 的更改

- `com.ibm.mq.MQEnvironment`:
  - 已除去 `setDefaultConnectionManager(ConnectionManager)` 方法。
  - `getDefaultConnectionManager()` 方法现在返回 `MQConnectionManager`。
- `com.ibm.mq.MQQueueManager`:
  - 已除去 `MQQueueManager(String, ConnectionManager)` 构造函数。 它将替换为现有 `MQQueueManager(String, MQConnectionManager)` 方法。
  - 已除去 `MQQueueManager(String, Hashtable, ConnectionManager)` 构造函数。 它将替换为现有 `MQQueueManager(String,Hashtable, MQConnectionManager)` 方法。

  - 已除去 MQQueueManager(String, int, ConnectionManager) 构造函数。 它将替换为现有 MQQueueManager(String, int, MQConnectionManager) 方法。
- com.ibm.mq.MQSimpleConnectionManager:
  - 已除去 allocateConnection(MQManagedConnectionFactory,ConnectionRequestInfo) 方法。 它将替换为 allocateConnection(MQManagedConnectionFactory,MQConnectionRequestInfo) 方法，该方法会抛出 MQException 来代替 ResourceException 以指示失败。
  - 已除去 createConnection(MQManagedConnectionFactory,ConnectionRequestInfo) 方法。 它将替换为 createConnection (MQManagedConnectionFactory,MQConnectionRequestInfo) 方法，该方法会抛出 MQException 来代替 ResourceException 以指示失败。
  - 已除去 recycleConnection(MQManagedConnectionFactory,ConnectionRequestInfo) 方法。 它将替换为 recycleConnection(MQManagedConnectionFactory,MQConnectionRequestInfo) 方法，该方法会抛出 MQException 来代替 ResourceException 以指示失败。

对 CipherSuite 支持的更改

请参阅第 30 页的『Java 和 JMS: 对 CipherSuite 支持的更改』，以了解更多信息。

相关概念

第 36 页的『安全性: 对 CipherSuite 支持的更改』
对于 IBM MQ 8.0，已扩展先前发行版中已提供的 SHA-2 支持。

Java: 对 Java 跟踪行为的更改

对于 IBM MQ 8.0 ，已更新用于控制 Java-language 客户机的跟踪的方法。

现在， traceControl 工具提供了用于控制 Java 跟踪行为的功能。 有关更多详细信息，请参阅 使用 IBM MQ classes for Java 和 IBM MQ classes for JMS 控制正在运行的进程中的跟踪 。

Java: 对 MQException 和 MQDataException 类日志记录的更改

已除去 IBM MQ 8.0 的 com.ibm.mq.MQException 类的日志记录功能，该类将新 MQExceptions 的详细信息写入日志。

com.ibm.mq.MQException 类历来将新 MQExceptions 的详细信息写入日志。 缺省日志为 System.err，但此缺省值可能被应用程序代码覆盖。 要记录的异常代码可以通过同一类上的方法进行定制，以便可以排除和包含特定代码。

虽然已除去日志记录功能，但已保留相关静态公共字段 MQEnvironment.log 以及方法 MQEnvironment.logInclude(整数) 和 MQEnvironment.logExclude(整数) 以与现有代码兼容，但没有任何效果。

已以相同方式修正 com.ibm.mq.headers.MQDataException 类。

V 8.0.0.2 JMS: 对 "字节" 和 "流" 消息状态的更改

为了支持 Java Message Service 2.0，已对 IBM MQ 8.0.0 修订包 2 进行了更改。

JMS 2.0 规范规定，刚刚发送的 "字节" 或 "流" 消息应处于只读状态。 从 IBM MQ 8.0.0 修订包 2 开始，可以设置 com.ibm.msg.client.jms.ByteStreamReadOnlyAfterSend= xxxx (其中 xxxx 为 TRUE 或 FALSE) 的 jms.config 文件中的系统属性或条目，以定义 IBM MQ classes for JMS 是应该将刚刚发送的消息的状态设置为只读还是只读:

- TRUE 表示对象在发送后设置为只读。
- FALSE 表示对象设置为仅在发送后写入。 这是缺省值。

有关更多信息，请参阅 用于配置 JMS 客户机行为的属性。

有关 JMS 2.0 的更多详细信息，请参阅 JMS 2.0 规范 (Java.net)。

V 8.0.0.2 JMS: 版本 8.0 中的异常侦听器更改

在 IBM MQ 8.0 中，使用 IBM MQ classes for JMS 时，将更改调用应用程序的 JMS ExceptionListener 的方式。

V 8.0.0.2 从 IBM MQ 8.0.0 修订包 2 开始，缺省情况下，IBM MQ classes for JMS 只会在连接中断时才会将异常发送到异常侦听器，这种做法符合 JMS 规范。 要让 IBM MQ classes for JMS 将所有异常通知到异常侦听器，您的应用程序必须将连接工厂的 ASYNCEXCEPTION 属性设置为 ASYNC_EXCEPTIONS_ALL。

V 8.0.0.6 APAR IT14820(包含在 IBM MQ 8.0.0 修订包 6 中) 更新了 IBM MQ classes for JMS ， 以便:

- 无论应用程序是使用同步还是异步消息使用者，都将针对任何连接中断异常调用由应用程序注册的 ExceptionListener 。
- 如果 JMS 会话使用的 TCP/IP 套接字中断，将调用由应用程序注册的 ExceptionListener。
- 当应用程序使用异步消息使用者，并且应用程序使用的 JMS ConnectionFactory 将 ASYNC_EXCEPTIONS 属性设置为 ASYNC_EXCEPTIONS_ALL 值时，在消息传递期间发生的非连接中断异常（例如，MQRC_GET_INHIBITED）将传至该应用程序的 ExceptionListener。

注: 仅为连接中断异常调用一次 ExceptionListener，即使两个 TCP/IP 连接（一个由 JMS 连接使用，一个由 JMS 会话使用）均已中断。

相关信息

IBM MQ classes for JMS 中的异常

ASYNCEXCEPTION

JMS: 对 IBM MQ classes for JMS 的更改

对于 IBM MQ 8.0，将对 IBM MQ classes for JMS 进行一些更改。

强制实施唯一客户机标识

根据 JMS 2.0 规范的要求， IBM MQ classes for JMS 现在检查建立连接时提供的非空客户机标识是否唯一。否则，将抛出 `javax.jms.InvalidClientIDException` 。

某些依赖于先前行为且复用了客户机标识的现有应用程序现在可能会失败。 要允许此类应用程序继续运行而不进行编码更改，可以通过在启动应用程序时将系统属性 `com.ibm.mq.jms.SupportMQExtensions` 设置为 `true` 来禁用对客户机标识唯一性的检查。

对 CipherSuite 支持的更改

请参阅第 30 页的『Java 和 JMS: 对 CipherSuite 支持的更改』，以了解更多信息。

除去对 `Direct` 和 `DirectHTTP` 传输的支持

在 IBM MQ 的先前发行版中不推荐使用 `Direct` 和 `DirectHTTP` 传输，并且已在 IBM MQ 8.0 中除去这些传输。

虽然已在 IBM MQ 8.0 中除去直接 IP 传输，但您仍可以使用 MQ Explorer 或 JMS 管理工具或以编程方式创建和管理专门用于直接 IP 传输的连接工厂。 这意味着您可以更新 IBM MQ 8.0 上的 JNDI 连接工厂，该连接工厂可能由仍支持直接 IP 传输的较早版本使用。 但是，如果尝试创建 版本 8.0 连接，那么会生成以下异常:

```
JMSFMQ1006: The value of 'DirectIP' for property 'Transport Type' is not valid.
```

相关概念

第 36 页的『安全性: 对 CipherSuite 支持的更改』
对于 IBM MQ 8.0，已扩展先前发行版中已提供的 SHA-2 支持。

相关信息

SupportMQExtensions

JMS: 对 JMS 2.0

为了支持 Java Message Service 2.0，已对 IBM MQ 8.0 进行了更改。

IBM MQ 8.0 支持 JMS 2.0 版本的 JMS 标准。 有关更多详细信息，请参阅位于 Java.net 的 JMS 2.0 规范。

任何要与 IBM MQ 一起使用的代码都必须使用产品随附的 `jms.jar` 文件。 对于 IBM MQ 8.0，此 `jms.jar` 文件现在是一组 JMS 2.0 接口。

已除去先前版本所使用的多个 JAR 文件:

- 已除去 `CL3Export.jar` 和 `CL3Nonexport.jar` 。
- 已除去 `dhbcore.jar` 和 `connector.jar` 。
- 已除去 `ldap.jar`， `jndi.jar` 和 `jta.jar` ，因为它们现在是 JDK 的一部分

JMS: 对 PROVIDERVERSION 属性的更改

JMS **PROVIDERVERSION** 属性选择 Java 应用程序是使用排队的命令消息接口发布和预订，还是使用集成的调用接口发布和预订。 对于 IBM MQ 8.0 ，除了正常方式和迁移方式外，您还可以选择具有限制的正常方式。

对于 IBM MQ 8.0，可以将连接工厂 **PROVIDERVERSION** 属性设置为以下任何值:

8 - 正常方式
JMS 应用程序使用正常方式。 此方式使用 IBM MQ 队列管理器的所有功能来实现 JMS。

7 - 存在限制的正常方式
JMS 应用程序使用具有限制的正常方式。 此方式使用 JMS 2.0 API，但不使用诸如共享预订、延迟传递或异步发送之类的新功能。

6-迁移方式
JMS 应用程序使用迁移方式。 在迁移方式下， IBM MQ classes for JMS 使用与 IBM WebSphere MQ 6.0 随附的功能部件和算法类似的功能部件和算法。

unspecified (缺省值)
JMS 客户机使用算法来确定使用哪种操作方式。

相关信息

配置 JMS **PROVIDERVERSION** 属性

对集群主题行为的更改

在此发行版中，发布时行为已更改 (**PUB**) 参数在相同名称的集群主题定义中不同。

如果在集群中的多个队列管理器上定义了相同的指定受管主题对象，并且每个受管主题对象都设置了 CLUSTER 参数，那么所有集群主题对象参数的设置通常应在所有定义之间匹配。 如果任何设置不匹配，那么这可能会导致集群中的不同队列管理器使用主题参数的不同设置，从而观察到非确定性行为。

因此，检测到不匹配时，会将以下警告消息写入每个队列管理器的错误日志:

- distributed IBM i 在分布式系统和 IBM i 上，AMQ9465 和 AMQ9466。
- z/OS 在 z/OS 系统上，CSQX465I 和 CSQX466I。

在此发行版之前，如果 **PUB** 参数的设置因定义而异 (例如，如果一个设置为 `ENABLED` ，另一个设置为 `DISABLED`)，那么集群中的每个队列管理器都单独选择一个要使用的值。 这导致在整个集群中接受或拒绝发布是非确定性的。

通过在此发行版中引入 发布/预订集群的主题主机路由 ，已修改先前描述的行为。 如果至少有一个将发布参数设置为 `ENABLED` (或设置为 `ASPARENT` 并以本地方式解析为 `ENABLED`) 的集群主题定义，那么现在将接受发布。 仅当集群中主题的所有定义都设置为 `DISABLED` 时，才会拒绝发布。

在此发行版中，以下行为保持不变: 如果存在本地定义的受管主题对象 (无论是否设置了集群属性)，那么所有参数设置 (包括 **PUB** 参数的设置) 都将从本地定义中获取，而不是从远程队列管理器的集群主题定义中获取。

有关更多信息，请参阅 同名的多个集群主题定义。

发布/预订: 对 DISPLAY TOPIC 和 DISPLAY TSTATUS 输出的更改

从 IBM MQ 8.0 开始， **DISPLAY TOPIC** 命令的输出现在包含参数 **CLROUTE** 和 **CLSTATE**， **DISPLAY TPSTATUS** 命令现在包含参数 **CLROUTE**。

这些更改与 发布/预订集群的主题主机路由相关联。

请参阅 DISPLAY TOPIC 和 DISPLAY TPSTATUS 以获取更多信息。

发布/预订: 对预订选择字符串求值方式的更改

在此发行版中，将在由发布者发送的消息上进行发布/预订消息传递的消息选择，而不是在最终由订户接收的消息上进行选择。

对于具有 **SelectionString** 的预订，现在会根据发布程序放置的消息对字符串进行求值，然后再修改该消息以将其传递到每个订户。

此行为更改可能会影响具有选择字符串的订户，这些字符串引用在发布操作期间修改的字段。 例如:

- MQMD 字段 UserIdentifier， MsgId 和 CorrelId
- 消息属性，例如 UserData。

相关信息

选择字符串

选择字符串规则和限制

发布/预订: 使用 PROXYSUB (FORCE) 对代理预订传播进行的更改

在此发行版中，当您在参数 **PROXYSUB** 设置为 FORCE 的情况下配置主题对象时， IBM MQ 会自动禁用主题树中低于此点的主题字符串的个别代理预订。

当 **PROXYSUB** 设置为 FORCE 时，将生成以下配置:

- 将创建与此主题对象的主题字符串以及主题树中此点下方的所有主题字符串相匹配的通配符预订。
- 此通配符预订将传播到发布/预订拓扑中的所有相邻队列管理器。

在先前发行版中，将 **PROXYSUB** 设置为 FORCE 不会自动阻止为主题树中此点以下的主题字符串生成和传播个别代理预订。 在当前发行版中，将 **PROXYSUB** 设置为 FORCE 将自动禁用主题树中低于此点的主题字符串的个别代理预订。

在先前发行版中，要减少代理预订数，请将 **pscProxySubFlags** 调整参数设置为 1。 此设置显式禁用所有代理预订 (通配符 PROXYSUB(FORCE) 预订除外)。 在当前发行版中，不再需要设置 **pscProxySubFlags** 参数。

注: 不应在不同集群或层次结构流中的另一主题对象下配置主题层次结构中的一个主题对象。 这是因为跨此类发布/预订网络的发布流可能不明确，尤其是与设置 **PROXYSUB** 为 FORCE 时。 由于先前发行版传播了所有代理预订，而不考虑 **PROXYSUB** 设置，因此无法检测到此类配置的实例。 通过此发行版中的新行为，发布可能不会流向较低主题对象的集群或流。 如果检测到此类配置，那么会将警告消息写入日志。 警告消息标识取决于平台:

- IBM i distributed 在分布式系统和 IBM i 上， AMQ5824 或 AMQ5968
- z/OS 在 z/OS 上， CSQT824I 或 CSQT968I

要纠正此问题，请修改主题层次结构以除去嵌套的集群或流，或者将这两个对象都更改为具有相同的 **PROXYSUB** 设置。

有关使用 PROXYSUB(FORCE)时的性能注意事项，请参阅 发布/预订网络中的预订性能。

V 8.0.0.6 发布/预订: 尝试更改时未将代理预订修改为 ADMIN

不能修改 SUBTYPE 为 PROXY 的预订。

从 IBM MQ 8.0.0 修订包 6 开始，如果尝试修改代理预订，那么将报告一条错误消息，并且不会将 SUBTYPE 修改为 ADMIN。 请参阅 DISPLAY SUB 和 ALTER SUB。

队列管理器: 更改为缺省 TCP 缓冲区大小

对于 IBM MQ 8.0，对新队列管理器的发送和接收缓冲区大小的 TCP 节值进行了更改，在没有手动调整的情况下，先前缺省为固定大小的 32Kb 缓冲区。 z/OS 此更改不适用于 z/OS 。

现在，将使用 qm.ini 中的以下设置自动创建新的队列管理器:

```
TCP:
SndBuffSize=0
RcvBuffSize=0
RcvSndBuffSize=0
RcvRcvBuffSize=0
ClntSndBuffSize=0
ClntRcvBuffSize=0
SvrSndBuffSize=0
SvrRcvBuffSize=0
```

因此，对于 IBM MQ 8.0，缺省 TCP 发送和接收缓冲区大小设置为由新队列管理器的操作系统管理。 允许操作系统管理 TCP 缓冲区可以显着提高性能，例如通过 TCP 自动调整。

保留从较早发行版迁移的队列管理器的 TCP 发送和接收缓冲区设置。 有关如何手动设置 TCP 缓冲区大小以使用已迁移队列管理器的操作系统缺省行为的详细信息，请参阅 TCP， LU62， NETBIOS 和 SPX。

队列管理器: 为集群中的队列管理器添加的版本属性

从 IBM MQ 8.0 开始，可以显示与集群队列管理器关联的 IBM MQ 版本，格式为 VVRRMMFF， 其中 VV 是版本， RR 是发行版， MM 是维护级别， FF 是修订级别。

安全性: 证书撤销检查

如果客户机连接调用使用 SSL/TLS 通道，那么可以使用客户机配置文件的 SSL 节中的 **ClientRevocationChecks** 属性来确定如何配置证书撤销检查。

请参阅 客户机配置文件的 SSL 节。

安全性: 对 CipherSuite 支持的更改

对于 IBM MQ 8.0，已扩展先前发行版中已提供的 SHA-2 支持。

扩展的 SHA-2 支持

为以下平台提供了基本 SHA-2 支持:

- Windows 或 UNIX 和 Linux (自 IBM WebSphere MQ 7.0.1.4 起由基本队列管理器 SSL/TLS 支持提供)
- IBM i (在 IBM WebSphere MQ 7.1 中添加)
- z/OS z/OS (在 IBM WebSphere MQ 7.1 中添加)

针对以下组件 (从 IBM WebSphere MQ 7.1.0 修订包 3 和 IBM WebSphere MQ 7.5.0 修订包 2) 提供了基本 SHA-2 CipherSuite 支持:

- MQ Explorer
- Java/JMS
- 遥测
- Managed File Transfer 组件

SHA-2 支持已扩展为包含对以下组件的完整 SHA-2 CipherSuites 集合的支持 (对于 IBM MQ 8.0):

- MQ Explorer
- Java/JMS
- 遥测
- Managed File Transfer 组件。

CipherSuite 不再受支持

不再支持以下 CipherSuite:

- SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5

互操作性方面的限制

已从 Java 类中除去三个 Java CipherSuites ，并且不得再与基于 SSL 的 CipherSpecs 结合使用。 这三个 CipherSuite 现在只能与其对应的基于 TLS 的 IBM MQ CipherSpec 结合使用。 以下 CipherSuite 将受影响:

- SSL_RSA_WITH_3DES_EDE_CBC_SHA
- SSL_RSA_WITH_DES_CBC_SHA
- SSL_RSA_WITH_RC4_128_SHA

注意: MQ Explorer 将从 Java 类中获取其受支持的密码列表，因此 MQ Explorer 也不再支持这三个特定的密码。

有关更多信息，请参阅“相关链接”中的“SSL/TLS CipherSpec 和 CipherSuite”主题。

相关概念

第 31 页的『Java: 对 IBM MQ classes for Java 的更改』
对于 IBM MQ 8.0，已除去对 connector.jar 的不需要的依赖关系，并且已对公用类的 API 进行了更改。

相关参考

第 33 页的『JMS: 对 IBM MQ classes for JMS 的更改』
对于 IBM MQ 8.0，将对 IBM MQ classes for JMS 进行一些更改。

相关信息

指定 CipherSpec
Managed File Transfer 的 SHA-2 CipherSpec 和 CipherSuite
关于将 SHA-2 密码套件用于 MQTT 通道的系统需求
关于将 SHA-2 密码套件用于 MQTT 客户机的系统需求
IBM MQ classes for Java 中的 SSL/TLS CipherSpec 和 CipherSuite
IBM MQ classes for JMS 中的 SSL/TLS CipherSpec 和 CipherSuite

安全性: 新的 CONNAUTH CHCKLOCL 参数

在 IBM MQ 8.0 中，引入了新的队列管理器安全性参数 **CONNAUTH CHCKLOCL**。

属性 **CHCKLOCL** 用于开启本地连接的用户标识和密码检查。

在 IBM WebSphere MQ 7.1 与最新版本之间进行迁移时，每个队列管理器上的 **CONNAUTH CHCKLOCL** 属性都将设置为 NONE，从而确保版本与版本之间的连续性，但会关闭连接认证。

对于新的 IBM MQ 8.0 安装， **CONNAUTH CHCKLOCL** 属性设置为 OPTIONAL。 这意味着不需要用户标识和密码，但如果提供了用户标识和密码，那么它们必须是有效的对，否则将被拒绝。

有关更多信息，请参阅 连接认证: 配置。

安全性: UNIX 和 Linux 系统上基于 OAM 用户的许可权

在 UNIX 和 Linux 系统上，对象权限管理器 (OAM) 可以使用基于用户的授权以及基于组的授权。

在 版本 8.0 之前， UNIX 和 Linux 上的访问控制表 (ACL) 仅基于组。 从 版本 8.0 开始， ACL 基于用户标识和组，您可以使用基于用户的模型或基于组的模型进行授权，方法是将 **SecurityPolicy** 属性设置为相应的值，如 配置可安装服务 和 在 UNIX 和 Linux 上配置授权服务节中所述。

行为更改

使用基于用户的策略运行时，某些命令会返回与 IBM MQ 的先前发行版不同的信息。

- **dmpmqaut** 和 **dmpmqcfg** 命令显示基于用户的记录， PCF 等效操作也是如此。
- MQ Explorer 的 OAM 插件显示基于用户的记录并允许基于用户的修改。
- OAM **Inquire** 函数返回显示其支持用户的结果。

在 qm.ini 文件中启用基于用户的权限时，对 **setmqaut** 命令使用 **-p** 属性不会授予同一主组中的所有用户访问权，如 服务节格式中所述。

如果您开始采用基于用户的授权并且拥有许多用户，那么与基于组的模型相比， AUTH 队列上存储的记录可能更多，并且授权过程可能需要比先前更长的时间，因为要验证的记录更多。 预计这一增幅不会很大。 如果需要，您可以混合使用用户和组许可权。

迁移注意事项

如果将现有队列管理器的模型从组更改为用户，那么不会立即生效。 已经做出的授权继续适用。 连接到队列管理器的任何用户都将获得与之前相同的特权: 其标识所属的所有组的组合。 当针对用户标识发出新的 **setmqaut** 命令时，这些命令将立即生效。

如果使用用户策略创建新的队列管理器，那么此队列管理器将仅对创建该队列管理器的用户具有许可权 (通常为 mqm 用户标识，但不一定为该用户标识)。 还将自动向 mqm 组授予许可权。 但是，如果您没有 mqm 作为主组，那么 mqm 组将不会包含在初始授权集中。

如果从用户移动到组策略，那么不会自动删除基于用户的权限。 但是，在许可权检查期间不再使用这些参数。 在还原策略之前，保存当前配置，更改策略，重新启动队列管理器，然后重放脚本。 因为它现在是基于组的队列管理器，所以效果是基于主组存储用户标识规则。

相关信息

对象权限管理器 (OAM)
主体和组
队列管理器配置文件: 服务节格式
crtmqm (创建队列管理器) 命令

安全性: 在客户机上禁用 IBM MQ AMS

您可以在客户机上禁用 IBM MQ AMS ，以防止在连接到正在 IBM WebSphere MQ 7.1 或更低版本上运行的队列管理器时发生错误。 从 IBM MQ 8.0 开始，将更改用于对 C 客户机禁用 IBM MQ AMS 的环境变量。

在 IBM WebSphere MQ 7.5 中，可以按如下所示禁用 IBM WebSphere MQ Advanced Message Security :

- 对于 Java 客户机，请通过以下任一方式:
  - 通过设置环境变量 AMQ_DISABLE_CLIENT_AMS。
  - 通过设置 Java 系统属性 com.ibm.mq.cfg.AMQ_DISABLE_CLIENT_AMS。
  - 通过使用 DisableClientAMS 属性，在 IBM WebSphere MQ 7.5.0 修订包 5 中的 mqclient.ini 文件中的 **Security** 节下。
- 对于 C 客户机，请采用以下任一方式:
  - 从 IBM WebSphere MQ 7.5.0 修订包 4 设置环境变量 AMQ_DISABLE_CLIENT_AMS。
  - 使用 DisableClientAMS 属性，在 IBM WebSphere MQ 7.5.0 修订包 5 的 mqclient.ini 文件中的 **Security** 节下。

从 IBM MQ 8.0 开始，可以按如下所示禁用 IBM MQ AMS :

- 对于 Java 客户机，请通过以下任一方式:

  - 通过设置环境变量 AMQ_DISABLE_CLIENT_AMS。
  - 通过设置 Java 系统属性 com.ibm.mq.cfg.AMQ_DISABLE_CLIENT_AMS。
  - 通过使用 DisableClientAMS 属性，在 `mqclient.ini` 文件中的 **Security** 节下。
- 对于 C 客户机，请采用以下任一方式:
  - 通过设置环境变量 MQS_DISABLE_ALL_INTERCEPT。
  - 通过使用 DisableClientAMS 属性，在 `mqclient.ini` 文件中的 **Security** 节下。

相关信息

在客户机上禁用 Advanced Message Security

安全性: 客户机应用程序上的密码保护

从 IBM MQ 8.0 开始，您使用 MQCSP 结构发送的任何密码都可以受到保护，可以使用 IBM MQ 功能进行保护，也可以使用 SSL/TLS 加密进行加密。

z/OS 在 z/OS 上，通道启动程序地址空间使用集成加密服务设施 (ICSF) 来提高与客户机应用程序配合使用的密码保护算法的安全性。

请参阅 IBM MQ MQCSP 密码保护。

z/OS 请参阅 使用集成加密服务设施 (ICSF) 以获取更多信息。

安全性: 减少 SSL 和 TLS 的通道发送出口缓冲区空间

SSL 或 TLS 通道发送出口可由 `MQCXP ExitSpace` 字段保留的传输缓冲区中的最大空间量已减少 924 个字节至 14,328 个字节。

更改原因

协议 SSL 和 TLS 增加了它们的处理器使用率，同时 GSKit 现在更严格地实施限制; 版本 8.0 不容许过大的数据缓冲区。 如果具有在 SSL 或 TLS 通道上保留大量缓冲区空间的发送出口，那么在迁移后可能无法启动，发生错误 AMQ9186。 对于只有一端正在运行最新发行版的 IBM MQ 的通道，这可能是一个特殊问题。

该限制来自 RFC 6101 和相关 TLS 标准。 这些标准指定必须将 SSL 和 TLS 通道的数据分段为不超过 16Kb 的记录。 记录限制不会影响 IBM MQ 消息的最大大小; IBM MQ 了解底层传输协议需求，并在传输之前自动对每条消息进行分段，然后在远程端重新组装。 IBM MQ 仍然支持像在先前发行版中一样跨加密通道发送大型消息。

GSKit 现在严格实施记录层限制，这会影响内部 IBM MQ 发送缓冲区的大小。 由于通道发送出口有权访问发送缓冲区，因此它们受更改影响。 先前的 GSKit 版本未严格限制记录大小，如果它们的总大小超过 16Kb 限制，那么将在内部对 IBM MQ 数据流进行分段。 在 版本 8.0 中，行为已更改为对 SSL 和 TLS 记录实施严格的 16Kb 限制。 SSL 和 TLS 通道的传输段的最大大小已减小到符合 GSKit 限制。 这将使可供发送出口使用的缓冲区空间少于前发行版中的缓冲区空间。 新的发送出口缓冲区限制为 14,328 字节，允许 SSL 和 TLS 协议开销，还允许至少传输 IBM MQ 消息数据的 1Kb 。 大多数发送出口不需要这么多缓冲区空间，因此它们不受影响。 您应该尝试避免在 SSL 和 TLS 通道上保留大量发送出口缓冲区空间，因为如果出口正在使用大部分发送缓冲区空间，可能会发生性能问题。

Solaris 客户机: 安装方法中的更改

在 IBM MQ 8.0 中，已除去 *.img* 文件。 将以未打包的形式交付客户机。

而不是使用以下命令:

```
pkgadd -d ${image_dir}/client.img
```

现在使用以下命令:

```
pkgadd -d ${image_dir}
```

${image_dir} 中填充的 *mqm* 目录是安装包。 此格式与服务器映像的格式相匹配。

相关信息

在 Solaris 上安装 IBM MQ 客户机

遥测: 对 IBM MQ Telemetry Client Software Development Kit (SDK) 的更改

从 IBM MQ 8.0 开始，不再作为产品的一部分提供 IBM MQ Telemetry SDK。

此 SDK 先前包含用于创建自己的遥测应用程序的 MQTT 客户机库以及 第 249 页的『设备的 MQTT 守护程序』。

现在， SDK 的当前版本可作为免费下载 IBM Messaging Telemetry Client SupportPac 提供。

IBM MQ 8.0 的 Windows: 更改

对于 IBM MQ 8.0，已更改服务器和 64 位客户机安装的缺省安装位置。 所有安装的缺省数据路径中也有更改。 Windows 上用于 IBM MQ 8.0 的编译器已更改。

对于 IBM MQ 8.0，服务器和 64 位客户机安装的缺省安装位置以及所有安装的缺省数据路径已更改。 但是，如果要从先前发行版进行迁移，请注意迁移将保留旧路径。

Windows 服务器安装

对于 IBM MQ 8.0， Windows 的服务器安装已更改，如下所示:

- 对于 Windows 64 位系统，将 IBM MQ 的新 64 位版本安装到 `C:\Program Files\IBM\WebSphere MQ` 中，并将数据目录放入 `C:\ProgramData\IBM\MQ` 中。
- `bin` 目录包含 32 位二进制文件。
- `bin64` 目录包含 64 位二进制文件。
- `tools\lib` 目录包含用于开发的 32 位库。
- `tools\lib64` 目录包含用于开发的 64 位库。
- IBM MQ.NET 组合件位于 `bin` 文件夹中。
- C 和 C++ 的样本现在编译为 64 位。 现在， `tools\c\samples\bin64` 文件夹中提供了 64 位 C 语言样本。 `tools\c\samples\bin` 文件夹中还提供了一些 32 位样本。 现在， `tools\cplus\samples\bin64\vn` 文件夹中提供了 C++ 样本应用程序。

Windows 队列管理器

IBM MQ 8.0 for Windows 队列管理器运行 64 位进程，并支持 32 位和 64 位应用程序。 在先前版本中，Windows 上的队列管理器仅由 32 位进程组成，即使队列管理器在 64 位 Windows 平台上运行也是如此。

命令，二进制文件等的位置在 版本 8.0 中已更改。 所有命令现在都位于 `bin64` 文件夹中。 `bin` 文件夹包含 32 位库。 可以从 `bin` 或 `bin64` 文件夹执行队列管理器命令。

必须将 64 位 Windows 上的队列管理器出口编译为 64 位。 必须先重新编译任何 32 位队列管理器出口，然后才能将其与 64 位队列管理器配合使用。

在 Windows 任务管理器中标识流程

在 64 位 Windows 平台上，可以在 Windows 任务管理器列出的流程中轻松识别产品较早发行版的 IBM MQ 8.0 流程和流程。 由于 IBM MQ 8.0 之前发行版的队列管理器是 32 位的，因此 版本 8.0 之前发行版的所有队列管理器进程都具有与其进程名称相关联的 "* 32"。 在 版本 8.0 中，队列管理器为 64 位，因此 Windows 任务管理器中的队列管理器进程没有关联的 "* 32"。

Microsoft Visual Studio 编译器的更改

对于 IBM MQ 8.0，已使用 Microsoft Visual Studio 2012 C/C++ 编译器编译队列管理器和客户机库。 产品随附的样本也是使用 Microsoft Visual Studio 2012 C/C++ 编译器构建的。

V 8.0.0.2 在 IBM MQ 8.0 之前构建的所有应用程序都应该继续使用 64 位队列管理器。 但是，在 IBM MQ 8.0.0 修订包 2 之前，对用于 IBM MQ 8.0 的编译器的更改意味着需要重建所有使用 C++ 对象接口的应用程序，因为 Microsoft Visual Studio 2012 C++ 标准库不向后兼容。 使用 C 接口的应用程序不受影响。

V 8.0.0.2 从 V 8.0.0 修订包 2 开始，IBM MQ 提供了使用 Microsoft Visual Studio 2005 C++ 编译器构建的 C++ 客户机库。 这些库是对使用 Microsoft Visual Studio 2012 C++ 编译器构建的现有 IBM MQ 8.0 C++ 库的补充。 使用 IBM MQ 早于 版本 8.0 的发行版构建的应用程序可以使用这些库，因为 Microsoft Visual Studio 2005 C++ 标准库向后兼容。

V 8.0.0.4 从 V 8.0.0 修订包 4 开始， IBM MQ 提供了使用 Microsoft Visual Studio 2012 C++ 编译器构建的其他 C++ 客户机库。 这些附加库的命名与使用 Microsoft Visual Studio 2012 C++ 编译器构建的现有 IBM MQ 8.0 C++ 库不同。 用于这些附加库的不同命名意味着您可以在同一机器上运行使用 Microsoft Visual Studio 2012 编译的 IBM MQ C++ 应用程序和使用 Microsoft Visual Studio 2005 编译的应用程序。

注: 包含 IBM MQ C++ 库的 `VS2005` 文件夹仅安装在 64 位 Windows 客户机的 `bin` 目录中。 未针对 32 位 Windows 客户机安装安装 `VS2005` 文件夹。

将修订包应用于 Windows

V 8.0.0.4

您可以使用修订包来修补服务器，如先前发行版中一样。

要点: 但是，通过使用 MSI 次升级 (在顶部安装) 来修补 32 位和 64 位客户机。 不支持使用服务器修订包对这些客户机进行补丁。

有关受支持的安装方法的完整详细信息，请参阅 Windows: 在 IBM MQ 服务器上应用维护级别升级 和 Windows: 在 IBM MQ 客户机上应用维护级别升级。

相关信息

IBM MQ 8.0 上 Windows 的更改简介
在 Windows 上构建 C++ 程序

Windows: 程序和数据目录位置

在安装 IBM MQ 8.0 时，程序数据和二进制文件的位置取决于您是否正在运行全新安装，是否已卸载先前版本，以及是否正在使用 32 位操作系统或 64 位操作系统。

首次安装

首次安装 IBM MQ 时，如果选择定制安装选项，那么可以选择 IBM MQ 产品的位置以及 IBM MQ 数据和日志的位置。

在 IBM MQ 8.0 之前，缺省情况下会将数据和日志放入第一个安装所在的目录中。

在 版本 8.0 处，缺省位置已更改。 缺省情况下，产品二进制文件将放入 `C:\Program Files\IBM\WebSphere MQ` 中，缺省情况下，产品数据和日志将放入 `C:\ProgramData\IBM\MQ` 中。

后续安装和重新安装

在指定了数据目录之后，在任何安装的安装过程中，都不能对后续安装进行更改。 仅当 IBM MQ 安装在 64 位操作系统上时，才会将其作为 64 位版本进行安装。

在 版本 8.0 中，缺省数据目录为 `C:\ProgramData\IBM\MQ`，除非计算机上存在先前版本的 IBM MQ ，在这种情况下，新安装将继续使用现有数据目录。

现有 IBM WebSphere MQ 7.5 或 版本 7.1 安装

有三个可能的升级路径:

- 卸载 版本 7.1 或 版本 7.5 并安装 版本 8.0。
- 安装 版本 8.0 并要求其升级 版本 7.1 或 版本 7.5。

- 与 版本 7.1 或 版本 7.5 一起安装 版本 8.0，然后卸载 版本 7.1 或 版本 7.5。

安装 版本 7.5 或 版本 7.1 时，缺省情况下会将程序二进制文件和数据都安装到 C:\Program Files（x86）\WebSphere MQ 中。

卸载 版本 7.5 或 版本 7.1 时，有关数据目录位置的信息将保留在注册表中。 在卸载 版本 7.5 或 版本 7.1 之后以及在安装 版本 8.0 之前，如果需要，可以运行脚本来整理卸载过程留下的文件和数据。

卸载 版本 7.5 或 版本 7.1 后安装 版本 8.0

卸载 版本 7.5 或 版本 7.1 后，将使用相同的安装名称但使用 版本 8.0 缺省程序二进制文件位置 C:\Program Files\IBM\WebSphere MQ 来安装 版本 8.0 。 即，程序文件从 Windows 32 位程序位置移至 Windows 64 位程序位置。

您可以使用定制安装选项来修改二进制路径，包括将其修改回 C:\Program Files（x86）\WebSphere MQ。

安装 版本 8.0 而不卸载 版本 7.5 或 版本 7.1

如果在未卸载 版本 7.1 或 版本 7.5 的情况下安装 版本 8.0，并选择升级 版本 7.1 或 版本 7.5 安装，那么新的程序二进制文件将替换 版本 7.1 或 版本 7.5 二进制文件，因此缺省情况下，新的二进制文件位于 C:\Program Files（x86）\WebSphere MQ 中。 将保留现有数据路径，因此缺省情况下，数据路径也是 C:\Program Files（x86）\WebSphere MQ。

安装 版本 8.0 以与 版本 7.5 或 版本 7.1 共存

如果随 版本 7.1 或 版本 7.5 一起安装 版本 8.0，那么将选择唯一路径，缺省情况下为 C:\Program Files\IBM\WebSphere MQ。 保留现有数据路径，因此缺省情况下，数据路径为 C:\Program Files（x86）\WebSphere MQ。

现有 IBM WebSphere MQ 7.0.1 安装

缺省情况下， 版本 7.0.1 数据和程序二进制文件安装到 C:\Program Files（x86）\WebSphere MQ 中。

有三个可能的升级路径:

- 卸载 版本 7.0.1 并安装 版本 8.0。
- 安装 版本 8.0 并要求其升级 版本 7.0.1。
- 与 版本 7.0.1.6 + 一起安装 版本 8.0，然后卸载 版本 7.0.1。

卸载 版本 7.0.1 后安装 版本 8.0

如果在卸载 版本 7.0.1 之后安装 版本 8.0，那么安装程序将在 C:\ProgramData\IBM\MQ 目录中查找不存在的任何现有数据。 版本 8.0 程序文件安装到 C:\Program Files\IBM\WebSphere MQ 中，缺省情况下，程序数据使用 C:\ProgramData\IBM\MQ。

如果机器上有现有数据，那么必须选择定制选项，并手动将数据路径设置为 版本 7.0.1 先前用作其数据路径的位置，以便选取任何队列管理器。

安装 版本 8.0 以升级 版本 7.0.1

如果在未卸载 版本 7.0.1 的情况下安装 版本 8.0 并选择升级 版本 7.0.1 安装，那么新的程序二进制文件将替换 版本 7.0.1 二进制文件，因此缺省情况下，二进制文件的位置在 C:\Program Files（x86）\WebSphere MQ 中。

将保留现有数据路径，因此缺省情况下，数据路径也是 C:\Program Files（x86）\WebSphere MQ。

安装 版本 8.0 以与 版本 7.0.1 共存

如果安装 版本 8.0 而不卸载 版本 7.0.1，并且选择与 版本 7.0.1 安装一起安装，那么新的程序二进制文件将放在新的缺省位置 C:\Program Files\IBM\WebSphere MQ 中。

保留现有数据路径，因此缺省情况下，数据路径为 C:\Program Files（x86）\WebSphere MQ。

相关信息

迁移方法概述

z/OS: 更改为可用 CipherSpecs

在 z/OS 上还有两个 CipherSpecs 可用。

z/OS 以下 CipherSpecs(先前仅在分布式平台上可用) 现在也在 z/OS 上可用:

- ECDHE_RSA_AES_128_GCM_SHA256
- ECDHE_RSA_AES_256_GCM_SHA384

此更改是在 APAR PI97499 下进行的。

有关 CipherSpecs 的平台可用性的更多信息，请参阅 启用 CipherSpecs。

z/OS: 更改为在错误消息插入中显示映射的 IP 地址

z/OS 上的 CHINIT 地址空间中的错误消息中显示的映射 IP 地址现在显示为不带映射前缀。

包含映射的 IP 地址的错误消息

在先前版本的 IBM MQ for z/OS 中，包含映射 IP 地址作为插入内容的错误消息如下所示:

```
CSQX599E cpf CSQXRESP Channel channel-name ended abnormally
connection ::ffff:9.1.2.3
```

在 IBM MQ for z/OS 版本 8.0 中，包含映射 IP 地址作为插入内容的错误消息如下所示:

```
CSQX599E cpf CSQXRESP Channel channel-name ended abnormally
connection 9.1.2.3
```

此外，使用 **CONNAME** 参数发出 STOP CHANNEL 命令时，请确保 **CONNAME** 参数中指定的值与 DISPLAY CHSTATUS 中所示完全相同。

相关信息

分布式排队消息 (CSQX ...)

z/OS: 更改为 MAKECLNT

MAKECLNT CSQUTIL 工具已稳定。 您应改为使用 **runmqsc -n**。

有关更多信息，请参阅 runmqsc。

z/OS: 对日志 RBA 和 URID 长度的更改

控制台消息中的日志 RBA 和恢复单元标识 (URID) 以及 IBM MQ 8.0 中的命令响应现在为 8 字节长，显示为 16 个字符的十六进制值。 无论是否使用 OPMODE 启用了 版本 8.0 新功能，以及是否已迁移队列管理器以使用 8 字节日志 RBA，都是如此。

在先前发行版中，日志 RBA 和 URID 的长度为 6 字节，显示为 12 个字符的十六进制值。

有关更多信息，请参阅第 350 页的『较大的日志相对字节地址』。

对 z/OS， 版本 8.0 的 IBM MQ Managed File Transfer 的更改

在 z/OS 上的 IBM WebSphere MQ File Transfer Edition 与 z/OS 上的 IBM MQ Managed File Transfer 版本 8.0 之间进行了以下更改。

配置布局和文件许可权

已对 UNIX 系统服务中 IBM WebSphere MQ File Transfer Edition 和 IBM MQ 版本 8.0 之间的配置布局进行了一些重大更改。 有关更多信息，请参阅 IBM WebSphere MQ File Transfer Edition 版本 7.0.4 与 IBM MQ 版本 7.5 或更高版本之间的更改。

此外，这些文件现在设置了 USS 文件许可权以限制访问权。 有关更多信息，请参阅 文件系统许可权。

将环境变量从 FTE_CONFIG 更改为 BFG_DATA

对于 IBM MQ Managed File Transfer for z/OS 版本 8.0，定义数据目录位置的环境变量已从 FTE_CONFIG 更改为 BFG_DATA。

但是，已迁移的配置仍支持 FTE_CONFIG。 要允许兼容，可以在同一位置中具有旧的 IBM WebSphere MQ File Transfer Edition 配置和新的 IBM MQ Managed File Transfer 配置。

z/OS: 通道启动程序 SMF 记录

通道启动程序 (CHINIT) 可以生成 SMF 统计信息记录和记帐记录以及有关任务和通道的信息。

CHINIT 可以生成具有以下类型信息的 SMF 统计信息记录和记帐记录:

- 任务: 分派器，适配器，域名服务器 (DNS) 和 SSL。 这些任务构成了所谓的 CHINIT 统计信息。
- 通道: 提供类似于 DIS CHSTATUS 命令提供的记帐信息。 这称为通道记帐。

有关更多信息，请参阅第 288 页的『通道启动程序的 SMF 记录』。

IBM i distributed在 z/OS 以外的平台上， IBM MQ 通过将 PCF 消息写入 SYSTEM.ADMIN.STATISTICS.QUEUE。 有关如何在除 z/OS 以外的平台上记录统计信息的更多信息，请参阅 通道统计信息消息数据 。

不再支持 z/OS: WLM/DNS

z/OS Communications Server 不再支持 WLM/DNS ，因此 IBM MQ for z/OS 版本 8.0 中不再使用队列管理器属性 DNSWLM 和 DNSGROUP。

z/OS Communications Server

从 z/OS V1R11 开始，在 z/OS Communications Server 中不再提供对 WLM/DNS 作为 IP 地址路由机制的支持。

不再使用队列管理器属性

以下队列管理器属性不再导致 INDISP (GROUP) TCP 侦听器尝试进行 WLM/DNS 注册。

DNSGROUP
在使用工作负载管理器 for Dynamic Domain Name Services Support (WLM/DNS) 时，处理队列共享组的入站传输的 TCP 侦听器所连接的组的名称。

如果此名称为空，那么将使用队列共享组名。

此属性的 PCF 参数标识为 MQCA_DNS_GROUP

DNSWLM
处理队列共享组的入站传输的 TCP 侦听器是否向 WLM/DNS 注册:

NO
侦听器不会向工作负载管理器注册。

这是队列管理器的初始缺省值，也是 IBM MQ for z/OS 版本 8.0 中支持的唯一值。

YES
侦听器将向工作负载管理器注册。

此值在 IBM MQ for z/OS 版本 8.0 中不受支持，如果已设置此值，那么将导致消息 CSQX093I 由 CHINIT 发出。

此属性的 PCF 参数标识为 MQIA_DNS_WLM

备用支持

综合系统分发器可用作通用 IP 地址路由的备用功能，而不是 WLM/DNS。 有关更多信息，请参阅 使用综合系统分发器建立 TCP 连接。

相关信息

使用综合系统分发器建立 TCP 连接

ALTER QMGR

z/OS: 对 DISPLAY 命令输出的更改

现在，在 DISPLAY 命令的输出中显示的具有空白或空值的字段将显示在方括号之间没有空格。

此更改与 DISPLAY 命令输出的所有空白或空值字段相关。

例如，在 IBM MQ for z/OS 的先前发行版中， `DISPLAY QMGR SSLKEYR` 的输出 (其中 **SSLKEYR** 字段包含空白或 NULL 值) 为 SSLKEYR( )。

从 IBM MQ for z/OS 版本 8.0 开始，输出为 SSLKEYR()。

IBM MQ 8.0.0 的新消息，已更改的消息和已除去的消息

已为 IBM MQ 8.0.0 添加的新消息的摘要，以及指向更多信息的链接。 还会列出已更改或移除的任何消息

版本 8.0.0 的新增和已更改的 z/OS 消息

已添加大量新消息，并且已更改 IBM MQ for z/OS 版本 8.0.0 的某些现有消息。

- 第 45 页的『在 IBM MQ for z/OS 版本 8.0.0 中添加的消息』
- 第 46 页的『已更改 IBM MQ for z/OS CSQ 消息』

在 IBM MQ for z/OS 版本 8.0.0 中添加的消息

恢复日志管理器消息

CSQJ164I *csect-name* 日志归档已延迟，所有可用的卸载任务都在使用中

CSQJ168I *csect-name* 日志归档不再延迟

消息管理器消息

CSQM079I *csect-name* 由于 AMS 版本和作业名 *jobname* 不兼容而拒绝策略访问尝试

CSQM523I *csect-name* CLUSTER 或 CLROUTE 当前无法改变

CSQM526I *csect*-名称 当前无法改变集群或 CLroute*csect*-名称 SSL 3.0 通道不允许使用证书标签

缓冲区管理器消息

CSQP054I 缓冲池 *n* 现在位于条形上方

CSQP055I 缓冲池 *n* 现在位于条形下方

CSQP056E 缓冲池 *n* 的 ALTER BUFFPOOL 命令失败

主题管理器消息

CSQT824I *csect-name* Topic *topic-1* 依赖于来自不同发布/预订层次结构流的主题 *topic-2* 的 PROXYSUB (FORCE)

CSQT967E *csect-name* 无法将代理预订交付到队列管理器 *queue-manager*，原因 =*mqrc (mqrc-text)*

CSQT968I *csect-name* 集群 *cluster-name* 中的主题 *topic-1* 依赖于主题 *topic-2* 的 *PROXYSUB (FORCE)*

CSQT971E *csect-name* 任务 未能停顿

CSQT983E *csect-name* 任务 失败，原因为 *mqrc (mqrc-text)*，请在 *n* 分钟后重试

CSQT984E *csect-name* 任务 在尝试处理消息时迁到 *n* 次出现原因 *mqrc (mqrc-text)* 。

CSQT987E *csect-name* 任务 失败，原因是 *mqrc (mqrc-text)* 在 *n* 分钟内重试

CSQT988E *csect-name* 任务 失败，原因是 *mqrc (mqrc-text)* 在 *n* 分钟内重试

CSQT989E *csect-name* 任务 在尝试处理消息时迁到 *n* 次出现原因 *mqrc (mqrc-text)* 。

CSQT990E *csect-name* 任务 在尝试处理消息时迂到 *n* 次出现原因 *mqrc (mqrc-text)* 。

CSQT991I *csect-name* 任务 已从先前错误状态恢复

CSQT996E *csect-name* 在队列管理器 *qmgr-name*，集群 *cluster-name*，主题字符串 *topic-string* 上创建代理预订失败，原因 *=mqrc (mqrc-text)*

CSQT997E *csect-name* 在队列管理器 *qmgr-name*，集群 *cluster-name*，主题字符串 *topic-string* 上取消代理预订失败，原因 *=mqrc (mqrc-text)*

CSQT998E *csect-name* Proxy 预订再同步在队列管理器 *qmgr-name*，集群 *cluster-name* 上失败，原因 *=mqrc (mqrc-text)*

CSQT999E *csect-name* 代理预订在队列管理器 *qmgr-name* 上重新同步失败，集群 *cluster-name*，原因 *=mqrc (mqrc-text)csect-name* 任务在队列 队列 上迂到无效的消息

实用程序消息

CSQU179E 无法切换传输队列，因为通道启动程序未处于活动状态

分布式排队消息

CSQX469E *csect-name* 未收到集群 *cluster-name* 中队列管理器 *qmid* 上托管的 CLUSRCVR 通道 *channel-name* 的更新，预期 *n* 天前，剩余 *m* 天

CSQX878I *csect-name* 存储库命令错误，命令 命令，集群对象 *object-name*，发送方 *sender-id*，原因 *reason*

CSQX879E *csect-name* 冲突集群主题 *topic-name* 来自队列管理器 *qmgr-name*

初始化过程和常规服务消息

CSQY024I IBM MQ AMS for z/OS 未安装，但系统参数 SPLCAP 设置为 YES

CSQY025I IBM MQ AMS for z/OS 未安装，但系统参数 SPLCAP 设置为 YESIBM MQ AMS for z/OS 已安装。

CSQY336E *csect-name* 关键字 不允许-受限功能

CSQY337E *csect-name keyword* 值长度不允许-受限功能

IBM MQ Advanced Message Security 消息

CSQ0417I 保护质量: *qop*

CSQ0418I 容许: 容许-标志

CSQ0468I 找不到策略

服务设施消息

CSQ1134E 关键字抽取至少需要一个输出 DDNAME

CSQ1219I 日志记录包含 *n* BYTE RBA-QSG (*in-qsg*)

已更改 IBM MQ for z/OS CSQ 消息

针对 版本 8.0.0 更改了以下 CSQ 消息:

分布式排队消息 (CSQX ...)

CSQX776E 严重性已从 “参考” (I) 更改为 “错误” (E)

CSQX777E 严重性已从 “参考” (I) 更改为 “错误” (E)

CSQX782E 严重性已从 “参考” (I) 更改为 “错误” (E)

相关概念

第 74 页的『自 版本 8.0.0.0 以来新增，更改和除去了 z/OS 条消息』
自 版本 8.0.0.0 以来，已为 IBM MQ for z/OS 添加，更改或除去了大量消息。

相关信息

IBM MQ for z/OS 的消息

IBM MQ 8.0 修订包中新增和更改的内容

本节描述了 版本 8.0 修订包中的新增功能和已更改的功能。

Version 8.0.0, Fix Pack 16

V 8.0.0.16

没有专门与 Version 8.0.0, Fix Pack 16 相关的文档更新。

V 8.0.0FP15

V 8.0.0.15

- 第 50 页的『队列文件大小限制』
- 第 50 页的『更改为 IBM MQ classes for Java 和 IBM MQ classes for JMS 的缺省跟踪名称』
- Windows Linux UNIX 第 51 页的『更新为 IBM MQ AMQP 服务』

V 8.0.0FP14

V 8.0.0.14

- 第 51 页的『启用 IBM MQ 随附的 LDAP 客户机库代码的动态跟踪』
- 第 51 页的『更改 XMS .NET 查询集群队列的 BackoutThreshold 和 BackoutRequeueQName 所需的权限』
- z/OS 第 51 页的『更改 dmpmqcfg 命令使用的查询』
- 第 51 页的『新建环境变量 AMQ_NO_BAD_COMMS_DATA_FDCS』

Version 8.0.0, Fix Pack 13

V 8.0.0.13

没有与 IBM MQ 8.0.0FP13 中的功能和资源更改相关的文档更新。

V 8.0.0FP12

V 8.0.0.12

- 第 52 页的『更改 IBM MQ classes for JMS 查询集群队列的 BackoutThreshold 和 BackoutRequeueQName 所需的权限』
- distributed 第 52 页的『新建环境变量 AMQ_CONVEBCDICNEWLINE』
- AIX 第 52 页的『更改 IBM MQ 样本代码中用于构建 Sybase XA 切换文件的链接机制』
- Windows 第 52 页的『对 Windows 程序目录安全许可权的更改』

V 8.0.0FP11

V 8.0.0.11

- Windows 第 53 页的『在 Windows 上添加 Microsoft Visual Studio 2013 C/C++ 运行时』
- 第 53 页的『对 IBM MQ 客户机连接的行为更改』
- 第 53 页的『针对 IBM MQ classes for JMS 的回退队列和死信队列处理期间生成的消息的更新』

V 8.0.0FP10

V 8.0.0.10

- 第 54 页的『激活规范上 targetClientMatching 属性的资源适配器更新』
- 第 54 页的『runmqras 命令的新 -caseno 参数』

V 8.0.0 修订包 9

V 8.0.0.9

- 第 54 页的『除去 JSON4J.jar 文件和 com.ibm.msg.client.mqlight 包』
- 第 55 页的『java.security.policy 文件的其他许可权』
- 第 55 页的『允许在队列管理器上选择性地禁用 TLS v1.0 的新属性』
- 第 55 页的『runmqras 实用程序的增强功能』
- 第 55 页的『对 fteModifyAgent 或 fteModifyLogger 命令的更改』

V 8.0.0 修订包 8

V 8.0.0.8

- 第 56 页的『新建环境变量 AMQ_SSL_LDAP_SERVER_VERSION』
- 第 56 页的『对 dmpmqcfg 命令的更改』
- 第 56 页的『对 strmqm 命令的更改』
- 第 56 页的『runmqras 实用程序的增强功能』
- 第 56 页的『更改为处理与队列管理器集群组件所使用的内部 MQ 命令消息中的损坏字符数据相关的错误』
- 第 56 页的『对针对消息到文件传输恢复的 groupId 属性的传输标识和值进行比较检查』
- 第 56 页的『更改为 Managed File Transfer 代理权限队列所需的许可权』

V 8.0.0 修订包 7

V 8.0.0.7

- 第 57 页的『向 JmsConstants 接口添加了新的常量 JMS_IBM_SUBSCRIPTION_USER_DATA』
- Linux 第 57 页的『nproc 的 mqconfig 更改』
- 第 57 页的『IBM MQ classes for Java 的样本配置文件』
- 第 57 页的『更改 Managed File Transfer fteCleanAgent 命令的行为』
- 第 57 页的『在 Managed File Transfer 中启用 MQCSP 认证』
- 第 58 页的『Managed File Transfer installation.properties 文件的新 messagePublicationFormat 属性』
- 第 58 页的『IBM MQ Virtual System Pattern Type』

V 8.0.0 修订包 6

V 8.0.0.6

- 第 58 页的『JMS 异常侦听器更新』
- 第 59 页的『支持 JMS ObjectMessage 中的类名允许列表』
- Windows 第 59 页的『除去对使用 .NET for MQCNO_CLIENT_BINDING 和 MQCNO_LOCAL_BINDING 的限制』
- 第 59 页的『限制在分发列表中使用主题别名队列』
- 第 59 页的『已更新 GSKit 版本』
- z/OS distributed 第 59 页的『不推荐使用的 CipherSpecs』
- z/OS distributed 第 60 页的『新增 MFT 代理属性 additionalWildcardSandboxChecking』
- z/OS 第 60 页的『用于 z/OS 上的 MFT 代理程序的新代理程序属性 adminGroup』
- 第 60 页的『对 fteMigrateAgent 命令的更改』

- Linux AIX 第 60 页的『IBM MQ Virtual System Pattern Type』

V 8.0.0 修订包 5

V 8.0.0.5

- z/OS distributed 第 61 页的『使用 ADOPTCTX(YES) 参数时采用其他安全上下文』
- Windows 第 61 页的『支持与 Oracle 12c 数据库的 JDBC 连接』
- distributed 第 61 页的『支持 IBM MQ Appliance 上的 IBM MQ Light API』

V 8.0.0 修订包 4

V 8.0.0.4

- z/OS distributed 第 62 页的『限制消息的到期时间』
- Windows Linux UNIX 支持 MQ Light API
- Windows Linux UNIX 第 62 页的『支持将 SSL 和 TLS 检查合并到产品中』
- Windows Linux 第 62 页的『可重新分发的客户机』
- distributed 第 62 页的『保护数据库认证详细信息』
- distributed 第 62 页的『权限配置事件』
- Windows 第 63 页的『以不同方式命名的 IBM MQ C++ 库』
- Windows 第 63 页的『受管 .NET 客户机应用程序的用户标识和密码』
- Linux 第 63 页的『对 Docker 的支持』
- Linux AIX 第 63 页的『IBM MQ Virtual System Pattern Type』
- z/OS 第 63 页的『IMS 中的 IBM MQ classes for JMS』
- z/OS distributed 第 63 页的『新的 MFT 代理属性 failTransferOnFirst 故障』

V 8.0.0 修订包 3

- Linux UNIX 第 64 页的『安全性 :PAM 认证』
- z/OS distributed 第 64 页的『不推荐使用的 CipherSpecs』
- z/OS distributed 第 64 页的『Managed File Transfer 的可维护性增强功能』
- Windows Linux AIX 第 65 页的『支持 MQTT V 3.1.1 Oasis 标准』
- Linux 第 65 页的『IBM MQ Virtual System Pattern Type』

V 8.0.0 修订包 2

V 8.0.0.2

- Linux 第 65 页的『Linux on POWER Systems - Little Endian 支持』
- distributed 第 65 页的『安全性 :LDAP 授权』
- z/OS distributed 第 66 页的『不推荐使用的 CipherSpecs』

- Windows 第 66 页的『IBM MQ 使用 Microsoft Visual Studio 2005 构建的 C++ 客户机库』
- Windows 第 66 页的『独立 IBM MQ.NET 客户机』
- Linux 第 66 页的『IBM MQ Virtual System Pattern Type』
- z/OS 第 67 页的『CICS OSGi JVM 服务器中的 IBM MQ classes for JMS』

V 8.0.0 修订包 1

V 8.0.0.1

没有与 IBM MQ 8.0.0 修订包 1 中的功能和资源更改相关的文档更新。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

第 45 页的『IBM MQ 8.0.0 的新消息，已更改的消息和已除去的消息』
已为 IBM MQ 8.0.0 添加的新消息的摘要，以及指向更多信息的链接。 还会列出已更改或移除的任何消息

第 77 页的『先前版本中新增和更改的内容』
指向有关在 IBM MQ 8.0 之前的产品版本中发生的新功能以及对功能和资源的更改 (包括稳定化，废弃和移除) 的信息的链接。

第 80 页的『产品文档中的发行版和平台图标』
IBM Documentation 中的单个信息集中提供了所有 IBM MQ 8.0 修订包和平台的产品文档。 如果特定于给定修订包或平台的信息，那么此信息由矩形图标指示。 适用于所有修订包和平台的信息保持未标记状态。

相关信息

IBM MQ 系统需求
IBM MQ、WebSphere MQ 和 MQSeries 产品自述文件 Web 页面
WebSphere MQ 的建议修订
IBM MQ 计划维护发布日期

V 8.0.0.15 V 8.0.0FP15 中新增及更改的内容

在此部分中描述了对 IBM MQ 8.0.0FP15 中功能和资源的更改。

- 第 50 页的『队列文件大小限制』
- 第 50 页的『更改为 IBM MQ classes for Java 和 IBM MQ classes for JMS 的缺省跟踪名称』
- Windows Linux UNIX 第 51 页的『更新为 IBM MQ AMQP 服务』

队列文件大小限制

从 V 8.0.0FP15 开始，缺省情况下，队列管理器将最大队列文件大小限制为 2 TB。

更改为 IBM MQ classes for Java 和 IBM MQ classes for JMS 的缺省跟踪名称

从 V 8.0.0FP15 开始，已更新 IBM MQ classes for Java， IBM MQ classes for JMS 和 IBM MQ 资源适配器，以便当线程使用单独的 Java ClassLoaders 运行时，缺省情况下，每个线程都会生成对其自己的 ClassLoader 特定跟踪文件的跟踪:

- IBM MQ classes for Java 和 IBM MQ classes for JMS 的缺省跟踪名称的格式为: `mqjava_%PID%.cl%u.trc`。
- 如果您正在使用可重定位的 JAR 文件 `com.ibm.mq.mqallclient.jar`，那么缺省跟踪名称的格式为: `mqjavaclient_%PID%.cl%u.trc`。

其中，*%PID%* 是正在跟踪的应用程序的进程标识，*%u* 是唯一编号，用于区分在不同 Java 类装入器下运行跟踪的线程之间的文件。

有关更多信息，请参阅 跟踪 IBM MQ classes for Java 应用程序 和 跟踪 IBM MQ classes for JMS 应用程序的子主题。

更新为 IBM MQ AMQP 服务

Windows Linux UNIX

从 V 8.0.0FP15 开始，已更新 IBM MQ AMQP 服务，以便在设置 **SSLPEER** 属性并且 **SSLCAUTH** 属性设置为 REQUIRED 时，使用 **SSLPEER** 属性检查客户机证书的 "专有名称"。

有关更多信息，请参阅 创建和使用 AMQP 通道。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

相关信息

IBM MQ 系统需求

IBM MQ、WebSphere MQ 和 MQSeries 产品自述文件 Web 页面

V 8.0.0.14 V 8.0.0FP14 中新增及更改的内容

在此部分中描述了对 IBM MQ 8.0.0FP14 中功能和资源的更改。

- 第 51 页的『启用 IBM MQ 随附的 LDAP 客户机库代码的动态跟踪』
- 第 51 页的『更改 XMS .NET 查询集群队列的 BackoutThreshold 和 BackoutRequeueQName 所需的权限』
- z/OS 第 51 页的『更改 dmpmqcfg 命令使用的查询』
- 第 51 页的『新建环境变量 AMQ_NO_BAD_COMMS_DATA_FDCS』

启用 IBM MQ 随附的 LDAP 客户机库代码的动态跟踪

在 V 8.0.0FP14 之前，无法在不停止或启动队列管理器的情况下打开和关闭 LDAP 客户机跟踪。

从 V 8.0.0FP14 开始，您可以使用 **strmqtrc** 命令打开 LDAP 客户机跟踪，并使用 **endmqtrc** 命令关闭 LDAP 客户机跟踪，而无需停止或启动队列管理器。 要启用此行为，还必须将环境变量 **AMQ_LDAP_TRACE** 设置为非空值。 有关更多信息，请参阅 启用 LDAP 客户机库代码的动态跟踪。

更改 XMS .NET 查询集群队列的 BackoutThreshold 和 BackoutRequeueQName 所需的权限

从 V 8.0.0FP14 开始，已更新 XMS .NET 客户机，因此仅需要查询访问权即可查询集群队列的 **BackoutThreshold** 和 **BackoutRequeueQName** 。 有关更多信息，请参阅 *XMS* 中的 *Poison* 消息中的 集群队列的打开选项 。

更改 dmpmqcfg 命令使用的查询

z/OS

从 V 8.0.0FP14 开始， **dmpmqcfg** 命令使用的查询在缺省情况下仅查询 QSGDISP (QMGR) 定义。 您可以使用环境变量 **AMQ_DMPMQCFG_QSGDISP_DEFAULT** 来查询其他定义。 有关更多信息，请参阅 dmpmqcfg。

新建环境变量 AMQ_NO_BAD_COMMS_DATA_FDCS

从 V 8.0.0FP14 开始，可以使用 **AMQ_NO_BAD_COMMS_DATA_FDCS** 环境变量来禁止在初始通信流上报告 AMQ9207E 错误消息时捕获 FFST。 有关更多信息，请参阅环境变量。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

相关信息

IBM MQ 系统需求

IBM MQ、WebSphere MQ 和 MQSeries 产品自述文件 Web 页面

V 8.0.0.12 V 8.0.0FP12 中新增及更改的内容

在此部分中描述了对 IBM MQ 8.0.0FP12 中功能和资源的更改。

- 第 52 页的『更改 IBM MQ classes for JMS 查询集群队列的 BackoutThreshold 和 BackoutRequeueQName 所需的权限』
- distributed 第 52 页的『新建环境变量 AMQ_CONVEBCDICNEWLINE』
- AIX 第 52 页的『更改 IBM MQ 样本代码中用于构建 Sybase XA 切换文件的链接机制』
- Windows 第 52 页的『对 Windows 程序目录安全许可权的更改』

更改 IBM MQ classes for JMS 查询集群队列的 BackoutThreshold 和 BackoutRequeueQName 所需的权限

从 V 8.0.0FP12 开始，已更新 IBM MQ classes for JMS，因此仅需要查询访问权即可查询集群队列的 **BackoutThreshold** 和 **BackoutRequeueQName**。 在 V 8.0.0FP12 之前，还需要浏览和获取访问权。 有关更多信息，请参阅 在 IBM MQ classes for JMS 中处理有害消息。

新建环境变量 AMQ_CONVEBCDICNEWLINE

distributed

在 V 8.0.0FP12 之前， **ConvEBCDICNewline** 属性 (可用于指定 IBM MQ 如何将 EBCDIC NL 字符转换为 ASCII 格式) 仅通过使用 `mqs.ini` 文件可用。 从 V 8.0.0FP12 开始，可以使用 **AMQ_CONVEBCDICNEWLINE** 环境变量而不是 **ConvEBCDICNewline** 节属性，例如，在无法使用 `mqs.ini` 文件的情况下，在客户机端提供 ConvEBCDICNewline 功能。 有关更多信息，请参阅 所有队列管理器 和 环境变量。

更改 IBM MQ 样本代码中用于构建 Sybase XA 切换文件的链接机制

AIX

从 V 8.0.0FP12 开始，提供的 Makefile 已修改为使用不同的链接选项 (**-bstatic -bdynamic**)，这将适应 Sybase 15 ESD#5 和更高版本中提供的 XA 库文件格式的更改。 有关更多信息，请参阅 Sybase 配置。

对 Windows 程序目录安全许可权的更改

Windows

现在， Windows 上的 IBM MQ 安装程序会将其他许可权限制设置为 MQ 安装目录的安全配置的一部分。 执行此操作的逻辑在安装，升级，修改和修订包安装时运行。

您可能会发现，由于安全性的提高，您无法像以前那样执行某些操作。 例如：

- MQ 管理员 (他不是 Administrators 组的成员) 无法再编辑或重新编译 `Tools` 子目录中的样本程序。 如果要执行此操作，请获取目录 (或您感兴趣的部分) 的副本，并更改构建脚本的副本以反映新位置。

但是，在正常使用时，您应该不知道更改，只是安装程序进行更改所需的额外时间很少。 在此期间，消息 **正在初始化安全性 ...** 将显示。 安装修订包文件或应用补丁时，将发生类似的短暂暂停。

更新安全性会将日志 (`amqidsec-<Installationname>.txt`) 写入机器上的 TEMP 目录。 如果在定制操作 "iwiLaunchAmqidsec" 中看到主安装失败，那么应参阅此文件。

注: 安装修订包文件时，必须在机器上具有名为 "mqm" 的安全组。 如果您已在机器上安装 IBM MQ，那么将会出现这种情况。 但是，如果您的机器上只有 IBM MQ 修订包 (或者您希望先安装修订包文件，然后再安装 IBM MQ)，那么将需要创建此本地组。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

相关信息

IBM MQ 系统需求

IBM MQ、WebSphere MQ 和 MQSeries 产品自述文件 Web 页面

V 8.0.0.11 V 8.0.0FP11 中新增及更改的内容

在此部分中描述了对 IBM MQ 8.0.0FP11 中功能和资源的更改。

- Windows 第 53 页的『在 Windows 上添加 Microsoft Visual Studio 2013 C/C++ 运行时』
- 第 53 页的『对 IBM MQ 客户机连接的行为更改』
- 第 53 页的『针对 IBM MQ classes for JMS 的回退队列和死信队列处理期间生成的消息的更新』

在 Windows 上添加 Microsoft Visual Studio 2013 C/C++ 运行时

Windows

从 IBM MQ 8.0.0FP11 开始， IBM MQ 的某些元素需要 Microsoft Visual Studio 2013 C/C++ (VS2013) 运行时。 这些运行时由 IBM MQ 8.0.0FP11 或更高版本的任何全新安装进行安装。

如果机器上不存在 VS2013 运行时，请在安装全新客户机或服务器时，将先前发行版升级到这些级别，或者应用修订包。 在大多数情况下，此过程不需要用户操作，但是，如果在应用服务器修订包时迁到安装故障，那么可能需要指定一个环境变量，以便您可以手动安装运行时。 有关更多信息，请参阅 在服务器修订包中安装 Microsoft Visual Studio 2013 C/C++ 运行时。

对 IBM MQ 客户机连接的行为更改

从 V 8.0.0FP11 开始，对客户机通道连接到队列管理器的方式进行了一些细微更改。 这些更改最具体是在安全出口与 CHLAUTH 映射规则的应用之间的交互中。 有关更多信息，请参阅技术说明 对 APAR IT25839 引入的 MQ 客户机连接的行为更改。

针对 IBM MQ classes for JMS 的回退队列和死信队列处理期间生成的消息的更新

从 IBM MQ 8.0.0FP11 开始，当 IBM MQ classes for JMS 尝试将消息重新排队到指定的回退队列或队列管理器的死信队列失败时，会将一条消息写入日志，指示消息的消息标识，无法将消息移动到的回退或死信队列的名称以及阻止移动的异常或错误。

- 将消息移动到回退队列失败会导致以下日志消息:

```
JMSWMQ1120 = 消息 "message ID" ，无法移至回退队列
"backout queue name" ，原因为 "exception or error" ，将
移动到死信队列或废弃。
```

- 未能将消息移动到死信队列会导致以下日志消息:

```
JMSWMQ0035 = 消息 "message ID" ，无法移至死信队列
"dead-letter queue name" ，因为异常 "exception or error" ，
它将返回到源队列。
```

- 如果在使用者上注册了一个异常侦听器，该异常侦听器尝试并未能将消息重新排队到死信队列以及正在写入的日志消息，那么会将包含异常消息文本的异常发送到此异常侦听器:

```
JMSWMQ1079: 无法将消息写入死信队列。
```

有关更多信息，请参阅 JMS 异常消息。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

相关信息

IBM MQ 系统需求

IBM MQ、WebSphere MQ 和 MQSeries 产品自述文件 Web 页面

V 8.0.0.10 V 8.0.0FP10 中新增及更改的内容

在此部分中描述了对 IBM MQ 8.0.0FP10 中功能和资源的更改。

- 第 54 页的『激活规范上 targetClientMatching 属性的资源适配器更新』
- 第 54 页的『runmqras 命令的新 -caseno 参数』

激活规范上 targetClientMatching 属性的资源适配器更新

已更新 IBM MQ JCA 资源适配器 (MQ-RA)，以便可以为激活规范配置 **targetClientMatching** 属性。 您可以配置此属性，以便在请求消息不包含 MQRFH2 头时，在应答消息中包含 MQRFH2 头。 这意味着，在发送消息时将包含应用程序在应答消息上定义的所有消息属性。 有关更多信息，请参阅 为激活规范配置 targetClient 匹配属性。

runmqras 命令的新 -caseno 参数

-caseno 参数等同于 **-pmrno**，但它允许指定有效的 Salesforce 个案号。 **-caseno** 或 **-pmrno** 都是可选参数，但不允许同时提供这两个参数。 有关更多信息，请参阅 runmqras (收集 IBM MQ 诊断信息)。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

相关信息

IBM MQ 系统需求

IBM MQ、WebSphere MQ 和 MQSeries 产品自述文件 Web 页面

V 8.0.0.9 V 8.0.0 修订包 9 中新增及更改的内容

在此部分中描述了对 IBM MQ 8.0.0 修订包 9 中功能和资源的更改。

- 第 54 页的『除去 JSON4J.jar 文件和 com.ibm.msg.client.mqlight 包』
- 第 55 页的『java.security.policy 文件的其他许可权』
- 第 55 页的『允许在队列管理器上选择性地禁用 TLS v1.0 的新属性』
- 第 55 页的『runmqras 实用程序的增强功能』
- 第 55 页的『对 fteModifyAgent 或 fteModifyLogger 命令的更改』

除去 JSON4J.jar 文件和 com.ibm.msg.client.mqlight 包

IBM MQ classes for Java 和 IBM MQ classes for JMS 不需要 `JSON4J.jar` 文件和 `com.ibm.msg.client.mqlight` 包，因此将从 V 8.0.0 修订包 9 进行以下更改:

- `JSON4J.jar` 文件将从 *V.R.M.F*`-WS-MQ-Install-Java-All.jar` 文件中除去，其中 *V.R.M.F* 是产品版本号，例如 `8.0.0.9`。
- 将从 `com.ibm.mq.allclient.jar` 文件的清单文件中的类路径语句中除去对 `JSON4J.jar` 文件的引用。
- `com.ibm.msg.client.mqlight` 包将不再包含在 `com.ibm.mq.allclient.jar` 文件内。

请参阅 单独安装 IBM MQ classes for JMS， 为 IBM MQ classes for JMS 安装的内容和 为 IBM MQ classes for Java 安装的内容。

java.security.policy 文件的其他许可权

从 V 8.0.0 修订包 9 开始，如果 Java 应用程序使用 Java Security Manager，那么必须将 RuntimePermission 添加到应用程序所使用的 java.security.policy 文件，否则将向应用程序抛出异常。 在通过到队列管理器的 TCP/IP 连接管理多复用对话的分配和关闭过程中，客户机需要此 RuntimePermission。

有关更多信息，请参阅 在 Java Security Manager 下运行 IBM MQ classes for Java 应用程序。

允许在队列管理器上选择性地禁用 TLS v1.0 的新属性

从 V 8.0.0 修订包 9 开始，在 qm.ini 文件中的 SSL 节下提供了新属性:

```
SSL:
   AllowTLSV1=NO
```

如果在启动队列管理器之前在 qm.ini 文件中设置了此属性，那么队列管理器不会接受使用 TLS v1.0 协议的入站连接。 同样，如果使用 AUTHINFO 对象配置了 LDAP 连接，那么如果为 AUTHINFO 对象启用了安全通信，那么仅使用 TLS 1.2 与 LDAP 服务器进行通信。

或者，可以为用于启动队列管理器，侦听器和通道进程的环境设置 **AMQ_TLS_V1_DISABLE** 环境变量。

如果设置了任一属性，并且不允许在网络层进行 TLS 1.0 连接尝试，那么队列管理器的命令服务器还会拒绝尝试定义或更改通道定义以使用 TLS 1.0 CipherSpec。

缺省队列管理器行为保持不变，因此如果未设置新属性或环境变量，那么将继续接受 TLS 1.0 连接。

runmqras 实用程序的增强功能

Solaris Linux AIX

从 V 8.0.0 修订包 9 开始，对 **runmqras** 实用程序进行了以下增强:

- Solaris Linux AIX 缺省情况下，将检索环境变量信息。
- distributed 缺省情况下，将检索队列管理器数据目录列表。
- Linux UNIX 将 leak 部分添加到 **runmqras** 命令以收集 IBM MQ 进程资源使用情况信息。

有关更多信息，请参阅 runmqras (收集 IBM MQ 诊断信息)。

对 fteModifyAgent 或 fteModifyLogger 命令的更改

从 V 8.0.0 修订包 9 开始，将在 APAR IT22423 下添加其他检查，以便验证使用 **fteModifyAgent** 或 **fteModifyLogger** 命令的 **-serviceJVMOptions** 参数对代理程序或记录器的 JVM 选项进行的任何更新，以确保正确指定了这些选项。 有关更多信息，请参阅 有关更新代理程序或记录器 JVM 选项的指南。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

相关信息

IBM MQ 系统需求

IBM MQ、WebSphere MQ 和 MQSeries 产品自述文件 Web 页面

V 8.0.0.8 V 8.0.0 修订包 8 中新增及更改的内容

在此部分中描述了对 IBM MQ 8.0.0 修订包 8 中功能和资源的更改。

- 第 56 页的『新建环境变量 AMQ_SSL_LDAP_SERVER_VERSION』
- 第 56 页的『对 dmpmqcfg 命令的更改』
- 第 56 页的『对 strmqm 命令的更改』
- 第 56 页的『runmqras 实用程序的增强功能』
- 第 56 页的『更改为处理与队列管理器集群组件所使用的内部 MQ 命令消息中的损坏字符数据相关的错误』
- 第 56 页的『对针对消息到文件传输恢复的 groupId 属性的传输标识和值进行比较检查』
- 第 56 页的『更改为 Managed File Transfer 代理权限队列所需的许可权』

新建环境变量 AMQ_SSL_LDAP_SERVER_VERSION

从 V 8.0.0 修订包 8 开始，可以设置环境变量 AMQ_SSL_LDAP_SERVER_VERSION，以确保在针对更高版本的 LDAP 证书撤销列表 (CRL) 服务器验证证书时，IBM MQ 将所需的轻量级目录访问协议 (LDAP) 版本 (即 LDAP v2 或 LDAP v3) 传递到 GSKit。有关更多信息，请参阅环境变量。

对 dmpmqcfg 命令的更改

从 V 8.0.0 修订包 8 开始，将命令行开关 **-w** 添加到 **dmpmqcfg**。此开关允许您增加 **dmpmqcfg** 等待来自命令服务器的应答消息的时间 (以秒计)。有关更多信息，请参阅 dmpmqcfg。

对 strmqm 命令的更改

从 V 8.0.0 修订包 8 开始，**strmqm** 命令在完全启动队列管理器之前检查 qm.ini 文件中的 CHANNELS 和 SSL 节的语法，这使您更容易查看错误内容，并在 **strmqm** 发现 qm.ini 文件包含任何错误时快速进行更正。有关更多信息，请参阅 strmqm。

runmqras 实用程序的增强功能

从 V 8.0.0 修订包 8 开始，将 mft 部分添加到 **runmqras** 命令以捕获由 **fteRas** 命令获取的数据。有关更多信息，请参阅 runmqras (收集 IBM MQ 诊断信息)。

更改为处理与队列管理器集群组件所使用的内部 MQ 命令消息中的损坏字符数据相关的错误

从 V 8.0.0 修订包 8 开始，将更改队列管理器处理与队列管理器的集群组件所使用的内部 MQ 命令消息中的损坏字符数据相关的错误的方式。可能包含不可转换字符的字段将被清空，并且将重试转换。如果转换成功，那么将使用空白描述来存储更新，并写入错误消息 AMQ6174 和 AMQ9880，以通知管理员。如果消息仍不可转换，那么会将其移至死信队列并写入错误消息。

对针对消息到文件传输恢复的 groupId 属性的传输标识和值进行比较检查

在 V 8.0.0 修订包 5，APAR IT18213 除去了比较传输标识的值以及传输请求 XML 有效内容中 **groupId** 属性的值的比较检查。从 V 8.0.0 修订包 8 开始，将更新 Managed File Transfer 以复原此比较检查。如果这两个标识等效，那么源代理现在将该标识用作消息标识匹配选项 (而不是组标识匹配选项)，以用于在消息到文件传输的输入队列上进行的第一次 **MQGET** 尝试。另请参阅 将数据从消息传输到文件。

更改为 Managed File Transfer 代理权限队列所需的许可权

从 V 8.0.0 修订包 8 开始，当通过设置代理属性 **authorityChecking=true** 来启用用户权限管理时，inquire 是对所有代理权限队列的必需许可权。

有关更多信息，请参阅 IBM MQ Managed File Transfer 操作的用户权限 和 MFT agent.properties 文件。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

相关信息

IBM MQ 系统需求

IBM MQ、WebSphere MQ 和 MQSeries 产品自述文件 Web 页面

V 8.0.0.7 V 8.0.0 修订包 7 中新增及更改的内容

在此部分中描述了对 IBM MQ 8.0.0 修订包 7 中功能和资源的更改。

- 第 57 页的『向 JmsConstants 接口添加了新的常量 JMS_IBM_SUBSCRIPTION_USER_DATA』
- Linux 第 57 页的『nproc 的 mqconfig 更改』
- 第 57 页的『IBM MQ classes for Java 的样本配置文件』
- 第 57 页的『更改 Managed File Transfer fteCleanAgent 命令的行为』
- 第 57 页的『在 Managed File Transfer 中启用 MQCSP 认证』
- 第 58 页的『Managed File Transfer installation.properties 文件的新 messagePublicationFormat 属性』
- Linux AIX 第 58 页的『IBM MQ Virtual System Pattern Type』

向 JmsConstants 接口添加了新的常量 JMS_IBM_SUBSCRIPTION_USER_DATA

从 V 8.0.0 修订包 7 开始，将更新 IBM MQ classes for JMS，以便在从包含带有 MQPS 文件夹的 RFH2 头的队列中使用消息时，会将与 Sud 键关联的值 (如果存在) 作为字符串属性添加到返回到 IBM MQ classes for JMS 应用程序的 JMS Message 对象。为了使应用程序能够从消息中检索此属性，会向 JmsConstants 接口添加新的常量 JMS_IBM_SUBSCRIPTION_USER_DATA。此新属性可与方法 `javax.jms.Message.getStringProperty(java.lang.String)` 配合使用，以检索预订用户数据。有关更多信息，请参阅 检索用户预订数据 和 DEFINE SUB。

nproc 的 mqconfig 更改

Linux

在 Linux 上，每个线程都实现为一个轻量级进程 (LWP)，并且每个 LWP 都被视为一个针对资源限制 nproc 的进程。因此，需要根据线程数设置 nproc。

从 V 8.0.0 修订包 7 开始，**mqconfig** 脚本已修改为根据正在运行的线程数而不是进程数来显示进程数。

有关 mqconfig 和 nproc 的更多信息，请参阅 在 Linux 和 mqconfig 上配置和调整操作系统。

IBM MQ classes for Java 的样本配置文件

从 V 8.0.0 修订包 7 开始，在 IBM MQ classes for Java 安装目录的 `bin` 子目录中提供了样本配置文件 `mqjava.config`。有关更多信息，请参阅 IBM MQ classes for Java 配置文件。

更改 Managed File Transfer `fteCleanAgent` 命令的行为

从 V 8.0.0 修订包 7 开始，**fteCleanAgent** 命令已更新，因此必须通过向命令传递相应参数以及提供代理程序名称来指定要清除的 Managed File Transfer 状态。此行为更改可确保缺省情况下，**fteCleanAgent** 不会清除指定代理的所有进行中和暂挂的传输，资源监视器定义和调度的传输定义。

如果需要，您可以通过将 `command.properties` 文件中新的 `failCleanAgentWithNoArguments` 属性设置为 false 来还原到 **fteCleanAgent** 的先前行为。

有关更多信息，请参阅 fteCleanAgent (清除受管文件传输代理) 和 command.properties 文件。

在 Managed File Transfer 中启用 MQCSP 认证

从 V 8.0.0 修订包 7 开始，您可以禁用缺省兼容性方式，并为以下对象的连接认证启用 MQCSP 认证:

- 连接到协调队列管理器或命令队列管理器的 Managed File Transfer 代理程序。

- 连接到协调队列管理器或命令队列管理器的 MQ Explorer MFT 插件。

有关更多信息，请参阅启用 MQCSP 认证方式。

Managed File Transfer `installation.properties` 文件的新 `messagePublicationFormat` 属性

从 V 8.0.0 修订包 7 开始，会将新属性 **messagePublicationFormat** 添加到 `installation.properties` 文件。 此属性允许您指定 MFT 代理程序用于其状态 XML 消息的消息发布格式。 有关更多信息，请参阅 installation.properties 文件。

IBM MQ Virtual System Pattern Type

Linux AIX

V 8.0.0 修订包 7 随 IBM MQ Virtual System Pattern Type V 1.0.0.7 提供。

有关更多信息，请参阅 IBM MQ 虚拟系统模式类型概述。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

相关信息

IBM MQ 系统需求

IBM MQ、WebSphere MQ 和 MQSeries 产品自述文件 Web 页面

V 8.0.0.6 V 8.0.0 修订包 6 中新增及更改的内容

在此部分中描述了对 IBM MQ 8.0.0 修订包 6 中功能和资源的更改。

- 第 58 页的『JMS 异常侦听器更新』
- 第 59 页的『支持 JMS ObjectMessage 中的类名允许列表』
- Windows 第 59 页的『除去对使用 .NET for MQCNO_CLIENT_BINDING 和 MQCNO_LOCAL_BINDING 的限制』
- 第 59 页的『限制在分发列表中使用主题别名队列』
- 第 59 页的『已更新 GSKit 版本』
- z/OS distributed 第 59 页的『不推荐使用的 CipherSpecs』
- z/OS distributed 第 60 页的『新增 MFT 代理属性 additionalWildcardSandboxChecking』
- z/OS 第 60 页的『用于 z/OS 上的 MFT 代理程序的新代理程序属性 adminGroup』
- 第 60 页的『对 fteMigrateAgent 命令的更改』
- Linux AIX 第 60 页的『IBM MQ Virtual System Pattern Type』

JMS 异常侦听器更新

从 V 8.0.0 修订包 6 开始，将更新 IBM MQ classes for JMS ， 以便:

- 针对任何连接中断异常调用由应用程序注册的 ExceptionListener，无论应用程序使用的是同步消息使用者还是异步消息使用者。
- 如果 JMS 会话使用的 TCP/IP 套接字中断，将调用由应用程序注册的 ExceptionListener。
- 当应用程序使用异步消息使用者，并且应用程序使用的 JMS ConnectionFactory 将 ASYNC_EXCEPTIONS 属性设置为 ASYNC_EXCEPTIONS_ALL 值时，在消息传递期间发生的非连接中断异常（例如，MQRC_GET_INHIBITED）将传至该应用程序的 ExceptionListener。

注: 仅为连接中断异常调用一次 ExceptionListener，即使两个 TCP/IP 连接（一个由 JMS 连接使用，一个由 JMS 会话使用）均已中断。

有关更多信息，请参阅 IBM MQ classes for JMS 中的异常。

支持 JMS ObjectMessage 中的类名允许列表

从 V 8.0.0 修订包 6 开始， IBM MQ classes for JMS 支持 JMS ObjectMessage 接口实现中类的允许列表。 允许列表定义哪些 Java 类可以使用 ObjectMessage.setObject() 序列化以及哪些可以使用 ObjectMessage.getObject() 取消序列化。

有关更多信息，请参阅 JMS ObjectMessage 中的类名允许列表 和 在 Java Security Manager 下针对 JMS 应用程序运行 IBM MQ 类。

除去对使用 .NET for MQCNO_CLIENT_BINDING 和 MQCNO_LOCAL_BINDING 的限制

Windows

从 V 8.0.0 修订包 6 开始，已更新 Microsoft Windows Communication Foundation (WCF) 的 IBM MQ 定制通道，以便在从仅客户机安装运行时使用正确的客户机连接配置。 有关更多信息，请参阅 使用 MQCONNX 调用连接到队列管理器。

限制在分发列表中使用主题别名队列

分发列表不支持使用指向主题对象的别名队列。 从 V 8.0.0 修订包 6 开始，如果别名队列指向分发列表中的主题对象，那么 IBM MQ 将返回 MQRC_ALIAS_BASE_Q_TYPE_ERROR。

已更新 GSKit 版本

已在 V 8.0.0 修订包 6 中更新 GSKit 版本。 新版本的 GSKit 会改变生成 `.sth` 文件以隐藏密钥数据库密码时使用的隐藏文件格式。 先前版本的 GSKit 无法读取使用此版本的 GSKit 生成的隐藏文件。

要确保使用 V 8.0.0 修订包 6 或更高版本生成的隐藏文件与应用程序和其他 IBM MQ 安装兼容，必须更新为包含兼容版本的 GSKit 的 IBM MQ 版本。 以下修订包包含兼容版本的 GSKit:

- v7.1.0.8
- v7.5.0.8
- v8.0.0.6
- v9.0.0.1

如果无法更新应用程序或其他 IBM MQ 安装，那么可以请求与较早版本兼容的隐藏文件格式。 将 **runmqakm** 或 **runmqckm** 命令与 `-stash` 或 `-stashpw` 选项配合使用时，请包含 `-v1stash` 命令行参数。 不能使用 iKeyman GUI 来生成与较早版本兼容的隐藏文件。

不推荐使用的 CipherSpecs

z/OS distributed

从 V 8.0.0 修订包 6 开始，不推荐使用以下 CipherSpecs :

- Windows Linux UNIX FIPS_WITH_3DES_EDE_CBC_SHA
- TRIPLE_DES_SHA_US
- TLS_RSA_WITH_3DES_EDE_CBC_SHA
- Windows Linux UNIX ECDHE_ECDSA_3DES_EDE_CBC_SHA256
- Windows Linux UNIX ECDHE_RSA_3DES_EDE_CBC_SHA256

z/OS 要重新启用 z/OS 上已不推荐使用的任何 CipherSpec，如果需要，请参阅相应的 SEC APAR 中的 + + HOLD 信息。

有关更多信息，请参阅不推荐使用的 CipherSpec。

新增 MFT 代理属性 additionalWildcardSandboxChecking

z/OS distributed

从 V 8.0.0 修订包 6 开始，如果已使用用户或代理沙箱配置代理，以限制代理可以将文件传输到的位置，那么可以通过将 additionalWildcardSandboxChecking 属性设置为 true 来指定要对该代理的通配符传输进行其他检查。 有关更多信息，请参阅 通配符传输的其他检查 和 agent.properties 文件。

用于 z/OS 上的 MFT 代理程序的新代理程序属性 adminGroup

z/OS

从 V 8.0.0 修订包 6 开始，将添加新的代理程序属性 **adminGroup** 以用于 z/OS 上的 Managed File Transfer 代理程序。 此属性定义可执行以下操作的用户组的名称:

- 使用 **fteStartAgent** 命令启动代理。
- 使用 **fteStopAgent** 命令停止代理。
- 使用 **fteSetAgentTraceLevel** 命令启用或禁用代理的跟踪。
- 使用 **fteShowAgentDetails** 命令显示代理详细信息。

有关更多信息，请参阅 agent.properties 文件中的 **adminGroup** 属性。

对 fteMigrateAgent 命令的更改

从 V 8.0.0 修订包 6 开始，已更新 **fteMigrateAgent** 命令以确保跟踪用于查看用户是否为管理员的检查。

z/OS 还更新了该命令，以检查用户是否满足 (至少) 下列其中一个条件，以便在 z/OS 上运行迁移命令:

- 成为 mqm 组的成员 (如果 mqm 组存在)。
- 是 BFG_GROUP_NAME 环境变量中指定的组（如果指定了一个组）的成员。
- 在 BFG_GROUP_NAME 环境变量中未设置任何值。

有关 **fteMigrateAgent** 命令的更多信息，请参阅 fteMigrate 代理 。

IBM MQ Virtual System Pattern Type

Linux AIX

从 V 8.0.0 修订包 6 开始， IBM MQ Virtual System Pattern Type 包含 AMQP 服务组件，该组件提供对 MQ Light API 的支持。 有关更多信息，请参阅 创建和使用 AMQP 通道。

支持在模式中可选安装 IBM MQ Advanced 组件，手动管理 IBM MQ 数据目录以防止在删除实例时删除数据，以及在随后部署的模式实例中复用现有 IBM MQ 队列管理器数据。 有关更多信息，请参阅 IBM MQ Virtual System Pattern Type 软件组件的配置参数。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

相关信息

IBM MQ 系统需求

IBM MQ、WebSphere MQ 和 MQSeries 产品自述文件 Web 页面

V 8.0.0.5 V 8.0.0 修订包 5 中新增及更改的内容

在此部分中描述了对 IBM MQ 8.0.0 修订包 5 中功能和资源的更改。

- z/OS distributed 第 61 页的『使用 ADOPTCTX(YES) 参数时采用其他安全上下文』
- 第 61 页的『支持与 Oracle 12c 数据库的 JDBC 连接』
- distributed 支持 IBM MQ Appliance 上的 IBM MQ Light API

使用 ADOPTCTX(YES) 参数时采用其他安全上下文

z/OS distributed

从 V 8.0.0 修订包 5 开始，在认证信息对象上使用 **ADOPTCTX(YES)** 参数时，除非在 `qm.ini` 文件的通道节中设置 **ChlauthEarlyAdopt** 参数，否则无法采用其他安全上下文。

有关更多信息，请参阅 通道节的属性。

支持与 Oracle 12c 数据库的 JDBC 连接

从 V 8.0.0 修订包 5 开始，新文件 `jdbcora12.dll` 随 IBM MQ Windows 服务器安装映像一起提供，以支持与 Oracle 12c 数据库的 JDBC 连接。 请参阅 在 Windows 上配置 JTA/JDBC 协调。

支持 IBM MQ Appliance 上的 IBM MQ Light API

distributed

从 V 8.0.0 修订包 5 开始，可以在 IBM MQ Appliance 上使用 AMQP 服务组件。 AMQP 服务组件提供对 IBM MQ Light API 的支持。

有关如何将 MQ Light 与 IBM MQ 配合使用的信息，请参阅 IBM MQ Light API 支持简介。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

相关信息

IBM MQ 系统需求

IBM MQ、WebSphere MQ 和 MQSeries 产品自述文件 Web 页面

V 8.0.0.4 V 8.0.0 修订包 4 中新增及更改的内容

在此部分中描述了对 IBM MQ 8.0.0 修订包 4 中功能和资源的更改。

- z/OS distributed 第 62 页的『限制消息的到期时间』
- Windows Linux UNIX 第 62 页的『对 IBM MQ Light API 的支持』
- Windows Linux UNIX 第 62 页的『支持将 SSL 和 TLS 检查合并到产品中』
- Windows Linux 第 62 页的『可重新分发的客户机』
- distributed 第 62 页的『保护数据库认证详细信息』
- distributed 第 62 页的『权限配置事件』
- Windows 第 63 页的『以不同方式命名的 IBM MQ C++ 库』
- Windows 第 63 页的『受管 .NET 客户机应用程序的用户标识和密码』
- Linux 第 63 页的『对 Docker 的支持』
- Linux AIX 第 63 页的『IBM MQ Virtual System Pattern Type』
- z/OS 第 63 页的『IMS 中的 IBM MQ classes for JMS』

- z/OS distributed 第 63 页的『新的 MFT 代理属性 failTransferOnFirst 故障』

限制消息的到期时间

z/OS distributed

从 V 8.0.0 修订包 4 开始，管理员可以使用队列或主题上的 **CUSTOM** 属性中指定的 *CAPEXPRY* 属性来限制放入队列或主题的任何消息的到期时间。

z/OS 要在 z/OS 上启用此功能，必须应用 APAR PI50761 并使用 MQSC 命令 **RECOVER QMGR (TUNE CAPEXPRY ON)** 启用新功能。

要点: RECOVER QMGR 命令保留供 IBM 使用，通常仅应在 IBM 支持人员的指导下使用。

z/OS 您可以在控制台输入命令，也可以通过在队列管理器启动期间添加到 CSQINP2 并置中处理的文件来输入队列管理器配置。 请注意，该命令的效果不会在队列管理器重新启动时持久存在。

有关消息到期处理的更多信息，请参阅 强制缩短到期时间。

对 IBM MQ Light API 的支持

Windows Linux UNIX

IBM MQ Light 是一种产品和消息传递 API，旨在使开发人员能够更快，更轻松地使其应用程序更具响应性和可伸缩性。 IBM MQ 通过使用新通道类型 AMQP 来支持 MQ Light API。 AMQP 通道在启动时定义一个端口号，该端口号接受来自 MQ Light 应用程序的连接。

使用 IBM MQ V8.0.0.4 制造更新 (而不是 V8.0.0.4 修订包) 来安装 AMQP 服务组件。 不能在低于 V8.0.0.4 的队列管理器版本上安装 AMQP 组件。

有关概述，请参阅 IBM MQ 支持 IBM MQ Light API 简介，有关如何将 IBM MQ 与 MQ Light 配合使用的示例，请参阅 MQ Light 方案。

支持将 SSL 和 TLS 检查合并到产品中

Windows Linux UNIX

V 8.0.0 修订包 4 包含最初通过 SupportPac MH03 通过 **mqcertck** 命令执行的 SSL 检查工具。 请参阅 mqcertck 以获取更多信息。

可重新分发的客户机

Windows Linux

V 8.0.0 修订包 4 包含 Linux x86-64 和 Windows 64 位平台上的可再分发客户机映像。 请参阅 可再分发的客户机 以获取更多信息。

保护数据库认证详细信息

distributed

从 V 8.0.0 修订包 4 开始，当 XAOpenString 包含密码时，您可以获取 IBM MQ 以保护此信息，而不是使密码在 `qm.ini` 文件中以纯文本显示。 IBM MQ 将用户名和密码 (以加密形式) 存储在另一个文件中，并使用这些凭证连接到数据库。 有关详细信息，请参阅 保护数据库认证详细信息。

权限配置事件

distributed

V 8.0.0 修订包 4 提供了允许您捕获权限记录更改的审计跟踪的事件消息，以及使用 REFRESH QMGR 命令上的新属性 (AUTHREC) 创建初始基本线图片的方法。

三个权限事件消息为:

- 更改权限记录
- 删除权限记录
- 刷新权限记录

以不同方式命名的 IBM MQ C++ 库

Windows

从 V 8.0.0 修订包 4 开始，IBM MQ 另外提供了一些名称不同的 C++ 客户机库。 这些库是使用 Microsoft Visual Studio 2012 C++ 编译器构建的。 除了使用 Microsoft Visual Studio 2012 C++ 编译器构建的现有 IBM MQ 8.0 C++ 库外，还提供了这些库。 由于这些额外的 IBM MQ C++ 库具有不同的名称，因此您可以在同一机器上运行使用 Microsoft Visual Studio 2012 编译的 IBM MQ C++ 应用程序和使用 Microsoft Visual Studio 2005 编译的应用程序。 有关更多信息，请参阅 在 Windows 上构建 C++ 程序。

受管 .NET 客户机应用程序的用户标识和密码

Windows

从 V 8.0.0 修订包 4 开始，通过受管 .NET 客户机应用程序指定的用户标识和密码在传递到客户机安全性出口的 IBM MQ .NET MQChannelDefinition 类中设置。 有关更多信息，请参阅 在 IBM WebSphere MQ .NET 中使用通道出口。

对 Docker 的支持

Linux

V 8.0.0 修订包 4 引入了对 Linux x86-64 平台上的 Docker 的支持。

请参阅 使用 Docker 配置 IBM MQ ，以获取有关如何将 Docker 与 IBM MQ 配合使用的更多信息。

IBM MQ Virtual System Pattern Type

Linux AIX

从 V 8.0.0 修订包 4 开始， IBM MQ Virtual System Pattern Type 可用于 Power AIX 以及 Linux。 在 Power AIX 上，它在 IBM PureApplication System W2700V 2.1.2 上运行。 有关更多信息，请参阅 IBM MQ 虚拟系统模式类型概述。

IMS 中的 IBM MQ classes for JMS

z/OS

V 8.0.0 修订包 4 添加了在 IMS 中使用 IBM MQ classes for JMS 的支持。

有关更多信息，请参阅 在 IMS 中使用 IBM MQ classes for JMS 。

新的 MFT 代理属性 failTransferOnFirst 故障

z/OS distributed

从 IBM MQ 8.0.0 修订包 4 开始，您可以通过在 `agent.properties` 文件中设置 failTransferOnFirstFailure 属性，将代理配置为在该受管传输中的传输项失败时立即使该受管传输失败。 有关更多信息，请参阅 agent.properties 文件。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

相关信息

IBM MQ 系统需求
IBM MQ、WebSphere MQ 和 MQSeries 产品自述文件 Web 页面

V 8.0.0.3 V 8.0.0 修订包 3 中新增及更改的内容

在此部分中描述了对 IBM MQ 8.0.0 修订包 3 中功能和资源的更改。

- Linux UNIX 第 64 页的『安全性 :PAM 认证』
- z/OS distributed 第 64 页的『不推荐使用的 CipherSpecs』
- 第 64 页的『Managed File Transfer 的可维护性增强功能』
- Windows Linux AIX 第 65 页的『支持 MQTT V 3.1.1 Oasis 标准』
- Linux 第 65 页的『IBM MQ Virtual System Pattern Type』

安全性 :PAM 认证

Linux UNIX

从 V 8.0.0 修订包 3 开始，可以在 UNIX 和 Linux 平台上使用可插拔认证方法 (PAM)。

有关更多信息，请参阅 使用可插入认证方法。

不推荐使用的 CipherSpecs

z/OS distributed

从 V 8.0.0 修订包 3 开始，不推荐使用以下 CipherSpecs :

- RC2_MD5_EXPORT
- RC4_MD5_EXPORT
- RC4_MD5_US
- RC4_SHA_US
- Windows Linux UNIX RC4_56_SHA_EXPORT1024
- IBM i TLS_RSA_EXPORT_WITH_RC2_40_MD5
- IBM i TLS_RSA_EXPORT_WITH_RC4_40_MD5
- IBM i TLS_RSA_WITH_RC4_128_MD5
- Windows Linux UNIX ECDHE_ECDSA_RC4_128_SHA256
- Windows Linux UNIX ECDHE_RSA_RC4_128_SHA256
- TLS_RSA_WITH_NULL_SHA256
- Windows Linux UNIX TLS_RSA_WITH_RC4_128_SHA256

有关更多信息，请参阅不推荐使用的 CipherSpec。

Managed File Transfer 的可维护性增强功能

从 V 8.0.0 修订包 3 开始，已对 Managed File Transfer 进行了以下更改:

- `installation.properties` 文件中 commandMessagePriority 属性的缺省值已更改为 8。 有关更多信息，请参阅 installation.properties 文件。
- `agent.properties` 文件中 logTransferRecovery 属性的缺省值已更改为 true。 有关更多信息，请参阅 agent.properties 文件。
- 如果代理程序迁到不可恢复错误，那么将生成首次故障数据捕获 (FDC)。

支持 MQTT V 3.1.1 Oasis 标准

Windows Linux AIX

从 V 8.0.0 修订包 3 可以设置 MQTT 通道以使用 Oasis MQTT V 3.1.1 客户机协议。 有关更多信息，请参阅 IBM MQ Telemetry Transport 格式和协议。

IBM MQ Virtual System Pattern Type

Linux

从 V 8.0.0 修订包 3 开始，您可以使用模式中的不同虚拟机来添加 IBM MQ Multi-Instance 支持。 有关更多信息，请参阅 添加 IBM MQ Multi-Instance 支持。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

相关信息

IBM MQ 系统需求

IBM MQ、WebSphere MQ 和 MQSeries 产品自述文件 Web 页面

V 8.0.0.2 V 8.0.0 修订包 2 中新增及更改的内容

在此部分中描述了对 IBM MQ 8.0.0 修订包 2 中功能和资源的更改。

- Linux 第 65 页的『Linux on POWER Systems - Little Endian 支持』
- IBM i UNIX 第 65 页的『安全性 :LDAP 授权』
- z/OS distributed 第 66 页的『不推荐使用的 CipherSpecs』
- Windows 第 66 页的『IBM MQ 使用 Microsoft Visual Studio 2005 构建的 C++ 客户机库』
- Windows 第 66 页的『独立 IBM MQ.NET 客户机』
- Linux 第 66 页的『IBM MQ Virtual System Pattern Type』
- z/OS 第 67 页的『CICS OSGi JVM 服务器中的 IBM MQ classes for JMS』
- V 8.0.0.2 第 67 页的『JMS 异常侦听器更改』

Linux on POWER Systems - Little Endian 支持

Linux 从 V 8.0.0 修订包 2 开始， IBM MQ 添加了对 Linux on POWER Systems - Little Endian 平台的支持，该平台仅运行 64 位应用程序。 有关更多信息，请参阅 在 Linux Ubuntu 或 Linux on POWER Systems - Little Endian 上安装 IBM MQ 服务器。

安全性 :LDAP 授权

IBM i UNIX

从 V 8.0.0 修订包 2 开始，您可以使用 LDAP 存储库进行授权，而无需本地操作系统标识。

LDAP 授权在以下平台上可用:

- UNIX 平台
- IBM i

有关更多信息，请参阅 LDAP 授权概述。

不推荐使用的 CipherSpecs

z/OS distributed

从 V 8.0.0 修订包 2 开始，不推荐使用以下 CipherSpecs：

- IBM i AES_SHA_US
- DES_SHA_EXPORT
- Windows Linux UNIX DES_SHA_EXPORT1024
- Windows Linux UNIX FIPS_WITH_DES_CBC_SHA
- NULL_MD5
- NULL_SHA
- TLS_RSA_WITH_DES_CBC_SHA
- IBM i TLS_RSA_WITH_NULL_MD5
- IBM i TLS_RSA_WITH_NULL_SHA
- Windows Linux UNIX ECDHE_ECDSA_NULL_SHA256
- Windows Linux UNIX ECDHE_RSA_NULL_SHA256
- Windows Linux UNIX TLS_RSA_WITH_NULL_NULL

有关更多信息，请参阅不推荐使用的 CipherSpec。

IBM MQ 使用 Microsoft Visual Studio 2005 构建的 C++ 客户机库

Windows

IBM MQ 8.0 C++ 客户机提供面向对象的应用程序编程接口 (API)，用于以 C++ 语言开发消息传递应用程序。 IBM MQ 8.0 C++ 库是使用 Microsoft Visual Studio 2012 C++ 编译器构建的。 使用 Microsoft Visual Studio 2012 编译器构建的 IBM MQ C++ 库与使用 IBM MQ 8.0 C++ 库之前的 IBM WebSphere MQ 发行版开发的 IBM MQ C++ 应用程序不兼容。 这是因为 Microsoft Visual Studio 2012 C++ 标准库不向后兼容。

从 V 8.0.0 修订包 2 开始，IBM MQ 提供了使用 Microsoft Visual Studio 2005 C++ 编译器构建的 C++ 客户机库。 除了使用 Microsoft Visual Studio 2012 C++ 编译器构建的现有 IBM MQ 8.0 C++ 库外，还提供了这些库。 使用 IBM MQ 早于 版本 8.0 的发行版构建的应用程序可以使用这些库，因为这些库与较早版本兼容。 有关更多信息，请参阅 在 Windows 上构建 C++ 程序。

V 8.0.0.4 有关在同一机器上使用使用 Microsoft Visual Studio 2012 和 Microsoft Visual Studio 2005 编译的 IBM MQ C++ 的信息，请参阅 第 63 页的『以不同方式命名的 IBM MQ C++ 库』。

独立 IBM MQ.NET 客户机

Windows

从 V 8.0.0 修订包 2 开始， IBM MQ.NET 客户机使您能够打包和部署组合件，而无需在生产系统上安装完整的 IBM MQ 客户机以运行应用程序。 您可以在安装了完整 IBM MQ 客户机的机器上构建 IBM MQ.NET 应用程序，然后将 IBM MQ.NET 组合件 (即 `amqmdnet.dll`) 与应用程序一起打包，并在生产系统上进行部署。 这些应用程序可以是在 Azure PaaS 环境中运行的传统 Windows.NET 应用程序，服务或 Microsoft Azure Web/工作程序应用程序。

在此类部署中， IBM MQ.NET 客户机仅支持与队列管理器的受管方式连接。 服务器绑定和非受管客户机方式的连接不可用，因为这两种方式需要完整的 MQ LightIBM MQ 客户机安装。 有关更多信息，请参阅 使用独立 IBM MQ.NET 客户机。

IBM MQ Virtual System Pattern Type

Linux

IBM MQ Virtual System Pattern Type for IBM PureApplication System（在本文档中简称 IBM MQ Virtual System Pattern Type）包含 IBM MQ 插件（虚拟系统软件组件）和关联的脚本包，您可以用于为 IBM MQ 环境创建虚拟系统模式。 有关更多信息，请参阅 IBM MQ Virtual System Pattern Type 概述。

CICS OSGi JVM 服务器中的 IBM MQ classes for JMS

z/OS

V 8.0.0 修订包 2 添加了在 CICS Open Services Gateway Initiative (OSGi) Java 虚拟机 (JVM) 服务器的特定版本中使用 IBM MQ classes for JMS 的支持。

有关更多信息，请参阅 在 CICS OSGi JVM 服务器中使用 IBM MQ classes for JMS

JMS 异常侦听器更改

V 8.0.0.2

从 V 8.0.0 修订包 2 开始，缺省情况下， IBM MQ classes for JMS 仅在连接中断时向异常侦听器传递异常。此行为与 JMS 规范一致。 要让 IBM MQ classes for JMS 将所有异常通知到异常侦听器，您的应用程序必须将连接工厂的 ASYNCEXCEPTION 属性设置为 ASYNC_EXCEPTIONS_ALL。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

相关信息

IBM MQ 系统需求

IBM MQ、WebSphere MQ 和 MQSeries 产品自述文件 Web 页面

版本 8.0 修订包的新增，已更改和已除去消息

针对 V 8.0.0 修订包 5 中的 IBM MQ 8.0 修订包添加，更改或除去的 IBM MQ AMQ 和 IBM MQ Managed File Transfer 消息的摘要，以及指向更多信息的链接。 自 版本 8.0.0.0 以来，已为 IBM MQ for z/OS 添加，更改或除去消息的摘要。

新增，已更改和已除去 IBM MQ AMQ 消息

针对 IBM MQ 8.0.0 修订包 5 或更高版本添加，更改或除去了大量 IBM MQ AMQ 消息。

- 第 67 页的『Version 8.0.0, Fix Pack 16 上新增，更改和除去的 AMQ 消息』
- 第 68 页的『V 8.0.0FP15 上新增，更改和除去的 AMQ 消息』
- 第 68 页的『V 8.0.0FP14 上新增，更改和除去的 AMQ 消息』
- 第 68 页的『Version 8.0.0, Fix Pack 13 上新增，更改和除去的 AMQ 消息』
- 第 68 页的『V 8.0.0FP12 上新增，更改和除去的 AMQ 消息』
- 第 68 页的『V 8.0.0FP11 上新增，更改和除去的 AMQ 消息』
- 第 69 页的『V 8.0.0FP10 上新增，更改和除去的 AMQ 消息』
- 第 69 页的『V 8.0.0 修订包 9 上新增，更改和除去的 AMQ 消息』
- 第 69 页的『V 8.0.0 修订包 8 上新增，更改和除去的 AMQ 消息』
- 第 70 页的『V 8.0.0 修订包 7 上新增，更改和除去的 AMQ 消息』
- 第 70 页的『V 8.0.0 修订包 6 上新增，更改和除去的 AMQ 消息』
- 第 70 页的『V 8.0.0 修订包 5 上新增，更改和除去的 AMQ 消息』

Version 8.0.0, Fix Pack 16 上新增，更改和除去的 AMQ 消息

V 8.0.0.16

以下消息已在 Version 8.0.0, Fix Pack 16 处更改。

- AMQ9456 ("队列" 在 "消息和说明" 中替换为 "对象")

Version 8.0.0, Fix Pack 16 上没有新消息或已除去的消息。

V 8.0.0FP15 上新增，更改和除去的 AMQ 消息

V 8.0.0.15

V 8.0.0FP15 上新增了以下消息。

- AMQ3572
- AMQ3573

V 8.0.0FP15 上没有已更改或已除去的消息。

V 8.0.0FP14 上新增，更改和除去的 AMQ 消息

V 8.0.0.14

V 8.0.0FP14 上新增了以下消息。

- AMQ3961
- AMQ7301

V 8.0.0FP14 上没有已更改或已除去的消息。

Version 8.0.0, Fix Pack 13 上新增，更改和除去的 AMQ 消息

V 8.0.0.13

以下消息已在 Version 8.0.0, Fix Pack 13 处更改。

- AMQ9545 (已从类型 "错误" 更改为类型 "信息")

Version 8.0.0, Fix Pack 13 上没有新消息或已除去的消息。

V 8.0.0FP12 上新增，更改和除去的 AMQ 消息

V 8.0.0.12

V 8.0.0FP12 上新增了以下消息:

- AMQ9568

以下消息已在 V 8.0.0FP12 处更改。

- AMQ6245 (Windows) (除去单词 "file")
- AMQ8416 (消息和响应已扩展)
- AMQ9512 (已更正 APAR IT26960 的说明)
- AMQ9513 (已扩展说明)

V 8.0.0FP12 上没有已除去的消息。

V 8.0.0FP11 上新增，更改和除去的 AMQ 消息

V 8.0.0.11

V 8.0.0FP11 上新增了以下消息:

- AMQ9567

以下消息在 V 8.0.0FP11 处已更改。

- AMQ9467 (响应已缩短，并且已修复指向原因码的链接)

V 8.0.0FP11 上没有已除去的消息。

V 8.0.0FP10 上新增，更改和除去的 AMQ 消息

V 8.0.10

V 8.0.0FP10 上没有新消息:

以下消息在 V 8.0.0FP10 处已更改。

- AMQ9467 (*Application Programming Reference* 手册 已更改为 产品文档)

V 8.0.0FP10 上没有已除去的消息。

V 8.0.0 修订包 9 上新增，更改和除去的 AMQ 消息

V 8.0.9

V 8.0.0 修订包 9 上新增了以下消息:

- AMQ3937

V 8.0.0 修订包 9 处没有已更改的消息。

在 V 8.0.0 修订包 9 上除去了以下消息:

- AMQ9224 (Windows)

V 8.0.0 修订包 8 上新增，更改和除去的 AMQ 消息

V 8.0.8

V 8.0.0 修订包 8 上新增了以下消息:

- AMQ8928
- AMQ9880

在 V 8.0.0 修订包 8 处更改了以下消息:

- AMQ5703
- AMQ6052
- AMQ6529
- AMQ6530
- AMQ6532
- AMQ6533
- AMQ6537
- AMQ6538
- AMQ6648
- AMQ6657
- AMQ6658
- AMQ6659
- AMQ6660
- AMQ6662
- AMQ6663
- AMQ6668
- AMQ9224
- AMQ9456
- AMQ9469
- AMQ9785

V 8.0.0 修订包 8 上没有已除去的消息。

V 8.0.0 修订包 7 上新增，更改和除去的 AMQ 消息

V 8.0.0.7

V 8.0.0 修订包 7 上新增了以下消息:

- AMQ3970
- AMQ3971

在 V 8.0.0 修订包 7 处更改了以下消息:

- AMQ6642
- AMQ9616

在 V 8.0.0 修订包 7 上除去了以下消息:

- AMQ6599

V 8.0.0 修订包 6 上新增，更改和除去的 AMQ 消息

V 8.0.0.6

V 8.0.0 修订包 6 上新增了以下消息:

- AMQ9849

在 V 8.0.0 修订包 6 处更改了以下消息:

- AMQ4635
- AMQ9557
- AMQ8602
- AMQ8617

V 8.0.0 修订包 6 上没有已除去的消息。

V 8.0.0 修订包 5 上新增，更改和除去的 AMQ 消息

V 8.0.0.5

V 8.0.0 修订包 5 上新增了以下消息:

- AMQ6520 到 AMQ6534

在 V 8.0.0 修订包 5 处更改了以下消息:

- AMQ4501
- AMQ5607_ibm
- AMQ6535
- AMQ6539
- AMQ6544
- AMQ6572
- AMQ6576
- AMQ6577
- AMQ6582
- AMQ6583
- AMQ6584
- AMQ6585
- AMQ6587

- AMQ6588
- AMQ6594
- AMQ6619
- AMQ6623
- AMQ6670 到 AMQ6675
- AMQ6678
- AMQ6684
- AMQ6692
- AMQ9874

在 V 8.0.0 修订包 5 上除去了以下消息:

- AMQ8067
- AMQ8302
- AMQ8303
- AMQ8305

相关信息

IBM MQ AMQ 消息

新增，已更改和已除去 IBM MQ Managed File Transfer 条消息

已为 IBM MQ Managed File Transfer 添加多条新消息。 已除去先前发行版中可用的某些消息。

- 第 71 页的『新增，已更改和已移除 Managed File Transfer 条消息 (位于 Version 8.0.0, Fix Pack 16)』
- 第 71 页的『新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0FP15)』
- 第 72 页的『新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0FP14)』
- 第 72 页的『新增，已更改和已移除 Managed File Transfer 条消息 (位于 Version 8.0.0, Fix Pack 13)』
- 第 72 页的『新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0FP12)』
- 第 72 页的『新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0FP11)』
- 第 72 页的『新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0FP10)』
- 第 72 页的『新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0 修订包 9)』
- 第 73 页的『新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0 修订包 8)』
- 第 73 页的『新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0 修订包 7)』
- 第 73 页的『新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0 修订包 6)』
- 第 74 页的『新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0 修订包 5)』

新增，已更改和已移除 Managed File Transfer 条消息 (位于 Version 8.0.0, Fix Pack 16)

V 8.0.0.16

Version 8.0.0, Fix Pack 16 上没有新的，已更改或已除去的消息。

新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0FP15)

V 8.0.0.15

V 8.0.0FP15 上新增了以下消息:

- BFGTR0083E

V 8.0.0FP15 上没有已更改或已除去的消息。

新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0FP14)

V 8.0.0.14

V 8.0.0FP14 上没有新的，已更改或已除去的消息。

新增，已更改和已移除 Managed File Transfer 条消息 (位于 Version 8.0.0, Fix Pack 13)

V 8.0.0.13

Version 8.0.0, Fix Pack 13 上没有新的，已更改或已除去的消息。

新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0FP12)

V 8.0.0.12

V 8.0.0FP12 上没有新的，已更改或已除去的消息。

新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0FP11)

V 8.0.0.11

V 8.0.0FP11 上没有新的，已更改或已除去的消息。

新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0FP10)

V 8.0.0.10

V 8.0.0FP10 上没有新的或已更改的消息:

在 V 8.0.0FP10 上除去了以下消息。

- BFGCL0745-BFGCL0749
- BFGCL0753-BFGCL0757

新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0 修订包 9)

V 8.0.0.9

V 8.0.0 修订包 9 上新增了以下消息:

- BFGCL0745E
- BFGCL0746E
- BFGCL0747E
- BFGCL0748I
- BFGCL0749E
- BFGCL0753E
- BFGCL0754I
- BFGCL0755I
- BFGCL0757E
- BFGCL0761E
- BFGCL0762E
- BFGCL0763E
- BFGCL0764I
- BFGCL0765I
- BFGCL0766I
- BFGCL0767I

- BFGCL0768I
- BFGCL0769I
- BFGCL0770W
- BFGCL0771W
- BFGCL0772I
- BFGCL0773I

V 8.0.0 修订包 9 上没有已更改或已除去的消息。

新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0 修订包 8)

V 8.0.0.8

V 8.0.0 修订包 8 上没有新的，已更改或已除去的消息。

新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0 修订包 7)

V 8.0.0.7

V 8.0.0 修订包 7 上新增了以下消息:

- BFGAG0187E
- BFGBR0208E
- BFGCL0752E
- BFGJE0006W
- BFGPC0053E

在 V 8.0.0 修订包 7 处更改了以下消息:

- BFGBR0065E
- BFGJE0001W
- BFGTR0077I

V 8.0.0 修订包 7 上没有已除去的消息。

新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0 修订包 6)

V 8.0.0.6

V 8.0.0 修订包 6 上新增了以下消息:

- BFGPC0008E
- BFGSS0077E
- BFGSS0078E
- BFGSS0079E
- BFGSS0080E
- BFGTR0078W
- BFGTR0079E
- BFGTR0080E

在 V 8.0.0 修订包 6 处更改了以下消息:

- BFGTR0062I
- BFGTR0063I
- BFGTR0067I
- BFGTR0068I

V 8.0.0 修订包 6 上没有已除去的消息。

新增，已更改和已移除 Managed File Transfer 条消息 (位于 V 8.0.0 修订包 5)

V 8.0.0.5

V 8.0.0 修订包 5 上新增了以下消息:

- BFGAP0064 到 BFGAP0065
- BFGBR0187 到 BFGBR0207
- BFGMQ1042 到 BFGMQ1043

以下消息已在 V 8.0.0 修订包 5 处更改:

- BFGCM0264

在 V 8.0.0 修订包 5 上除去了以下消息:

- BFGDM0122

相关信息

IBM MQ Managed File Transfer 诊断消息

自 版本 8.0.0.0 以来新增，更改和除去了 z/OS 条消息

自 版本 8.0.0.0 以来，已为 IBM MQ for z/OS 添加，更改或除去了大量消息。

- 第 74 页的『新的 z/OS 消息』
- 第 75 页的『已更改 z/OS 条消息』
- 第 77 页的『已除去 z/OS 条消息』

新的 z/OS 消息

自 版本 8.0.0.0 以来，新增了以下消息:

耦合设施管理器消息 (CSQE ...)

CSQE159I

CSQE161E

安全管理器消息 (CSQH ...)

CSQH046E

数据管理器消息 (CSQI ...)

CSQI039E

CSQI052E

恢复日志管理器消息 (CSQJ ...)

CSQJ164I

CSQJ168I

消息管理器消息 (CSQM ...)

CSQM001E

CSQM062I

CSQM102E

CSQM580I

主题管理器消息 (CSQT ...)

CSQT987E 到 CSQT990E

实用程序消息 (CSQU ...)

CSQU510I

CSQU511I

CSQU586I

CSQU587I

代理程序服务消息 (CSQV ...)

CSQV460I

CSQV461D

分布式排队消息 (CSQX ...)

CSQX117I

CSQX179I

CSQX180I

CSQX571E

CSQX674E

CSQX694I 到 CSQX695I

CSQX697I (重新编号为 CSQX696I , 位于 V 8.0.0FP15)

CSQX698I (重新编号为 CSQX697I , 位于 V 8.0.0FP15)

CSQX785E

服务设施消息 (CSQ1...)

CSQ1219I

子系统支持消息 (CSQ3...)

CSQ3018

Db2 管理器消息 (CSQ5...)

CSQ5038I

CSQ5039I

已更改 z/OS 条消息

在所有消息中都进行了以下更改:

- 链接已从信息中心链接迁移到 IBM Documentation 链接。
- 指向 "MQSC 参考" 的链接重新指向更精确的目标。
- 指向 "管理 IBM MQ for z/OS" 的链接将重新指向更精确的目标。
- 对其他属性或消息的引用将转换为链接。
- IBM WebSphere MQ 已更改为 IBM MQ。
- DB2 已重命名为 Db2。

自 版本 8.0.0.0 以来，将更改以下消息:

CICS 适配器和网桥消息 (CSQC ...)

CSQC381D (响应部分已除去链接。)

CSQC389D (响应部分已除去链接。)

耦合设施管理器消息 (CSQE ...)

CSQE038E (已改进 "说明" 部分。)

CSQE042E ("响应" 部分已扩展。)

CSQE137E (+ cpf 已更改为 /cpf。)

CSQE222E ("响应" 部分已扩展。)

安全管理器消息 (CSQH ...)

CSQH045E (已更正 "说明" 部分语法错误并添加了链接。)

数据管理器消息 (CSQI ...)

CSQI004I (已扩展 "响应" 部分。)

CSQI010I (代码块已更新。)

CSQI045I (已扩展系统程序员响应。)

CSQI046E (系统程序员响应已扩展。)

CSQI047E (系统程序员响应已扩展。)

CSQI063E (响应部分已更改链接。)

CSQI065I (已更新代码块并添加了有关 Page 类的信息。)

恢复日志管理器消息 (CSQJ ...)

CSQJ031D (系统程序员响应已扩展。)

CSQJ032E (系统程序员响应已扩展。)

CSQJ122E (响应部分已扩展。)

CSQJ134E (已修正 "解释" 部分扩展和输入错误。)

CSQJ151I (响应部分已除去链接。)

CSQJ152I 到 CSQJ154I (响应部分已除去链接。)

CSQJ212E (链接已更新。)

CSQJ213E (已更新链接。)

消息管理器消息 (CSQM ...)

CSQM078E "说明" 部分已扩展。

命令服务器消息 (CSQN ...)

CSQN202I (已纠正输入错误。)

缓冲区管理器消息 (CSQP ...)

CSQP014E (命令已更新 (QDEPTH 变为 CURDEPTH) , 并且在 "系统程序员响应" 中添加了额外段落。)

IMS 适配器消息 (CSQQ ...)

CSQQ153I (响应部分已除去链接。)

恢复管理器消息 (CSQR ...)

CSQR030I ("说明" 部分已扩展)。

实用程序消息 (CSQU ...)

CSQU050E (响应部分已除去链接。)

CSQU071E (响应部分已除去链接。)

CSQU112E (响应部分已除去链接。)

CSQU181E (说明和响应已缩短。)

CSQU505E 到 CSQU509E (响应部分已除去链接。)

CSQU564E (响应部分已更改链接。)

检测设施消息 (CSQW ...)

CSQW124E (已改进 "说明" 部分。)

分布式排队消息 (CSQX ...)

CSQX004I (在 "说明" 中添加了额外段落。)

CSQX126I 到 CSQX129I (已除去响应部分。)

CSQX192E (已重写和扩展 "操作和响应" 部分。)

CSQX294E (已改进 "响应" 部分。)

CSQX428E (消息文本已扩展。)

CSQX432I (消息文本已改进。)

CSQX503E (在 "说明" 中更新了消息并添加了额外段落。)

CSQX504E (添加了错误代码的长列表。)

CSQX511I ("说明" 部分已除去。)

CSQX512I ("说明" 部分已重写。)

CSQX519E ("说明" 部分已扩展。)

CSQX531E (响应部分已扩展。)

CSQX565E ("说明" 部分已扩展。)

CSQX630E (已扩展响应部分。)

CSQX634E (已扩展响应部分。)

CSQX645E (响应部分已扩展。)

CSQX658E (在 System Programmer Response 中固定了 Typo。)

CSQX697I 重新编号为 CSQX696I。 (在 "说明" 中修正了典型。)

CSQX698I 重新编号为 CSQX697I。

初始化过程和常规服务消息 (CSQY ...)

CSQY221I 响应部分已扩展。

CSQY291E 基本扩展和重写的解释和系统程序员响应

高级消息安全性 (CSQ0...)

CSQ0209E 响应部分已扩展。

CSQ0629E 添加了 Extra 参数 "errno2"。

IBM MQ-IMS 网桥消息 (CSQ2...)

CSQ2004E (已扩展 "说明" 部分。 有关 CSQ2PUTD 的额外段落将添加到 "说明" 和 "系统程序员响应" 中。)

CSQ2005I (在 "说明" 中添加了 Typo 固定段落和额外段落。)

子系统支持消息 (CSQ3...)

CSQ3111I ("说明" 部分已扩展。)

CSQ3202E ("说明" 部分已扩展。)

Db2 管理器消息 (CSQ5...)

CSQ5005E (已更新并扩展 "响应" 部分。)

通用命令预处理器消息 (CSQ9...)

CSQ9014E (已更新并扩展 "响应" 部分。)

CSQ9025E (已纠正输入错误。)

已除去 z/OS 条消息

自 版本 8.0.0.0 以来没有已除去的消息。

相关概念

第 45 页的『版本 8.0.0 的新增和已更改的 z/OS 消息』
已添加大量新消息，并且已更改 IBM MQ for z/OS 版本 8.0.0 的某些现有消息。

相关信息

IBM MQ for z/OS 的消息

先前版本中新增和更改的内容

指向有关在 IBM MQ 8.0 之前的产品版本中发生的新功能以及对功能和资源的更改 (包括稳定化，废弃和移除) 的信息的链接。

有关产品的先前版本中新增内容和更改内容的信息，请参阅该版本的产品文档中相应的部分。

IBM WebSphere MQ 7.5

- IBM WebSphere MQ V 7.5 中的新增内容
- IBM WebSphere MQ V 7.5 中更改的内容
- IBM WebSphere MQ V 7.5 修订包中的更改内容
- 在 IBM WebSphere MQ Advanced Message Security V 7.0.1 与 IBM WebSphere MQ V 7.5
- IBM WebSphere MQ File Transfer Edition

- IBM WebSphere MQ File Transfer Edition V7.0.4 或更低版本与 IBM WebSphere MQ V7.5 之间的更改

IBM WebSphere MQ 7.1

- IBM WebSphere MQ V 7.1 中的新增内容
- IBM WebSphere MQ V 7.1 中更改的内容
- IBM WebSphere MQ V 7.1 修订包中更改的内容

IBM WebSphere MQ 7.0.1 和更低版本

对于较旧版本的产品 (在 IBM Documentation 外部提供了文档)， 请参阅 Documentation 以了解较旧版本的 IBM MQ。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 22 页的『IBM MQ 8.0.0 中更改的内容』

第 46 页的『IBM MQ 8.0 修订包中新增和更改的内容』
本节描述了 版本 8.0 修订包中的新增功能和已更改的功能。

相关信息

迁移方法和概念概述

IBM MQ 8.0 系统需求

IBM MQ、WebSphere MQ 和 MQSeries 产品自述文件 Web 页面

声明

本信息是为在美国提供的产品和服务编写的。

IBM 可能在其他国家或地区不提供本文档中包含的产品、服务或功能特性。 有关您当前所在区域的产品和服务的信息，请向您当地的 IBM 代表咨询。 任何对 IBM 产品、程序或服务的引用并非意在明示或默示只能使用 IBM 的产品、程序或服务。 只要不侵犯 IBM 的知识产权，任何同等功能的产品、程序或服务都可以代替 IBM 产品、程序或服务。 但是， 评估和验证任何非 IBM 产品、程序或服务的操作，由用户自行负责。

IBM 公司可能已拥有或正在申请与本文档内容有关的各项专利。 提供本文档并未授予用户使用这些专利的任何许可。 您可以以书面形式将许可查询寄往:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

有关双字节（DBCS）信息的许可查询，请与您所在国家或地区的 IBM 知识产权部门联系，或用书面方式将查询寄往:

知识产权许可
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

本条款不适用英国或任何这样的条款与当地法律不一致的国家或地区: International Business Machines Corporation “按现状” 提供本出版物，不附有任何种类的 (无论是明示的还是暗含的) 保证，包括但不限于暗含的有关非侵权，适销和适用于某种特定用途的保证。 某些国家或地区在某些交易中不允许免除明示或暗含的保证。因此本条款可能不适用于您。

本信息中可能包含技术方面不够准确的地方或印刷错误。 此处的信息将定期更改；这些更改将编入本资料的新版本中。 IBM 可以随时对本出版物中描述的产品和/或程序进行改进和/或更改，而不另行通知。

本信息中对非 IBM Web 站点的任何引用都只是为了方便起见才提供的，不以任何方式 充当对那些 Web 站点的保证。 那些 Web 站点中的资料不是 IBM 产品资料的一部分，使用那些 Web 站点带来的风险将由您自行承担。

IBM 可以按它认为适当的任何方式使用或分发您所提供的任何信息而无须对您承担任何责任。

本程序的被许可方如果要了解有关程序的信息以达到如下目的：(i) 允许在独立创建的程序和其他程序（包括本程序）之间进行信息交换，以及 (ii) 允许对已经交换的信息进行相互使用，请与下列地址联系：

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

只要遵守适当的条件和条款，包括某些情形下的一定数量的付费，都可获得这方面的信息。

本资料中描述的许可程序及其所有可用的许可资料均由 IBM 依据 IBM 客户协议、IBM 国际软件许可协议或任何同等协议中的条款提供。

此处包含的任何性能数据都是在受控环境中测得的。 因此，在其他操作环境中获得的结果可能会明显不同。 有些测量可能是在开发级的系统上进行的，因此不保证与一般可用系统上进行的测量结果相同。 此外，有些测量可能是通过推算而估计的。 实际结果可能会不同。 本文档的用户应当验证其特定环境的适用数据。

涉及非 IBM 产品的信息可从这些产品的供应商、其出版说明或其他可公开获得的资料中获取。 IBM 没有对这些产品进行测试，也无法确认其性能的精确性、兼容性或任何其他关于非 IBM 产品的声明。 有关非 IBM 产品性能的问题应当向这些产品的供应商提出。

有关 IBM 未来方向或意向的所有声明均可能未经通知即变更或撤销，并且仅代表目标和目的。

本信息包含日常商业运作所使用的数据和报表的示例。 为了尽可能全面地说明这些数据和报表，这些示例包括个人、公司、品牌和产品的名称。 所有这些名称都是虚构的，如与实际商业企业所使用的名称和地址有任何雷同，纯属巧合。

版权许可：

本信息包含源语言形式的样本应用程序，用以阐明在不同操作平台上的编程技术。 如果是为按照在编写样本程序的操作平台上的应用程序编程接口 (API) 进行应用程序的开发、使用、经销或分发为目的，您可以任何形式对这些样本程序进行复制、修改、分发，而无须向 IBM 付费。 这些示例并未在所有条件下作全面测试。 因此，IBM 不能担保或默示这些程序的可靠性、可维护性或功能。 这些样本程序“按现状”提供，不附有任何种类的保证。 对于因使用样本程序所引起的任何损害，IBM 概不负责。

凡这些样本程序的每份拷贝或其任何部分或任何衍生产品，都必须包括如下版权声明：©（贵公司的名称）（年）。 此代码的部分来自 IBM 公司的 样本程序。 © Copyright IBM Corp. 2004 , 2023. All rights reserved.

如果您正在查看本信息的软拷贝，那么图片和彩色图例可能无法显示。

商标

IBM， IBM 徽标 ibm.com， Passport Advantage， WebSphere， MQSeries 和 z/OS 是 International Business Machines Corp. 的商标或注册商标。 在全球许多管辖区域注册的商标或注册商标。 其他产品和服务名称可能是 IBM 或其他公司的商标。 当前的 IBM 商标列表可从 Web 站点 https://www.ibm.com/legal/copytrade.shtml 上的 "版权和商标信息" 部分获取。

Microsoft、Windows、Windows NT 和 Windows 徽标是 Microsoft Corporation 在美国和/或其他国家或地区的商标。

Java 和所有基于 Java 的商标和徽标是 Oracle 和/或其附属公司的商标或注册商标。

Adobe、Adobe 徽标、PostScript 和 PostScript 徽标是 Adobe Systems Incorporated 在美国和/或其他国家或地区的注册商标或商标。

UNIX 是 Open Group 在美国和其他国家或地区的注册商标。

此处包括的 Oracle Outside In Technology 遵循受限的使用许可证，并且仅能与此应用程序结合使用。

Intel 是 Intel Corporation 或其子公司在美国和其他国家或地区的商标或注册商标。

根据来自 Linux Foundation 的再许可使用注册商标 Linux，这是 Linus Torvalds 的专用被许可方，该商标在全球范围内的所有者。

IBM MQ 的辅助功能选项

辅助功能可以帮助残障用户（如行动不便或视力不佳的用户）成功使用信息技术产品。

以下列表包含 IBM MQ 中的主要辅助功能选项。 您可以使用屏幕朗读器软件来聆听屏幕上的显示内容。

- 支持仅限于键盘的操作
- 支持屏幕朗读器的常用接口

使用 IBM MQ 的完全可访问方法是使用命令行界面。 有关使用命令的更多信息，请参阅 如何使用 IBM MQ 控制命令 和 使用 MQSC 命令执行本地管理任务。

对于 Windows，安装 IBM MQ 的可访问方式是使用非交互式安装。 有关更多信息，请参阅 使用 msiexec 进行高级安装。

请访问 https://www.ibm.com/able，以获取有关 IBM 对辅助功能选项所作承诺的更多信息。

相关信息

MQ Explorer 中的辅助功能选项

产品文档中的发行版和平台图标

IBM Documentation 中的单个信息集中提供了所有 IBM MQ 8.0 修订包和平台的产品文档。 如果特定于给定修订包或平台的信息，那么此信息由矩形图标指示。 适用于所有修订包和平台的信息保持未标记状态。

修订包编号图标

将在修订包中交付对 IBM MQ 8.0 的更新。 在给定修订包中新增的信息由包含修订包编号的深蓝色图标标识。 例如: V 8.0.0.1

平台图标

平台图标用于标识仅适用于特定平台或平台组的信息。

表 3: 平台图标

图标	平台
AIX	AIX
HP-UX	HP-UX.
Solaris	Solaris.
UNIX	所有受支持的 UNIX 平台。 即，此表中先前列出的所有平台。
Linux	Linux
Windows	Windows
IBM i	IBM i.
distributed	分布式。 除 z/OS 以外的所有受支持平台。 即，此表中先前列出的所有平台。
z/OS	z/OS

注: 未显式标记为 IBM i 或 z/OS 的任何信息也适用于 IBM MQ Appliance。

相关概念

第 13 页的『IBM MQ 8.0.0 中的新增功能』
IBM MQ 8.0.0 提供了改进的扩展和灵活性，使其能够更好地支持大规模消息传递企业解决方案。

第 46 页的『IBM MQ 8.0 修订包中新增和更改的内容』
本节描述了 版本 8.0 修订包中的新增功能和已更改的功能。

第 77 页的『先前版本中新增和更改的内容』
指向有关在 IBM MQ 8.0 之前的产品版本中发生的新功能以及对功能和资源的更改 (包括稳定化，废弃和移除) 的信息的链接。

IBM MQ 8.0 快速入门指南

使用本指南可以实现 IBM MQ 8.0 入门。

本地语言版本

To obtain the Quick Start Guide in other languages, print the language-specific PDF from the Quick Start DVD.

产品概述

WebSphere MQ 提供了通用的消息传递主干，它具有强大的连接能力，可以为应用程序提供灵活可靠的消息传递功能，并能够集成各种采用了面向服务的体系结构 (SOA) 的现有 IT 资产。

WebSphere MQ 通过其消息排队接口来应用程序可移植性，该接口是一个在所有受支持的平台上都可以使用的统一应用程序编程接口。 WebSphere MQ 还支持 Java 消息服务 (JMS) 应用程序编程接口以及基于队列的消息传递和发布/预订消息传递。 “受管文件传输”可以扩展您的 WebSphere MQ 消息传递网络以支持基于文件的集成，此外还提供广泛的安全功能，可以用于保护消息传递系统和通过此系统的消息。

您可以使用 WebSphere MQ Explorer 图形用户界面来管理和配置整个 WebSphere MQ 网络。

访问软件和文档。

本产品包括以下项:

- “快速入门”DVD，其中包含美国英语和其他语言版本的《快速入门指南》。
- 对于每一个支持平台，此 DVD 包含运行时组件代码和其他所需产品。

如果从 IBM Passport Advantage 下载产品，请参阅 Passport Advantage 和 Passport Advantage Express (https://www.ibm.com/software/passportadvantage/) Web 站点以获取更多信息。

可从 https://www.ibm.com/software/integration/wmq/library/获取所有版本的 WebSphere MQ 的产品文档。 具体而言， WebSphere MQ V 8.0 产品文档也在 IBM Documentation (https://www.ibm.com/docs/SSFKSJ_8.0.0/com.ibm.mq.helphome.v80.doc/WelcomePagev8r0.htm) 中提供。

您还可以从 https://public.dhe.ibm.com/software/integration/wmq/docs/V8.0/PDFs 以一系列 PDF 文件形式下载 WebSphere MQ Version 8.0 产品文档。

文档中提供了服务和支持信息。

有关如何使用 WebSphere MQ Explorer 的信息，既可以从 WebSphere MQ Explorer 获取，也可以从产品文档中获取。

评估硬件和系统需求

有关所有受支持平台上的硬件和软件需求的详细信息，请参阅 System Requirements for WebSphere MQ (https://www.ibm.com/software/integration/wmq/requirements/)。

查看安装体系结构

有关产品的各个组件如何工作的概述，请参阅产品文档中的主题 WebSphere MQ (https://www.ibm.com/docs/SSFKSJ_8.0.0/com.ibm.mq.pro.doc/q001020_.htm)。

WebSphere MQ 体系结构中既包括使用单个队列管理器的简单体系结构，也包括互相连接的队列管理器的复杂网络。 有关规划 WebSphere MQ 体系结构的更多信息，请参阅产品文档的 规划 (https://www.ibm.com/docs/SSFKSJ_8.0.0/com.ibm.mq.pla.doc/q004690_.htm) 部分。

安装产品

有关 AIX， HP-UX， Linux， Solaris， IBM i 或 Microsoft Windows 上 WebSphere MQ 的安装指示信息，以及有关所需硬件和软件配置的详细信息， 请参阅产品文档的 安装 (https://www.ibm.com/docs/SSFKSJ_8.0.0/com.ibm.mq.ins.doc/q008250_.htm) 部分。

有关 z/OS 上 WebSphere MQ 的安装指示信息以及所需硬件和软件配置的详细信息，请参阅产品文档的 安装 WebSphere MQ for z/OS (https://www.ibm.com/docs/SSFKSJ_8.0.0/com.ibm.mq.ins.doc/q009640_.htm) 部分。

入门

产品文档的 方案 (https://www.ibm.com/docs/SSFKSJ_8.0.0/com.ibm.mq.pro.doc/q004360_.htm) 部分中的 入门 方案说明了如何在 Windows 上开始使用 WebSphere MQ 。 如果您此前没有使用过 WebSphere MQ，并且希望快速入门，请使用此方案。

其他方案通过引导您完成相应的任务步骤来帮助您配置或使用产品的功能部件。 这些方案中包括其他内容的链接，可以帮助您更好地了解感兴趣的领域。

详细信息

有关 WebSphere MQ 的更多信息，请参考以下资源：

产品自述文件
产品自述文件 (`readme.html`) 包含在产品介质中，安装产品组件时会安装自述文件。 产品自述文件 Web 页面 (https://www.ibm.com/support/docview.wss?rs=171&uid=swg27006097) 上提供了最新版本。

IBM 支持门户网站
IBM 支持门户网站中提供的支持信息包括以下资源：

- 支持技术说明 (https://www.ibm.com/support/search.wss? q = websphere + mq)
- 可用的下载和其他资源 (https://www.ibm.com/support/home/product/P439881V74305Y86/IBM_MQ)
- WebSphere 和 CICS 支持的社交媒体通道 (https://www.ibm.com/support/docview.wss?uid=swg21410956#2)

声明

IBM WebSphere MQ Version 8.0 Licensed Materials-Property of IBM. © Copyright IBM Corp. 2006 , 2022. U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp. IBM， IBM 徽标 ibm.com®， AIX， CICS， Passport Advantage, WebSphere 和 z/OS 是 International Business Machines Corp. 的商标或注册商标。 在全球许多管辖区域注册的商标或注册商标。 Java 和所有基于 Java 的商标和徽标是 Oracle 和/或其附属公司的商标或注册商标。 Linux 是 Linus Torvalds 在美国和/或其他国家或地区的商标。 Microsoft、Windows 和 Windows 徽标是 Microsoft Corporation 在美国和/或其他国家或地区的商标。 其他产品和服务名称可能是 IBM 或其他公司的商标。 当前的 IBM 商标列表可在 Web 站点上的 “版权和商标信息”（www.ibm.com/legal/copytrade.shtml） 部分获取。

部件号: CF4BZML

Printed in Ireland

IBM MQ 8.0 信息指南

信息指南包含指向各种 IBM MQ 8.0 资源的链接。

此路线图汇集了来自不同来源的信息，以帮助您了解有关 IBM MQ 的特定区域的更多信息。 单击指向路线图中每个部分的链接，以查看可用的资源。

- 产品概述
- 技术概述
- 方案
- 规划
- 迁移和升级
- 安装
- 安全性
- 配置
- 管理
- 开发应用程序
- 监控和性能
- 故障诊断与支持
- 引用型

表 *4: IBM MQ* 信息路线图表

类别	信息资源
产品概述	IBM MQ 的总体用途，功能和新功能概述。 **第 7 页的『关于 IBM MQ』** 用于帮助您开始使用 IBM MQ 8.0 的介绍性信息，包括产品简介以及本发行版的新增内容和已更改内容的概述。 **IBM MQ 8.0** 此 IBM Redbooks 出版物涵盖 IBM MQ 8.0 中的核心增强功能以及必须理解的概念。 **IBM MQ 产品 Web 页面** 包含指向资源和其他信息的链接的产品 Web 页面。 **IBM MQ 系统需求** 包含指向 IBM MQ 不同发行版的系统需求的链接的 Web 页面。 从 IBM MQ 8.0 开始，您可以使用软件产品兼容性报告 (SPCR) 工具来查找有关受支持操作系统，系统需求，先决条件和可选受支持软件的信息。 **第 85 页的『IBM Documentation Offline 应用程序中的 IBM MQ 8.0』** 您可以将 IBM MQ 8.0 消息传递文档下载到您本地安装的 IBM Documentation 的脱机版本中。
技术概述	**第 125 页的『IBM MQ 技术概述』** 用于帮助您了解消息排队以及 IBM MQ 8.0 提供的其他功能的信息。

表 4: IBM MQ 信息路线图表 (继续)	
类别	**信息资源**
方案	每个方案都会引导您完成一组重要任务，并帮助您配置主要产品功能部件。 这些场景包括指向其他内容的有用链接，以帮助您更好地了解感兴趣的区域。 **IBM MQ 方案** 入门 方案说明如何开始使用 IBM MQ。 如果您之前未使用过 IBM MQ ，并且想要快速入门，请使用此方案。 其他方案通过引导您完成相应的任务步骤来帮助您配置或使用产品的功能部件。 **将 WebSphere Application Server 连接到 IBM MQ** 包含在各种场景中引导您完成将 WebSphere Application Server 连接到 IBM MQ 所需的关键任务的信息。 **将 WebSphere Application Server Liberty 概要文件连接到 IBM MQ** 包含在各种场景中引导您完成将 WebSphere Application Server Liberty 概要文件连接到 IBM MQ 所需的关键任务的信息。 **将 IBM MessageSight 连接到 IBM MQ 和 WebSphere Application Server** 包含在各种场景中引导您完成将 IBM MessageSight 连接到 IBM MQ 和 WebSphere Application Server 所需的关键任务的信息。
规划	**规划** 规划 IBM MQ 环境时，请考虑 IBM MQ 为单个和多个队列管理器体系结构以及点到点和发布/预订消息传递样式提供的支持。 还要规划资源需求以及日志记录和备份工具的使用。
迁移和升级	**迁移和升级** 迁移是更新队列管理器和其他对象 (例如应用程序或管理过程) 的过程。 要迁移队列管理器以在新的代码级别上运行，必须首先升级 IBM MQ 以安装新的代码级别。 验证升级是否成功后，请迁移队列管理器以及与其关联的所有应用程序和资源。 在开始此过程之前，请根据本文档中的信息创建迁移计划。 distributed **IBM WebSphere MQ / IBM MQ 迁移指南** 本指南提供了一些信息，可帮助您规划在分布式平台上从旧版本迁移到新版本的过程。 您可以在 Web 浏览器中查看指南，也可以将其下载为 PDF 文件。 z/OS **IBM WebSphere MQ / IBM MQ for z/OS 迁移指南** 本指南提供了帮助您规划从旧版本迁移到 z/OS 上的新版本的过程的信息。 您可以在 Web 浏览器中查看指南，也可以将其下载为 PDF 文件。
安装	**安装和卸载** 用于帮助您准备安装，安装产品和验证安装的信息。 还有一些信息可以帮助您卸载产品。
安全性	**安全性** IBM MQ 安装中要考虑的安全性方面，包括标识和认证，授权，审计，机密性和数据完整性。

表 4: IBM MQ 信息路线图表 (继续)	
类别	**信息资源**
配置	**配置** 在一台或多台计算机上创建一个或多个队列管理器，并在开发，测试和生产系统上配置这些队列管理器及其相关资源，以处理包含业务数据的消息。
管理	**管理 IBM MQ** 管理队列管理器和关联的资源。
开发应用程序	**开发应用程序** 开发应用程序以发送和接收消息，以及管理队列管理器和相关资源。 IBM MQ 支持以过程化语言、面向对象语言和框架编写的应用程序。
监控和性能	**监视和性能** 监视信息和指导信息，以帮助提高队列管理器网络的性能，并调整提示以帮助提高队列管理器网络的性能。
故障诊断与支持	**故障诊断和支持** 帮助您诊断和解决队列管理器网络或 IBM MQ 应用程序问题的方法。 **IBM SupportAssistant Web 页面** IBM Support Assistant (ISA) 可通过提供对支持相关信息和故障诊断工具的访问权来帮助您解决 IBM 软件产品问题。 **IBM MQ 支持门户网站 Web 页面** IBM Support Portal for IBM MQ。 **IBM 支持论坛** 搜索 "MQ"。
引用型	**参考** 有关配置，管理，开发应用程序，遥测，安全性，监视，故障诊断和支持以及诊断消息的参考信息。

IBM Documentation Offline 应用程序中的 IBM MQ 8.0

如果您处于无法访问因特网的气郄环境中，请使用 "深色商店" 应用程序 "IBM Documentation Offline" 来查看 IBM MQ 8.0 产品文档的下载。

IBM Documentation Offline 具有两个组件:

- **IBM Documentation Offline 应用程序**。 这是 IBM Documentation 的本地可安装脱机版本。
- **安装到 IBM Documentation Offline 应用程序中的文档包**。 这些包包含与在 IBM Documentation 中在线发布的文档相同的文档。

要下载应用程序和 IBM MQ 8.0 文档包，您需要登录到 IBM Documentation。 有关更多信息，请参阅 IBM Documentation Offline。

产品文档和程序目录的 IBM MQ 8.0 PDF 文件

您可以下载 PDF 格式的 IBM MQ 8.0 文档和 IBM MQ for z/OS 程序目录。

IBM MQ 8.0 产品文档 PDF 文件

IBM MQ 8.0 PDF 文件包含与 IBM Documentation 中发布的 HTML 版本相同的信息。 您可以通过单击下表中的链接来下载这些内容。

注: 您还可以将 HTML 格式的 IBM MQ 8.0 产品文档下载到本地安装的脱机版本的 IBM Documentation 中。有关更多信息，请参阅第 85 页的『IBM Documentation Offline 应用程序中的 IBM MQ 8.0』。

表 *5:* 产品文档的 *PDF* 文件以及它们如何对应于 *IBM Documentation* 中的部分	
PDF 文件名和下载链接	**产品文档部分**
wmq80_qsg_en.pdf Translations of this Quick Start Guide are available to download 这里.	《快速入门指南》
mq80.overview.pdf	关于 IBM MQ 技术概述 方案
mq80.plan.pdf	规划 IBM MQ
mq80.migrate.pdf	维护和迁移 IBM MQ
mq80.install.pdf	安装 IBM MQ
mq80.secure.pdf	保护 IBM MQ
mq80.configure.pdf	配置 IBM MQ
mq80.administer.pdf	管理 IBM MQ
mq80.develop.pdf	为 IBM MQ 开发应用程序
mq80.monitor.pdf	IBM MQ 的监视和性能
mq80.troubleshoot.pdf	IBM MQ 的故障诊断和支持
mq80.reference.pdf	IBM MQ 参考
mq80.refcon.pdf	IBM MQ 配置参考
mq80.refadmin.pdf	IBM MQ 管理参考
mq80.refdev.pdf	IBM MQ 开发应用程序参考
mq80.mft.pdf	Managed File Transfer
mq80.explorer.pdf	MQ Explorer
mq80.guiwizard.pdf	IBM MQ GUI 向导帮助
mq80.vsys_patterntype.pdf	IBM MQ Virtual System Pattern Type
mq80.xms.pdf	IBM Message Service Client for .NET
distributed distributed MQ_Migration_Guide.pdf	IBM WebSphere MQ/IBM MQ 迁移指南
z/OS z/OS WMQ_zOS_Migration.pdf	IBM WebSphere MQ/IBM MQ for z/OS 迁移指南

注: PDF 文件必须位于同一文件夹中， PDF 文件之间的链接才能正常工作。

IBM MQ for z/OS Program Directory PDF 文件

z/OS

以下 IBM MQ for z/OS Program Directory 文档可作为 PDF 文件下载:

- IBM MQ for z/OS V8.0 程序目录 (GI13-3328-00)

- IBM MQ Advanced Message Security for z/OS V8.0 Program Directory (GI13-3329-00)
- IBM MQ Managed File Transfer for z/OS V8.0 Program Directory (GI13-3330-00)
- IBM MQ Advanced for z/OS V8.0 Program Directory (GI13-3331-00)
- IBM IBM WebSphere MQ for z/OS Value Unit Edition V8.0 程序目录 (GI13-3332-00)

术语表

本词汇表提供 IBM WebSphere MQ 软件及产品的术语和定义。

本词汇表中使用了以下交叉引用：

- 请参阅从非首选术语指引您参看首选术语或者从缩写指引您参看拼写完整的形式。
- 另请参阅将指引您查阅相关术语或对照术语。

第 87 页的『A』 第 90 页的『B』 第 90 页的『C』 第 94 页的『D』 第 96 页的『E』 第 97 页的『F』 第 98 页的『G』 第 99 页的『H』 第 100 页的『I』 第 101 页的『J』 第 102 页的『K』 第 102 页的『L』 第 103 页的『M』 第 107 页的『N』 第 108 页的『O』 第 109 页的『P』 第 111 页的『Q』 第 112 页的『R』 第 115 页的『S』 第 119 页的『T』 第 121 页的『U』 第 122 页的『V』 第 122 页的『W』 第 123 页的『X』

A

异常结束原因码 (abend reason code)
一个 4 字节的十六进制代码，它唯一地标识在 z/OS 操作系统上运行的程序存在的问题。

抽象类 (abstract class)
在面向对象程序设计中，指的是表示概念的类；从该类派生的类表示该概念的实现。 无法从抽象类构造对象；即，无法将抽象类实例化。 另请参阅父类 (parent class)。

抽象语法表示法 1 (Abstract Syntax Notation One, ASN.1)
这是定义信息数据的语法时应遵循的国际标准。 它定义了若干个简单数据类型，并指定引用这些类型以及指定这些类型的值时采用的表示法。 每当需要定义信息的抽象语法，而不以任何形式约束要传输的信息的编码方式时，即可应用 ASN.1 表示法。

访问控制
在计算机安全性中，指的是确保用户只能访问计算机系统中他们拥有权限的那些资源的过程。

访问控制表 (access control list，ACL)
计算机安全性中与对象相关联的列表，此列表标识了所有可以访问该对象的主体集及其访问权。

ACL
请参阅访问控制表（access control list，ACL）。

活动日志 (active log)
一个具有固定大小的数据集，恢复事件在发生时将记录在该数据集中。 当活动日志已满时，活动日志的内容将复制到归档日志中。

活动队列管理器实例 (active queue manager instance)
正在运行的多实例队列管理器的实例，该实例正在处理请求。 一个多实例队列管理器仅有一个活动实例。

适配器 (adapter)
这是一个中间软件组件，它使两个其他软件组件能够相互通信。

地址空间
可用于计算机程序或进程的地址范围。 地址空间可以引用物理存储器和/或虚拟存储器。 另请参阅联合地址空间 (allied address space) 和缓冲池 (buffer pool)。

管理包 (administration bag)
WebSphere MQ 管理接口（MQAI）中为了管理 WebSphere MQ 而创建的一种数据包，暗示能够更改数据项的顺序、创建列表和检查消息中的选择器。

管理主题对象 (administrative topic object)
一种允许您将特定的非缺省属性指定给主题的对象。

管理员命令 (administrator command)
用于管理 WebSphere MQ 对象（例如，队列、进程和名称列表）的命令。

高级消息队列协议 (Advanced Message Queuing Protocol, AMQP)
一种开放式源代码连线协议，用于接收、排队、路由和传递消息。

高级程序间通信 (Advanced Program-to-Program Communication，APPC)
SNA LU 6.2 协议的一种实现，它允许互连的系统通信和共享程序的处理。

亲缘关系
具有某种关系或互相依赖的对象之间的关联。

警报
一条消息或其他指示，用于指示满足一组指定条件的事件或即将发生的事件。

报警监视器 (alert monitor)
WebSphere MQ for z/OS 中 CICS 适配器的一个组件，它处理对 WebSphere MQ for z/OS 的连接请求引发的未安排的事件。

别名队列 (alias queue)
一个 WebSphere MQ 对象，其名称是定义到本地队列管理器的基本队列或主题的别名。 当应用程序或队列管理器使用别名队列时，将解析别名并对相关联的基本对象执行所请求的操作。 有关进一步的信息，请参阅别名队列 (alias queue) 主题。

别名队列对象 (alias queue object)
一种 WebSphere MQ 对象，其名称是定义到本地队列管理器的基本队列的别名。 当应用程序或队列管理器使用别名队列时，将解析别名并对相关联的基本队列执行所请求的操作。

联合地址空间 (allied address space)
连接到 WebSphere MQ for z/OS 的 z/OS 地址空间。

联合 (ally)
请参阅联合地址空间 (allied address space)。

备用用户权限 (alternate user authority)
一个用户标识提供其他用户标识以进行安全性检查的能力。 应用程序打开 WebSphere MQ 对象时，可以在 MQOPEN、MQPUT1 或 MQSUB 调用中提供队列管理器用于权限检查的用户标识，而不是提供与该应用程序相关联的用户标识。 有关进一步的信息，请参阅备用用户权限 (alternate user authority) 主题。

备用用户安全性 (alternate user security)
在 z/OS 中打开 WebSphere MQ 对象时，应用程序请求备用用户权限时执行的权限检查。

AMQP
请参阅高级消息队列协议 (Advanced Message Queuing Protocol)。

AMQP 通道
一种为 AMQP 1.0 兼容的应用程序提供一定程度支持的通道类型。 MQ Light 客户机或其他 AMQP 1.0 兼容的客户机可以连接到 IBM MQ AMQP 通道。

APAR
请参阅授权程序分析报告 (authorized program analysis report)。

APF
请参阅授权程序设施 (authorized program facility)。

API 交叉出口
用户编写的程序，在概念上类似于 API 出口。 它仅受 WebSphere MQ for z/OS 上的 CICS 应用程序支持。

API 出口 (API exit)
用户编写的程序，用于监视或修改 MQI 调用的函数。 对于应用程序发出的每个 MQI 调用，将在队列管理器开始处理该调用之前以及在队列管理器处理完该调用之后分别调用一次 API 出口。 API 出口可以检查和修改 MQI 调用中的任何参数。

APPC
请参阅高级程序间通信 (Advanced Program-to-Program Communication)。

应用程序定义的格式 (application-defined format)
消息中的应用程序数据，用户应用程序定义该数据的含义。 另请参阅内置格式 (built-in format)。

应用程序环境 (application environment)
包括软件和支持该软件的服务器或网络基础结构的环境。

应用程序级别安全性 (application level security)
在应用程序发出 MQI 调用时启动的安全服务。

应用程序日志 (application log)
在 Windows 系统中记录重要应用程序事件的日志。

应用程序队列 (application queue)
应用程序通过消息队列接口 (MQI) 用于消息传递的本地队列。 应用程序队列通常设置为触发队列。

归档日志 (archive log)
存储设备上的一个数据集，当活动日志达到其大小限制时，WebSphere MQ 将每个活动日志数据集的内容复制到此数据集。 另请参阅恢复日志 (recovery log)。

ARM
请参阅自动重新启动管理器（automatic restart manager）。

ASN.1
请参阅抽象语法表示法 1 (Abstract Syntax Notation One)。

非对称密钥密码术 (asymmetric key cryptography)
一个使用两种密钥的密码术系统：对每个用户公开的公用密钥和仅对消息接收方或发送方公开的专用密钥。 另请参阅对称密钥密码术 (symmetric key cryptography)。 有关进一步的信息，请参阅密码术 (cryptography) 主题。

异步使用 (asynchronous consumption)
使用一组 MQI 调用的过程，这些调用允许应用程序使用来自一组队列的消息。 通过使用应用程序所标识的代码单元并传递消息或表示消息的标记，将消息传递到应用程序。

异步消息传递 (asynchronous messaging)
一种在程序之间进行通信的方法，通过这种方法，程序可以将消息放在消息队列上，然后继续进行处理，而不必等待对消息的应答。 另请参阅同步消息传递 (synchronous messaging)。

异步放置
应用程序放置消息的操作，该操作不需要等待队列管理器的响应。

属性

1. 这是用于描述实体的实体特征或特性；例如，员工的电话号码就是员工的其中一个属性。 另请参阅实体 (entity)。
2. 面向对象程序设计中的对象或类的属性，它可以与其他任何属性明显区分开。 属性通常描述状态信息。

认证 (authentication)
一种安全服务，证明计算机系统的某个用户确实是它所声明的用户。 实现此服务的常用机制有密码和数字签名。

认证信息对象 (authentication information object)
一种对象，它使用 LDAP 服务器来提供检查证书撤销列表 (CRL) 所需的定义，以支持安全套接字层 (SSL) 安全性。

权限检查 (authority check)
请参阅授权检查 (authorization check)。

授权 (authorization)
向用户、系统或进程授予对于对象、资源或功能的完全或受限访问权的过程。

授权检查 (authorization check)
当用户或应用程序试图访问系统资源时执行的安全性检查；例如，当管理员试图发出命令以管理 WebSphere MQ 时或应用程序试图连接至队列管理器时。

授权文件 (authorization file)
为一个对象、一类对象或所有类的对象提供安全性定义的文件。

授权服务 (authorization service)
在 WebSphere MQ on UNIX 和 Linux 系统以及 WebSphere MQ for Windows 中，这是一项服务，用于针对与命令或调用关联的用户标识提供命令和 MQI 调用的权限检查。

授权程序分析报告 (authorized program analysis report，APAR)
用于修正 IBM 所提供程序的受支持发行版中某个缺陷的请求。

授权程序设施 (authorized program facility，APF)
在 z/OS 环境中，允许对授权使用受限功能的程序进行标识的一种设施。

自动重新启动管理器 (automatic restart manager，ARM)
一种 z/OS 恢复功能，在批处理作业和已启动任务或者其运行所在的系统意外终止后，此功能可以自动重新启动这些作业和任务。

B

回退 (backout)
一个撤销当前工作单元期间对资源作出的所有更改的操作。 另请参阅落实 (commit)。

包 (bag)
请参阅数据包 (data bag)。

内存栅 (bar)
一种 z/OS 内存限制，在 64 位系统中，该限制已设置为 2 GB 。 内存栅将 2 GB 地址以下的存储器与 2 GB 地址以上的存储器分隔开。 内存栅以上的区域用于数据；任何程序都不会在内存栅以上的区域运行。

基本映射支持 (basic mapping support，BMS)
CICS 与应用程序之间的一种接口，它格式化输入和输出显示数据并传递多页输出消息（不管各种终端使用何种控制字符）。

行为 (behavior)
在面向对象程序设计中，指的是方法中包含的功能。

BMS
请参阅基本映射支持 (basic mapping support)。

Booch 方法 (Booch methodology)
一个面向对象的方法，它帮助用户使用面向对象的范例来设计系统。

引导数据集 (BSDS)
一个 VSAM 数据集，它包含了已知的 WebSphere MQ for z/OS 所有活动和归档日志数据集的清单，还包含了一个所有近期 WebSphere MQ for z/OS 活动的回绕清单。 重新启动 WebSphere MQ for z/OS 子系统必需 BSDS。

browse
在消息排队中，这指的是复制消息而不将其从队列中除去。 另请参阅获取 (get) 和放置 (put)。

浏览光标 (browse cursor)
在消息排队中，指的是浏览队列时使用的指示器，用于标识序列中的下一条消息。

BSDS
请参阅引导数据集 (bootstrap data set)。

缓冲池
内存的一个区域，在处理期间，数据页将读入该区域并在其中进行修改和保存。 另请参阅地址空间 (address space)。

内置格式 (built-in format)
消息中的应用程序数据，队列管理器定义该数据的含义。 另请参阅应用程序定义的格式 (application-defined format)。

C

加利福尼亚州
请参阅认证中心 (certificate authority)。

CAF
请参阅客户机连接功能部件 (client attachment feature)。

回调 (callback)
消息使用者或事件处理程序例程。

CCDT
请参阅客户机通道定义表 (client channel definition table)。

CCF
请参阅通道控制功能 (channel control function)。

CCSID
请参阅编码字符集标识 (coded character set identifier)。

CDF
请参阅通道定义文件 (channel definition file)。

认证中心 (certificate authority, CA)
这是发放数字证书的可信第三方组织或公司。 认证中心通常将验证被授予唯一证书的个体的身份。 另请参阅安全套接字层 (Secure Sockets Layer)。 有关进一步的信息，请参阅认证中心 (certificate authority) 主题。

证书链 (certificate chain)
密码彼此相关的证书的层次结构，以个人证书开头，以位于链顶部的根证书结尾。

证书到期 (certificate expiration)
数字证书包含证书有效的日期范围。 在有效日期范围外，证书被认定为“已到期”。

证书请求 (certificate request, CR)
证书签名请求 (certificate signing request) 的同义词。

证书撤销列表 (CRL)
在到达预定的截止日期前被撤销的证书列表。 证书撤销列表由认证中心维护，并且在安全套接字层 (SSL)握手期间用于确保涉及到的证书尚未被撤销。 有关进一步的信息，请参阅证书管理 (certificate management) 主题。

证书签名请求 (certificate signing request，CSR)
某个组织发送至认证中心 (CA) 以获取证书的电子消息。 该请求包含一个公用密钥并且已使用专用密钥进行签名；CA 在使用其自己的专用密钥进行签名后返回证书。

证书库 (certificate store)
密钥库的 Windows 名称。 有关进一步的信息，请参阅个人证书库 (personal certificate store) 主题。

CF
请参阅耦合设施 (coupling facility)。

CFSTRUCT
一种 WebSphere MQ 对象，用于描述队列管理器使用耦合设施列表结构的情况

通道
一种 WebSphere MQ 对象，用于定义两个队列管理器之间的通信链路（消息通道）或者客户机与队列管理器之间的通信链路（MQI 通道）。 另请参阅消息通道 (message channel)、MQI 通道 (MQI channel) 和队列管理器 (queue manager)。

通道回调 (channel callback)
一种确保与正确的机器建立通道连接的机制。 在通道回调中，发送方通道使用发送方的定义回调原始请求者通道。

通道控制功能 (channel control function，CCF)
一个用于将消息从传输队列移至通信链路并从通信链路移至本地队列的程序，以及一个允许设置和控制通道的操作员面板界面。

通道定义文件 (channel definition file，CDF)
一个包含通信信道定义的文件，这些定义使传输队列与通信链路相关联。

通道事件 (channel event)
一种事件，它报告通道操作期间（如启动或停止通道实例时）检测到的条件。 将在通道两端的队列管理器中生成通道事件。

通道出口程序 (channel exit program)
用户编写的程序，将从消息通道代理程序 (MCA) 处理序列中有限数量的位置之一调用该程序。

通道启动程序 (channel initiator)
WebSphere MQ 分布式排队的一个组件，用于监视启动队列并在满足触发条件时启动发送方通道。

通道侦听器 (channel listener)
WebSphere MQ 分布式排队的一个组件，它监视网络中的启动请求，然后启动接收通道。

检查点 (checkpoint)
程序中的一个位置，将在该位置执行检查，或在该位置记录数据以允许在中断时重新启动该程序。

CI
请参阅控制区间 (control interval)。

CipherSpec
在完成认证后应用于 SSL 消息的加密算法和散列功能的组合。

密码套件 (cipher suite)
认证、密钥交换算法和安全套接字层 (SSL) 密码规范的组合，用于数据的安全交换。

密文 (ciphertext)
加密的数据。 在使用密钥将密文转换为明文（已解密）之前，密文是无法阅读的。 另请参阅明文 (cleartext)。

循环日志记录 (circular logging)
在 WebSphere MQ on UNIX 和 Linux 系统以及 WebSphere MQ for Windows 中，将所有重新启动数据保留在日志文件环中的过程。 另请参阅线性日志记录 (linear logging)。 有关进一步的信息，请参阅日志记录类型 (Types of logging) 主题。

CL
请参阅命令语言 (Command Language)。

class
面向对象设计或程序设计中的一个模型或模板，可用于创建具有公共定义和公共属性、操作和行为的对象。 对象是类的实例。

类层次结构 (class hierarchy)
共享单继承的各个类之间的关系。

类库 (class library)
在面向对象程序设计中，指的是预先编写的类或已编码的模板的集合，程序员在开发应用程序时可以指定和使用其中的任何类或模板。

明文 (cleartext)
以可读形式通过网络发送的一串字符。 可以将其编码以进行压缩，但可以轻松将其解码。 另请参阅密文 (ciphertext)。

客户机
一个运行时组件，它为本地用户应用程序提供对服务器上排队服务的访问。 应用程序使用的队列驻留在服务器上。 另请参阅 WebSphere MQ 完全管理的 .NET 客户机， WebSphere MQ Java 客户机， WebSphere MQ MQI 客户机。

客户机应用程序 (client application)
一个应用程序，在工作站上运行且链接至客户机，授予应用程序对服务器上排队服务的访问权。

客户机连接功能部件（client attachment feature, CAF）
支持将客户机连接至 z/OS 的一种选项。

客户机通道定义表 (client channel definition table，CCDT)
包含一个或多个客户机连接通道定义的文件。 有关进一步的信息，请参阅客户机通道定义表 (client channel definition table) 主题。

客户机连接通道类型 (client-connection channel type)
与 WebSphere MQ 客户机关联的 MQI 通道定义的类型。 另请参阅服务器连接通道类型 (server-connection channel type)。

CLUSRCVR
请参阅集群接收方通道 (cluster-receiver channel)。

CLUSSDR
请参阅集群发送方通道 (cluster-sender channel)。

集群
在 WebSphere MQ 中，由一台或多台计算机上的两个或更多队列管理器构成的组，这些队列管理器提供自动互连，并允许在它们之间公布队列和主题以进行负载均衡和实现冗余。

集群队列 (cluster queue)
一个本地队列，它由集群队列管理器主管并定义为要放置的消息的目标，该消息来自已连接至集群中的任何队列管理器的应用程序。 必须以本地方式连接所有将检索消息的应用程序。

集群队列管理器 (cluster queue manager)
隶属于集群的队列管理器。 一个队列管理器可以隶属于多个集群。

集群接收方通道 (cluster-receiver channel，CLUSRCVR)
一种通道，集群队列管理器可以在该通道上接收来自集群中的其他队列管理器的消息以及来自库队列管理器的集群信息。

集群发送方通道 (cluster-sender channel，CLUSSDR)
一种通道，集群队列管理器可以在该通道上将消息发送到集群中的其他队列管理器以及将集群信息发送到库队列管理器。

集群主题 (cluster topic)
在集群队列管理器中定义并可用于集群中的其他队列管理器的管理主题。

集群传输队列 (cluster transmission queue)
一种传输队列，它保存来自某个队列管理器并且将发送到同一集群中的另一个队列管理器的所有消息。该队列的名称为 SYSTEM.CLUSTER.TRANSMIT.QUEUE。

CMS 密钥数据库 (CMS key database)
CMS 密钥数据库是 Windows 系统， UNIX 系统， Linux 以及这些平台的客户机支持的数据库格式。 以“.kdb”结尾的文件为 CMS 格式。 “.kdb”文件包含证书和密钥。

编码字符集标识 (CCSID)
一个 16 位的数字，包括特定的一组编码方案标识、字符集标识、代码页标识和唯一地标识编码图形字符表示的其他信息。

共存
WebSphere MQ 的两个或更多不同版本在同一台计算机上工作的能力。

命令
用于启动操作或服务的语句。 命令包含命令名缩写及其参数，如果适用，那么还包含标志。

命令包 (command bag)
MQAI 中的一种包，它是为管理 WebSphere MQ 对象而创建的，但不能在消息中更改数据项的顺序或创建列表。

命令事件 (command event)
关于 MQSC 或 PCF 命令已成功运行的通知。

命令语言 (Command Language，CL)
WebSphere MQ for iSeries 中的一种语言，可用于在命令行中或通过编写 CL 程序来发出命令。

命令前缀（command prefix，CPF）

1. WebSphere MQ for z/OS 中的字符串，用于标识 WebSphere MQ for z/OS 命令指向的队列管理器以及是从哪个队列管理器接收 WebSphere MQ for z/OS 操作员消息。
2. 由 1 至 8 个字符组成的命令标识。 命令前缀将命令标识为属于应用程序或子系统，而不是属于 z/OS。

命令服务器 (command server)
从系统命令输入队列读取命令、验证命令并将有效命令传递给命令处理器的 WebSphere MQ 组件。

落实 (commit)
应用在当前恢复单元 (UR) 或工作单元 (UOW) 期间作出的所有更改。 在完成该操作之后，可以开始新的 UR 或 UOW。

公共名称 (common name, CN)
X.509 证书的专有名称 (DN) 属性中的组成部分，用于表示通常与证书所有者相关联的名称。 对于人员而言，CN 通常是其实际姓名。 对于 Web 服务器而言，CN 是服务器的标准主机名和域名。 对于 WebSphere MQ 而言，该字段无任何特定需求，但许多管理员都使用队列管理器名称。 另请参阅“专有名称”(Distinguished Name)

完成代码 (completion code)
一个返回码，指示消息队列接口 (MQI) 调用的结束情况。

机密性 (confidentiality)
保护敏感信息不受未授权访问的安全服务。 加密是实现此服务的公共机制。

配置事件 (configuration event)
关于对象属性的通知。 创建、更改或删除对象时将生成这些通知，还可以通过显式请求来生成这些通知。

连接亲缘关系 (connection affinity)
一种通道属性，用于指定客户机应用程序用于连接至队列管理器（如果存在多个可用连接）的客户机通道定义。

连接工厂
一组配置值，生成一些连接以使 Java EE 组件能够访问资源。 连接工厂提供从应用程序到企业信息系统 (EIS) 的随需应变连接，并且允许应用程序服务器将 EIS 列入分布式事务。

连接句柄 (connection handle)
程序访问与其连接的队列管理器所使用的标识或标记。

构造函数 (constructor)
在面向对象程序设计中，指的是用于初始化对象的特殊方法。

使用 (consume)
从队列中除去一条消息并将其内容返回到调用应用程序。

使用者
接收并处理消息的应用程序。 另请参阅消息使用者 (message consumer)。

上下文安全性 (context security)
在 z/OS 中，当应用程序打开队列并指定它将设置放入队列的消息中的上下文时，或将它接收到的来自消息的上下文传递给放入队列的消息时执行的权限检查。

控制命令 (control command)
在 UNIX 和 Linux 系统以及 WebSphere MQ for Windows 上的 WebSphere MQ 中，可以从操作系统命令行以交互方式输入命令。 这样的命令只需要安装有 WebSphere MQ 产品；它不需要特殊的实用程序或程序来运行它。

控制区间 (control interval，CI)
直接访问存储器的一个固定长度的区域，VSAM 在其中存储记录和创建分布式可用空间。 控制区间是 VSAM 发送到直接访问存储器或从直接访问存储器得到的信息单元。 控制区间始终包含整数个物理记录。

受控关闭 (controlled shutdown)
请参阅停顿关闭 (quiesced shutdown)。

相关标识 (correlation identifier)
消息中的一个字段，用于提供标识相关消息的方法。 例如，相关标识用于使请求消息与其对应的应答消息相匹配。

耦合设施 (coupling facility，CF)
一个特殊的逻辑分区，它在综合系统中提供较高速度的高速缓存、列表处理和锁定功能。

CPF
请参阅命令前缀 (command prefix)。

CRL
请参阅证书撤销列表 (certificate revocation list)。

跨系统耦合设施 (cross-system coupling facility，XCF)
z/OS 的一个组件，它提供功能来支持综合系统中运行的已授权程序之间的协作。

密码术 (cryptography)
通过将信息变换（加密）为无法理解的格式（称为密文）来对信息进行保护的一种方法。 只有拥有密钥的用户才能将消息破译（解密）为明文。

D

DAE
请参阅转储分析和消除 (dump analysis and elimination)。

守护程序
一个程序，以无人照管方式运行来执行连续的或周期性的功能，例如网络控制。

数据包 (data bag)
MQAI 在管理队列管理器中使用的对象属性的容器。 存在三种数据包类型：用户（针对用户数据）、管理（针对使用假定选项进行的管理）和命令（针对在不假定任何选项的情况下进行的管理）。

数据转换接口 (data-conversion interface，DCI)
一种 WebSphere MQ 接口，客户或供应商编写的用于在不同机器编码和 CCSID 之间转换应用程序数据的程序必须符合该接口。

数据转换服务 (data-conversion service)
一种服务，用于将应用程序数据转换为其他平台上的应用程序所需的字符集和编码。

数据报 (datagram)
异步消息传递的一种形式，其中，应用程序发送消息，但不需要响应。 另请参阅请求/应答 (request/reply)。

数据完整性 (data integrity)
一种安全服务，它检测是否对数据进行了未授权的修改，或者说篡改。 该服务仅检测是否修改了数据；如果修改了数据，那么它不会将数据恢复到其原始状态。

数据项 (data item)
在 MQAI 中，数据包内包含的一个整数或字符串项。 数据项可以是用户项或系统项。

DCE
请参阅分布式计算环境 (Distributed Computing Environment)。

DCE 主体 (DCE principal)
使用分布式计算环境的用户标识。

DCI
请参阅数据转换接口 (data-conversion interface)。

DCM
请参阅数字证书管理器 (Digital Certificate Manager)。

死信队列 (dead-letter queue，DLQ)
一种队列，队列管理器或应用程序将无法传递至其正确目标的消息发送至该队列。

死信队列处理程序
一个实用程序，它监视死信队列 (DLQ) 并根据用户编写的规则表处理该队列中的消息。 WebSphere MQ 提供了一个样本死信队列处理程序。

解密 (decryption)
将已加密为秘密格式的数据进行解码的过程。 解密需要密钥或密码。

缺省对象 (default object)
已定义了所有属性的对象（例如，队列）的定义。 如果用户定义某个对象，但未指定该对象的所有可能的属性，那么队列管理器使用缺省属性来代替任何未指定的属性。

延迟连接 (deferred connection)
当 CICS 子系统在启动之前尝试连接到 WebSphere MQ for z/OS 时激活的暂挂事件。

派生 (derivation)
在面向对象程序设计中，指的是对一个类进行改进或扩展以生成另一个类的过程。

目的地

1. 消息将发送至的端点，如队列或主题。
2. 在 JMS 中，用于指定发送和接收消息的位置和方式的一种对象。

Diffie-Hellman 密钥交换 (Diffie-Hellman key exchange)
一种公用密钥交换算法，用于通过不安全的通道安全地建立共享机密。

数字证书 (digital certificate)
一个电子文档，用于标识个体、系统、服务器、公司或其他某个实体，以及将公用密钥与该实体相关联。 数字证书由认证中心发布且由该中心进行数字签署。

数字证书管理器 (Digital Certificate Manager，DCM)
在 i5/OS 系统上，用于管理数字证书并将它们用于 iSeries 服务器上的安全应用程序中的方法。 数字证书管理器从认证中心 (CA) 或其他第三方机构请求数字证书并进行处理。

数字签名 (digital signature)
一些信息，使用专用密钥进行加密且追加至消息或对象，以确保接收方收到真实和完整的消息或对象。 数字签名证明该消息或对象已通过其所属或具有访问权的实体、专用密钥或共享密钥的对称密钥进行签名。

直接路由 (direct routing)
用于在发布/预订集群中路由发布的一种选项。 利用直接路由，集群中的每个队列管理器都会将发布从任一发布队列管理器直接发送至集群中带有匹配预订的任何其他队列管理器。

断开连接
中断应用程序与队列管理器之间的连接。

专有名称 (distinguished name，DN)
一组“名称/值”对（例如“CN=人员姓名”和“C=国家或地区”），用于唯一地标识数字证书中的实体。

分布式应用程序 (distributed application)
消息排队中的一组应用程序，其中每个应用程序都可以连接至不同的队列管理器，但它们共同组成一个应用程序。

分布式计算环境 (Distributed Computing Environment，DCE)
网络计算中的一组服务和工具，它支持在异构操作系统和网络之间创建、使用和维护分布式应用程序。

分布式发布/预订 (distributed publish/subscribe)
多队列管理器环境中执行的发布/预订消息传递。

分布式队列管理 (distributed queue management)
消息排队中，在分布式队列管理器之间对消息通道的设置和控制。

分布式排队 (distributed queuing)
将消息从一个队列管理器发送至另一个队列管理器。 接收队列管理器可位于同一机器或远程机器上。

分发列表
一组队列，可以通过单个语句将一条消息放置在这些队列中。

DLQ
请参阅死信队列 (dead-letter queue)。

DN
请参阅专有名称 (distinguished name)。

双记录 (dual logging)
记录 WebSphere MQ for z/OS 活动的一种方法，其中每个更改都将记录到两个数据集中，因此，如果一个数据集在需要重新启动而不可用时，可以使用另一个。 另请参阅单一记录 (single logging)。

双方式 (dual mode)
请参阅双记录 (dual logging)。

转储分析和消除 (dump analysis and elimination，DAE)
一项 z/OS 服务，它允许安装程序抑制因与先前写入的转储重复而不再需要的 SVC 转储和 ABEND SYSUDUMP 转储。

持久预订 (durable subscription)
当关闭从预订应用程序到队列管理器的连接时将保留的预订。 当预订应用程序断开连接时，持久预订仍保留在原位置并且将继续传送发布内容。 当该应用程序重新连接时，它可以通过指定唯一订阅名称来使用同一个订阅。 另请参阅非持久订阅 (nondurable subscription)。

动态队列 (dynamic queue)
程序打开模型队列对象时创建的本地队列。

E

窃听 (eavesdropping)
通信安全性的一种违规，在该过程中，信息仍保持完整无缺，但其私密性已被损害。 另请参阅冒名 (impersonation) 和篡改 (tampering)。

Eclipse
一项开放式源代码计划，旨在向独立软件供应商 (ISV) 和其他工具开发商提供标准平台以开发与插件兼容的应用程序开发工具。

封装 (encapsulation)
面向对象程序设计中的技术，用于对客户机程序隐藏对象、函数或类的内在详细信息。

加密 (encryption)
在计算机安全性中，指的是以某种方式将数据变换为无法理解的格式的过程，以便无法获得原始数据，或者只能使用解密过程才能获得原始数据。

入队 (enqueue)
将消息或项放入队列中。

实体 (entity)
对安全服务（如 RACF）定义的用户、组或资源。

环境变量 (environment variable)
一个变量，指定操作系统或其他程序的运行方式，或指定操作系统可识别的设备。

ESM
请参阅外部安全性管理器 (external security manager)。

ESTAE
请参阅扩展指定任务异常出口 (extended specify task abnormal exit)。

事件数据 (event data)
在事件消息中，指的是消息数据中包含有关事件信息（如队列管理器名称和引起事件的应用程序）的部分。 另请参阅事件头 (event header)。

事件头 (event header)
在事件消息中，指的是消息数据中标识事件原因码的事件类型的部分。 另请参阅事件数据 (event data)。

事件消息 (event message)
一种消息，其中包含了与 WebSphere MQ 系统网络中检测事件的起源相关的信息（例如，事件类别、导致事件出现的应用程序名称以及队列管理器统计信息）。

事件队列 (event queue)
队列管理器检测到事件后，将事件消息放入此队列。 每个类别的事件（队列管理器、性能、配置、检测或通道事件）都具有自己的事件队列。

事件查看器 (Event Viewer)
由 Windows 系统提供用于检查和管理日志文件的一个工具。

异常侦听器 (exception listener)
可以由应用程序注册的类的一个实例，并且可以为该实例调用 onException() 方法以将 JMS 异常以异步方式传递到该应用程序。

独占方法 (exclusive method)
在面向对象程序设计中，指的是不希望展示多态性的方法；每个方法都具有特定结果。

扩展指定任务异常出口 (extended specify task abnormal exit，ESTAE)
一种 z/OS 宏，它提供了恢复能力并提供了对用户指定的出口例程的控制，以进行处理、诊断异常结束现象或指定重试地址。

外部安全性管理器 (external security manager，ESM)
对用户和资源执行安全性检查的安全性产品。 例如，RACF 就是一个 ESM。

F

故障转移
一种自动操作，它在发生软件、硬件或网络中断时切换到冗余或备用系统或节点。

FAP
请参阅格式和协议 (Formats and Protocols)。

联邦信息处理标准 (Federal Information Processing Standard)
一个标准，由美国国家标准技术学会在国家标准和国际标准不存在或不足以满足美国政府的要求时制订。

FFDC
请参阅首次故障数据捕获 (first-failure data capture)。

FFST
请参阅首次故障支持技术 (First Failure Support Technology)。

FFST 文件（FFST file）
请参阅首次故障支持技术文件（First Failure Support Technology file）。

FIFO
请参阅先进先出 (first-in first-out)。

FIPS
请参阅联邦信息处理标准 (Federal Information Processing Standard)。

首次故障数据捕获 (first-failure data capture，FFDC)

1. FFST 体系结构的 i5/OS 实现，提供问题识别，诊断数据的选择性转储，症状字符串生成和问题日志条目。
2. 一种问题诊断辅助，它确定错误，收集并记录关于这些错误的信息，然后将控制权返回给受影响的运行时软件。

首次故障支持技术 (First Failure Support Technology, FFST)
一种 IBM 体系结构，它定义了通过防御性编程技术检测错误的单一方法。 这些技术提供调试软件问题所必需的主动的（直到被要求才为被动的）问题识别和诊断输出的描述。

首次故障支持技术文件（First Failure Support Technology file，FFST file）
一个包含信息的文件，用于检测和诊断软件问题。 在 WebSphere MQ 中，FFST 文件的文件类型是 FDC。

先进先出 (first-in first-out，FIFO)
一种排队技术，在这种技术中，要检索的下一项是队列中存在时间最长的项。

强制关闭 (forced shutdown)
CICS 适配器的一种关闭方式，适配器不管当前任何活动任务的状态如何，都立即从 WebSphere MQ for z/OS 断开连接。 另请参阅停顿关闭 (quiesced shutdown)。

格式
在消息排队中，这指的是用于标识消息中应用程序数据性质的术语。

格式和协议 (Formats and Protocols，FAP)
消息排队中的一个定义，它定义队列管理器如何互相通信以及客户机如何与服务器队列管理器通信。

框架 (Framework)
WebSphere MQ 中一系列编程接口，它们允许客户或供应商编写程序以扩展或替换 WebSphere MQ 产品中提供的某些功能。 这些接口如下：数据转换接口 (DCI)、消息通道接口 (MCI)、名称服务接口 (NSI)、安全性启用接口 (SEI) 和触发器监视器接口 (TMI)。

朋友类 (friend class)
一种类，该类中的所有成员函数都被授予了对另一个类的私有成员和被保护成员的访问权。 它在另一个类的声明中进行命名并使用关键字 friend 作为该类的前缀。

FRR
请参阅功能恢复例程 (functional recovery routine)。

完整存储库 (full repository)
关于集群中每个队列管理器的信息的完整集合。 此信息集合称为存储库，有时称为完整存储库，通常由集群中的两个队列管理器进行保存。 另请参阅部分存储库 (partial repository)。

函数 (function)
一组指定的语句组，可被调用和求值并可对调用语句返回一个值。

功能恢复例程 (functional recovery routine，FRR)
一个 z/OS 恢复和终止管理器，它使恢复例程可以获得在程序中断事件中的控制。

G

网关队列管理器 (gateway queue manager)
一个集群队列管理器，用于将来自应用程序的消息路由至集群中的其他队列管理器。

通用跟踪设施 (generalized trace facility，GTF)
一个 z/OS 服务程序，它记录重大的系统事件，如 I/O 中断、SVC 中断、程序中断和外部中断。

类属安全性服务 API (Generic Security Services API)
请参阅类属安全性服务应用程序编程接口 (Generic Security Services application programming interface)。

类属安全性服务应用程序编程接口（类属安全性服务 API，GSS API）(Generic Security Services application programming interface，Generic Security Services API, GSS API)
用于访问安全服务的公共应用程序编程接口 (API)。

get
在消息排队中，这指的是使用 MQGET 调用从队列中除去消息并将其内容返回到调用应用程序。 另请参阅浏览 (browse) 和放置 (put)。

全局定义的对象 (globally defined object)
z/OS 中的一个对象，它的定义存储在共享库中。 该对象可用于队列共享组中的所有队列管理器。 另请参阅本地定义的对象 (locally defined object)。

全局跟踪 (global trace)
一个 WebSphere MQ for z/OS 跟踪选项，其中的跟踪数据来自整个 WebSphere MQ for z/OS 子系统。

全局事务 (global transaction)
可恢复的工作单元，由分布式事务环境中的一个或多个资源管理器执行并由外部事务管理器进行协调。

GSS API
请参阅类属安全性服务应用程序编程接口 (Generic Security Services application programming interface)。

GTF
请参阅通用跟踪设施 (generalized trace facility)。

H

握手
安全套接字层会话开始时的消息交换过程，它允许客户机使用公用密钥技术来认证服务器（对于服务器认证客户机的情况，该过程是可选的），然后允许客户机和服务器协作创建用于加密、解密和检测篡改的对称密钥。

固化消息 (hardened message)
一条消息，该消息已写入到辅助（磁盘）存储器，以便系统发生故障时不会丢失该消息。

头
请参阅消息头 (message header)。

脉动信号 (heartbeat)
一种信号，由一个实体向另一个实体发送，以传达发送信号的实体仍处于活动状态这一情况。

脉动信号流量 (heartbeat flow)
一个脉冲，当没有要发送的消息时，会将该脉冲从进行发送的消息通道代理程序 (MCA) 传递到进行接收的 MCA。 该脉冲将取消对进行接收的 MCA 的阻塞，否则该 MCA 将一直处于等待状态，直到一条消息到达或断开连接时间间隔到期为止。

脉动信号间隔 (heartbeat interval)
脉动信号流量之间将经历的时间（秒）。

层次结构
在发布/预订消息传递拓扑中，这指的是连接至父队列管理器的本地队列管理器。

HTTP
请参阅超文本传输协议 (Hypertext Transfer Protocol)。

超文本传输协议 (HTTP)
一种因特网协议，用于在 Web 上传输并显示超文本和 XML 文档。

I

标识
允许唯一地标识计算机的每个用户的安全服务。 实现此服务的公共机制是使用户标识与每个用户相关联。

标识上下文 (identity context)
用于标识应用程序用户的信息，该应用程序第一个将消息放置到队列中。

IFCID
请参阅检测设施组件标识 (instrumentation facility component identifier)。

ILE
请参阅集成语言环境 (Integrated Language Environment)。

立即关闭 (immediate shutdown)
WebSphere MQ 中不等待应用程序断开连接，就关闭队列管理器的一种关闭方式。 允许完成当前消息队列接口 (MQI) 调用，但在请求了立即关闭之后，新的 MQI 调用将失败。 另请参阅抢先关闭 (preemptive shutdown) 和停顿关闭 (quiesced shutdown)。

冒名 (impersonation)
通信安全性的一种违规过程，在该过程中，信息将传递到假冒预期接收方的人，或者某人假冒其他人发送信息。 另请参阅窃听 (eavesdropping) 和篡改 (tampering)。

入站通道 (inbound channel)
接收来自另一个队列管理器的消息的通道。

内置格式 (in-built format)
请参阅内置格式 (built-in format)。

索引 (index)
WebSphere MQ 管理接口（MQAI）中引用数据项的一种方式。

不确定的恢复单元 (indoubt unit of recovery)
恢复单元的状态，为该恢复单元请求了同步点，但尚未得到确认。

正在进行中 (inflight)
资源或恢复单元的状态，该资源或恢复单元尚未完成落实过程的准备阶段。

继承
一种面向对象程序设计技术，在其中可将现有类用作创建其他类的基础。 通过继承，更多的特定元素可包括更多常规元素的结构和行为。

初始化输入数据集 (initialization input data set)
WebSphere MQ for z/OS 在启动时使用的数据集。

启动队列
一个本地队列，队列管理器将触发器消息放置在该队列中。

起始符、启动程序 (initiator)
分布式排队中，在另一个系统上请求网络连接的一个程序。 另请参阅响应程序 (responder)。

输入参数 (input parameter)
MQI 调用的一个参数，用于提供信息。

插入顺序 (insertion order)
WebSphere MQ 管理接口（MQAI）中数据项放入数据包的顺序。

可安装服务 (installable service)
在 WebSphere MQ on UNIX 和 Linux 系统以及 WebSphere MQ for Windows 中，作为独立组件提供的其他功能。 每个组件的安装是可选的：可以改为使用内部组件或第三方组件。

实例 (instance)
属于某个类的对象的一次特定出现。 另请参阅对象 (object)。

实例数据 (instance data)
在面向对象程序设计中，指的是与对象相关联的状态信息。

中间证书
非根证书的签署者证书。 它由可信根专门发出，以发出终端实体服务器证书。 其结果是一个证书链，它以可信根 CA 开头，经历多个中间证书，最后以向组织发出的 SSL 证书结尾。

检测事件 (instrumentation event)
监视 WebSphere MQ 系统网络中的队列管理器资源定义、性能情况和通道情况的一种方式。

检测设施组件标识 (instrumentation facility component identifier，IFCID)
在 DB2 for z/OS 中，用于命名并标识事件跟踪记录的值。 作为 START TRACE 和 MODIFY TRACE 命令中的参数，它指定要跟踪相应的事件。

集成语言环境（Integrated Language Environment，ILE）
一组构造和接口，它为所有符合 ILE 的高级语言提供了公共运行时环境和运行时可绑定应用程序编程接口 (API)。

交互问题控制系统 (Interactive Problem Control System，IPCS)
MVS 和 z/OS 的一个组件，它允许联机问题管理、交互式问题诊断、驻留磁盘的异常结束转储的联机调试、问题跟踪和问题报告。

交互式系统生产率设施 (Interactive System Productivity Facility，ISPF)
作为全屏幕编辑器和对话管理器的 IBM 许可程序。 它用于编写应用程序，提供了一种在应用程序员和终端用户之间生成标准屏幕面板和交互式对话框的方法。

接口
面向对象程序设计中的一种抽象行为模型；一组函数或方法。

因特网协议 (Internet Protocol, IP)
在网络或互连网络中路由数据的协议。 此协议充当较高协议层与物理网络的媒介。 另请参阅传输控制协议 (Transmission Control Protocol)。

进程间通信 (interprocess communication，IPC)
程序相互之间发送消息所使用的进程。 套接字、信标、信号和内部消息队列是进程间通信的常见方法。 另请参阅客户机 (client)。

系统间通信 (intersystem communication，ISC)
一项 CICS 设施，它为来自其他计算机系统的通信提供入站和出站支持。

IP
请参阅因特网协议 (Internet Protocol)。

IPC
请参阅进程间通信 (interprocess communication)。

IPCS
请参阅交互问题控制系统 (Interactive Problem Control System)。

ISC
请参阅系统间通信 (intersystem communication)。

ISPF
请参阅交互式系统生产率设施 (Interactive System Productivity Facility)。

J

JAAS
请参阅 Java 认证和授权服务 (Java Authentication and Authorization Service)。

Java 认证和授权服务 (JAAS)
Java EE 技术中的一个标准 API，用于执行基于安全性的操作。 通过 JAAS，在允许应用程序与底层技术保持独立的同时，服务可以对用户进行认证和授权。

Java 消息服务 (Java Message Service, JMS)
一个应用程序编程接口，提供用于处理消息的 Java 语言函数。 另请参阅消息队列接口 (Message Queue Interface)。

Java 运行时环境 (Java Runtime Environment, JRE)
Java Developer Kit 的一个子集，包含那些构成标准 Java 平台的核心可执行程序和文件。 JRE 包含 Java 虚拟机 (JVM)、核心类和支持文件。

JMS
请参阅 Java 消息服务 (Java Message Service)。

JMSAdmin
一个管理工具，它使管理员能够定义 JMS 对象的属性并能够将这些属性存储在 JNDI 名称空间中

日志 (journal)
OS/400 的一个功能部件，由 WebSphere MQ for iSeries 用于控制对本地对象的更新。 每个队列管理器库都包含该队列管理器的一个日志。

JRE
请参阅 Java 运行时环境 (Java runtime environment)。

K

保持活动 (keepalive)
按预定义的时间间隔通过网络发送小包，以确定套接字是否仍正常工作的一种 TCP/IP 机制。

Kerberos
一个基于对称密钥密码术的网络认证协议。 Kerberos 对登录网络的每个用户都分配一个称为凭单的唯一密钥。 该凭单嵌入在通过网络发送的消息中。 消息的接收方使用该凭单对发送方进行认证。

密钥认证 (key authentication)
请参阅认证 (authentication)。

密钥库 (key repository)
用于存储数字证书及其相关联的专用密钥的存储器。

密钥环 (key ring)
在计算机安全性中，这指的是一个包含公用密钥、专用密钥、可信根和证书的文件。

密钥库 (keystore)
在安全性中，这指的是一个文件或硬件加密卡，其中存储了进行认证和加密所需的标识和专用密钥。 一些密钥库还包含可信密钥或公用密钥。 另请参阅信任库 (trust store)。

L

最后的消息 (last will and testament)
由客户机向监视器注册并在该客户机意外结束时由该监视器使用的某个对象。

LDAP
请参阅轻量级目录访问协议 (Lightweight Directory Access Protocol)。

轻量级目录访问协议 (LDAP)
一种开放式协议，使用 TCP/IP 来提供对支持 X.500 模型的目录的访问权，而且不会引发更为复杂的 X.500 目录访问协议 (DAP)的资源需求。 例如，LDAP 可用于在因特网或内部网目录中查找人员、组织和其他资源。

线性日志记录 (linear logging)
在 UNIX 和 Linux 系统上的 WebSphere MQ 以及 WebSphere MQ for Windows 中，将重新启动数据保留在一系列文件中的过程。 必要时会将新文件添加到该序列。 不会复用写入了数据的空间。 另请参阅循环日志记录 (circular logging)。 有关进一步的信息，请参阅日志记录类型 (Types of logging) 主题。

链路级别安全性 (link level security)
由消息通道代理程序 (MCA)、通信子系统或这两者相互结合来直接或间接启动的安全服务。

侦听器 (listener)
一个程序，检测入局请求并启动关联的通道。

远程队列的本地定义 (local definition of a remote queue)
属于本地队列管理器的一个 WebSphere MQ 对象，它定义了由另一队列管理器所拥有的队列的属性。 另外，它还用于队列管理器别名判别和应答队列别名判别。

locale
该设置用于标识语言或地理位置以及确定格式约定，例如排序、大小写转换、字符分类、消息语言、日期和时间表示方法以及数字表示方法。

本地定义的对象 (locally defined object)
z/OS 中的对象，它的定义存储在零页集上。 该定义只能由定义它的队列管理器访问。 另请参阅全局定义的对象 (globally defined object)。

本地队列 (local queue)
属于本地队列管理器的队列。 本地队列可以包含一列正在等待处理的消息。 另请参阅远程队列 (remote queue)。

本地队列管理器 (local queue manager)
程序已连接并且为程序提供消息排队服务的队列管理器。 另请参阅远程队列管理器 (remote queue manager)。

日志
WebSphere MQ 中的一个文件，它记录了队列管理器在接收、发送和传递消息时完成的工作，以便在队列管理器发生故障时进行恢复。

日志控制文件 (log control file)
在 UNIX 和 Linux 系统上的 WebSphere MQ 以及 WebSphere MQ for Windows 中，包含监视日志文件使用情况所需的信息 (例如，其大小和位置以及下一个可用文件的名称) 的文件。

日志文件
在 UNIX 和 Linux 系统上的 WebSphere MQ 以及 WebSphere MQ for Windows 中，记录队列管理器控制的数据的所有重要更改的文件。 如果主日志文件已满，WebSphere MQ 将分配辅助日志文件。

逻辑单元 (logical unit，LU)
一个访问点，用户或应用程序通过它访问 SNA 网络以便与另一个用户或应用程序通信。

逻辑单元 6.2 (logical unit 6.2，LU 6.2)
一种 SNA 逻辑单元，它支持在分布式处理环境中的程序之间进行常规通信。

逻辑工作单元标识 (logical unit of work identifier，LUWID)
唯一地标识网络中的某个线程的名称。 此名称由标准逻辑单元网络名、逻辑工作单元实例号和逻辑工作单元序号组成。

日志记录 (log record)
被视为日志文件中的一个单元的数据集。

日志记录序号 (log record sequence number，LRSN)
日志记录的唯一标识，它与数据共享成员相关联。 DB2 for z/OS 使用 LRSN 在数据共享环境中进行恢复。

LRSN
请参阅日志记录序号 (log record sequence number)。

LU
请参阅逻辑单元 (logical unit)。

LU 6.2
请参阅逻辑单元 6.2 (logical unit 6.2)。

LU 6.2 对话 (LU 6.2 conversation)
在 SNA 中，指的是两个事务程序之间通过 LU 6.2 会话建立的逻辑连接，以便这两个事务程序能够互相通信。

LU 6.2 对话级别安全性 (LU 6.2 conversation level security)
在 SNA 中，指的是对话级别安全性协议，它使伙伴事务程序能够认证启动对话的事务程序。

LU 6.2 会话 (LU 6.2 session)
在 SNA 中，指的是两个类型为 6.2 的逻辑单元之间的会话。

LU 名
VTAM 用来表示网络中的节点的名称。

LUWID
请参阅逻辑工作单元标识 (logical unit of work identifier)。

M

受管目标 (managed destination)
一个由队列管理器为选择使用受管预订的应用程序提供的队列，作为所发布消息将发往的目标。 另请参阅受管节预订 (managed subscription)。

受管句柄 (managed handle)
一个标识，当指定了队列管理器来管理已发送到预订的消息的存储时，MQSUB 调用将返回该标识。

受管预订 (managed subscription)
一种预订，因为应用程序不需要使用特定队列，所以队列管理器会为该预订创建订户队列来接收发布内容。 另请参阅受管目标 (managed destination)。

编组 (marshalling)
请参阅序列化 (serialization)。

MCA
请参阅消息通道代理程序 (message channel agent)。

MCI
请参阅消息通道接口 (message channel interface)。

介质映像 (media image)
在 UNIX 和 Linux 系统以及 WebSphere MQ for Windows 上的 WebSphere MQ 中，包含对象映像的日志记录序列。 可以根据此映像重新创建对象。

message

1. 在系统编程中，这指的是为终端操作员或系统管理员准备的信息。
2. 从一个应用程序传递至另一个应用程序的字节的字符串。 消息通常包含一个消息头（用于消息路由和识别）和一个有效内容（包含要发送的应用程序数据）。 该数据的格式与发送和接收应用程序均兼容。

消息亲缘关系 (message affinity)
在两个应用程序之间交换的会话式消息之间的关系，其中，这些消息必须由特定队列管理器或以特定顺序进行处理。

消息通道 (message channel)
在分布式消息排队中，这指的是将消息从一个队列管理器移至另一个队列管理器的机制。 消息通道包括两个消息通道代理程序（一端的一个发送方和另一端的一个接收方）和一条通信链路。 另请参阅通道 (channel)。

消息通道代理程序 (message channel agent，MCA)
一个程序，它将准备好的消息从传输队列传输至通信链路，或从通信链路传输至目标队列。 另请参阅消息队列接口 (Message Queue Interface)。

消息通道接口 (message channel interface，MCI)
一种 WebSphere MQ 接口，由客户或供应商编写的在 WebSphere MQ 队列管理器和另一消息传递系统之间传送消息的程序必须符合该接口。 另请参阅消息队列接口 (Message Queue Interface)。

消息使用者 (message consumer)

1. JMS 中在会话内创建的一个对象，用于从目标接收消息。
2. 获取并处理消息的一个程序、函数或组织。 另请参阅使用者 (consumer)。

消息上下文 (message context)
关于消息发起方的信息，它保存在消息描述符的字段中。 存在两种类别的上下文信息：身份上下文和起源上下文。

消息描述符 (message descriptor)
控制信息，它描述作为 WebSphere MQ 消息一部分的消息格式和表示。 消息描述符的格式由 MQMD 结构进行定义。

消息出口 (message exit)
一种通道出口程序，用于修改消息的内容。 消息出口通常成对地进行工作，通道每一端各一个。 在通道的发送端，将在消息通道代理程序 (MCA) 从传输队列获取消息后调用消息出口。 在通道的接收端，将在消息通道代理程序 (MCA) 将消息放置在其目标队列之前调用消息出口。

消息流控制 (message flow control)
一个分布式队列管理任务，它涉及设置和维护队列管理器之间的消息路由。

消息格式化服务 (Message Format Service，MFS)
一种 IMS 编辑工具，允许应用程序处理简单的逻辑消息而不是处理与设备相关的数据，从而简化应用程序开发过程。

消息组 (message group)
相关消息的逻辑组。 关系由放置这些消息的应用程序进行定义，并确保在生产者和使用者支持该分组时按放置顺序检索这些消息。

消息句柄 (message handle)
对消息的引用。 可使用句柄来获取对消息属性的访问权。

消息头 (message header)
消息的一部分，它包含控制信息，例如，唯一的消息标识、消息的发送方和接收方、消息优先级以及消息类型。

消息输入描述符 (message input descriptor，MID)
一种消息格式化服务 (MFS) 控制块，描述提供给应用程序的数据格式。 另请参阅消息输出描述符 (message output descriptor)。

消息侦听器 (message listener)
一个充当异步消息使用者的对象。

消息输出描述符 (message output descriptor，MOD)
一种消息格式化服务 (MFS) 控制块，描述应用程序产生的输出数据的格式。 另请参阅消息输入描述符 (message input descriptor)。

消息优先级 (message priority)
WebSphere MQ 中消息的一个属性，它会影响从队列中检索消息的顺序以及是否生成触发器事件。

消息生产者 (message producer)
在 JMS 中，这指的是由会话创建且用于将消息发送到目标的对象。 另请参阅生产者 (producer)。

消息属性 (message property)
与消息相关联的数据，采用名称/值对的格式。 消息属性可以用作消息选择器，以对发布内容进行过滤或从队列中有选择地获取消息。 可以使用消息属性来包含业务数据或关于处理过程的状态信息，而不必改变消息体。

消息队列接口 (Message Queue Interface，MQI)
由 WebSphere MQ 队列管理器提供的编程接口。 该编程接口允许应用程序访问消息排队服务。 另请参阅 Java 消息服务 (Java Message Service)， 消息通道代理 (message channel agent)， 消息通道接口 (message channel interface)。

消息队列管理 (message queue management，MQM)
WebSphere MQ for HP NonStop Server 中的一种工具，它允许访问 PCF 命令格式并控制命令以管理队列管理器、队列和通道。

消息排队 (message queuing)
一种编程技术，应用程序中的每个程序通过使用该技术在队列中放置消息来与其他程序通信。

消息重试 (message-retry)
一个可用于无法放置消息的 MCA 的选项。 MCA 可以等待预定义的时间量，然后尝试再次放置该消息。

消息段 (message segment)
太大而导致应用程序或队列管理器无法处理的消息的许多段之一。

消息选择器 (message selector)
在应用程序编程中，这是一个变长字符串，由应用程序用于仅注册其属性满足选择字符串所表示的结构化查询语言 (SQL) 查询的消息。 消息选择器的语法基于 SQL92 条件表达式语法的子集。

消息顺序编号 (message sequence numbering)
一种编程技术，在通过通信链路进行传输期间，将使用该技术给消息指定唯一编号。 该编号使接收进程能够检查是否已接收到所有消息，将它们按照原来的顺序放入队列中，并废弃重复的消息。

消息标记 (message token)
活动队列管理器中消息的唯一标识。

方法 (method)
在面向对象的设计或编程中，实施由操作指定的行为的软件。

MFS
请参阅消息格式化服务 (Message Format Service)。

MGAS
请参阅全局地址空间为主 (mostly global address space)。

Microsoft Cluster Server（MSCS）
一种通过将运行 Windows 的计算机分组为 MSCS 集群来提供高可用性的技术。 如果集群中的某个计算机突然遇到一系列问题中的任何一个问题，那么 MSCS 以有序方式关闭已中断的应用程序、将其状态数据转移到集群中的另一个计算机并在那里重新启动该应用程序。 有关进一步的信息，请参阅支持 Microsoft 集群服务 (MSCS) 主题。

Microsoft Transaction Server（MTS）
帮助 Windows 用户在中间层服务器上运行业务逻辑应用程序的一个设施。 MTS 将工作分为几个活动，这些活动是业务逻辑中独立的短块。

MID
请参阅消息输入描述符 (message input descriptor)。

MOD
请参阅消息输出描述符 (message output descriptor)。

模型队列对象 (model queue object)
一组队列属性，这些属性在程序创建动态队列时充当模板。

全局地址空间为主 (mostly global address space，MGAS)
一个灵活的虚拟地址空间模型，用在诸如 HP-UX 的系统中，它为共享应用程序保留了大部分地址空间。该模型可以增强共享大量数据的进程的性能。 另请参阅专用地址空间为主 (mostly private address space)。

专用地址空间为主 (mostly private address space，MPAS)
一个灵活的虚拟地址空间模型，用在诸如 HP-UX 的系统中，它可以将较大的地址空间块分配给进程。该模型可以增强需要大量数据空间的进程的性能。 另请参阅全局地址空间为主 (mostly global address space)。

MPAS
请参阅专用地址空间为主 (mostly private address space)。

MQAI
请参阅 WebSphere MQ 管理接口（WebSphere MQ Administration Interface）。

MQI
请参阅消息队列接口 (Message Queue Interface)。

MQI 通道
WebSphere MQ 客户机和服务器系统上的队列管理器之间的连接。 MQI 通道仅传输 MQI 调用并以双向方式进行响应。 另请参阅通道 (channel)。

MQM
请参阅消息队列管理 (message queue management)。

MQSC
请参阅 WebSphere MQ 脚本命令。

MQSeries
WebSphere MQ 以前的名称。

MQ Telemetry Transport (MQTT)
基于 TCP/IP 的开放式轻量级发布/预订消息传递协议，用于连接大量设备，如伺服系统、传动结构、智能手机、车辆、住宅、医疗、远程传感器和控制设备。 MQTT 旨在用于受限的设备以及带宽低、等待时间长或不可靠的网络。 该协议最大限度减少了网络带宽和设备资源需求，同时确保一定程度的可靠性和有保障的交付。 它越来越多地用于已连接设备的 机器到机器 (M2M) 或 物联网 世界，并且用于带宽和电池电量处于溢价的移动应用程序。

MQTT
请参阅 MQ Telemetry Transport。

MQTT 客户机 (MQTT client)
MQTT 客户机应用程序连接到支持 MQTT 的服务器，如 WebSphere MQ Telemetry 通道。 您可以编写自己的客户机以使用已发布的协议，或者下载免费的 IBM Messaging Telemetry Clients SupportPac。 典型的客户机负责从遥测设备中收集信息并将这些信息发布至服务器。 它还可预订主题、接收消息并使用该信息来控制遥测设备。

设备的 MQTT 守护程序
设备的 MQTT 守护程序是一种高级 MQTT V3 客户机。 它是为嵌入式系统设计的资源占用率非常小的 MQTT 服务器。 其主要用途是存储和转发来自遥测设备及其他 MQTT 客户机的消息，包括设备的其他 MQTT 守护程序。

注: 设备的守护程序不再可用。 有关备用解决方案，请参阅 eclipse.org “Mosquitto” 项目 (https://eclipse.org/mosquitto)。

MQTT 服务器
一种支持 MQ Telemetry Transport 协议的消息传递服务器。 它使 MQTT 客户机支持的移动应用程序和设备能够交换消息。 通常，它允许多个 MQTT 客户机同时连接到自己，并提供主数据中心，用于将消息分发至 MQTT 客户机。 MQTT 服务器可从 IBM 及其他公司获取。 IBM MessageSight、WebSphere MQ Telemetry 和设备的 MQTT 守护程序均为 IBM 提供的 MQTT 服务器。

MQXR 服务
请参阅遥测服务 (telemetry service)。

MSCS
请参阅 Microsoft Cluster Server。 有关进一步的信息，请参阅支持 Microsoft 集群服务 (MSCS) 主题。

MTS
请参阅 Microsoft Transaction Server。

多中继 (multi-hop)
在源队列管理器与目标队列管理器之间没有直接通信链路时经过一个或多个中间队列管理器。

多实例队列管理器 (multi-instance queue manager)
配置为与其他队列管理器实例共用队列管理器数据的队列管理器。 正在运行的多实例队列管理器的一个实例处于活动状态，其他实例处于备用就绪状态以从活动实例进行接管。 另请参阅队列管理器 (queue manager) 和单实例队列管理器 (single instance queue manager)。

N

名称列表 (namelist)
包含对象名称（例如，队列名称）列表的一种 WebSphere MQ 对象。

名称服务
在 WebSphere MQ on UNIX 和 Linux 系统以及 WebSphere MQ for Windows 中，用于确定哪个队列管理器拥有指定队列的设施。

名称服务接口
一种 WebSphere MQ 接口，客户或供应商编写的用于解决队列名称所有权的程序必须符合该接口。

名称变换 (name transformation)
在 UNIX 和 Linux 系统以及 WebSphere MQ for Windows 上的 WebSphere MQ 中，这是一个内部进程，用于更改队列管理器名称，使其对所使用的系统唯一且有效。 在外部看来，队列管理器名称保持不变。

嵌套包 (nested bag)
WebSphere MQ 管理接口（MQAI）中插入到另一数据包中的系统包。

嵌套
在 WebSphere MQ 管理接口（MQAI）中，对 WebSphere MQ 返回的信息的组合方式。

网络基本输入/输出系统 (NetBIOS，Network Basic Input/Output System)
在局域网中使用的与网络和个人计算机的标准接口，以提供消息、打印服务器和文件服务器功能。 使用 NetBIOS 的应用程序不必处理 LAN 数据链路控制 (DLC) 协议的详细信息。

网络基本输入/输出系统 (Network Basic Input/Output System)
请参阅 NetBIOS。

新技术文件系统 (New Technology File System，NTFS)
Windows 操作环境中的某个本机文件系统。

节点
在 Microsoft Cluster Server（MSCS）中，集群中的每台计算机。

非持久预订 (nondurable subscription)
仅当预订应用程序到队列管理器的连接未断开时才存在的预订。 当预订应用程序有意地断开或由于连接中断而断开与队列管理器的连接时，将除去此预订。 另请参阅持久订阅 (durable subscription)。

非持久消息 (nonpersistent message)
在重新启动队列管理器后不再存在的消息。 另请参阅持久消息 (persistent message)。

NSI
请参阅名称服务接口 (name service interface)。

NTFS
请参阅新技术文件系统 (New Technology File System)。

NUL
请参阅空字符 (null character)。

空字符 (null character，NUL)
一个值为 X'00' 的控制字符，它表示显示或打印的字符不存在。

O

OAM
请参阅对象权限管理器 (object authority manager)。

对象 (object)

1. 在 WebSphere MQ 中，是指一个队列管理器、队列、进程定义、通道、名称列表、认证信息对象、管理主题对象、侦听器、服务对象或者（仅限于 z/OS 上）一个 CF 结构对象或存储类。
2. 在面向对象的设计或编程中，是指组成数据的类的具体实现（实例）以及与该数据相关的操作。 对象包含由类定义的实例数据，但是该类拥有与该数据相关联的操作。

对象权限管理器 (object authority manager，OAM)
在 UNIX 和 Linux 系统上的 WebSphere MQ 中， WebSphere MQ for IBM i 和 WebSphere MQ for Windows 是用于命令和对象管理的缺省授权服务。 OAM 只能替换为客户提供的安全服务，或者只能与该安全服务组合在一起运行。

对象描述符 (object descriptor)
标识特定的 WebSphere MQ 对象的数据结构。 对象的名称和对象类型包含在该描述符中。

对象句柄 (object handle)
程序访问一起工作的 WebSphere MQ 对象所使用的标识或标记。

面向对象程序设计 (object-oriented programming)
一种基于数据抽象和继承的概念的编程方法。 与过程化编程技术不同，面向对象程序设计关注的不是如何完成某个事项，而是关注问题所涉及的数据对象以及如何操纵这些数据对象。

OCSP
请参阅在线证书状态协议 (Online Certificate Status Protocol)。

卸载 (offloading)
在 WebSphere MQ for z/OS 中的一个自动进程，在此进程中，队列管理器的活动日志将转移到它的归档日志中。

在线证书状态协议 (Online Certificate Status Protocol)
检查证书是否已撤销的方法。

单向认证 (one way authentication)
在此认证方法中，队列管理器会向客户机提供证书，但不会检查客户机到队列管理器的认证。

打开 (open)
建立对对象（如队列、主题或超链接）的访问。

开放式系统互连 (open systems interconnection，OSI)
依照国际标准化组织 (ISO) 的标准将开放式系统进行互连以交换信息。

开放式事务管理器访问 (Open Transaction Manager Access，OTMA)
IMS 的一个组件，它在 MVS 综合系统（sysplex）环境中实施以事务为基础的、无连接的客户机/服务器协议。 此协议的域限制为 z/OS 跨系统耦合设施 (XCF) 的域。 OTMA 将客户机连接至服务器，这样客户机可以在保持高性能的同时支持一个大型网络（或很多会话）。

OPM
请参阅原始程序模型 (original program model)。

原始程序模型 (original program model，OPM)
用于在引入 Integrated Language Environment (ILE) 模型之前编译源代码和创建高级语言程序的函数集。

OSGi 联盟 (OSGi Alliance)
由超过 20 家公司（包括 IBM）组成的联盟，旨在创立规范以制订语音、数据和多媒体无线及有线网络的开放管理标准的大纲。

OSI
请参阅开放式系统互连 (open systems interconnection)。

OSI 目录标准 (OSI directory standard)
称为 X.500 的标准，它定义一个综合的目录服务，其中包括信息模型、名称空间、功能模型和认证框架。 X.500 还定义客户机用于访问目录的目录访问协议 (DAP)。 轻量级目录访问协议 (LDAP) 解除了目录客户机执行 X.500 访问的某些负担，从而使目录可用于更广泛的机器和应用程序。

OTMA
请参阅开放式事务管理器访问 (Open Transaction Manager Access)。

出站通道 (outbound channel)
一个通道，它从传输队列获取消息并将这些消息发送到另一个队列管理器。

输出日志缓冲区 (output log-buffer)
WebSphere MQ for z/OS 中的缓冲区，它在恢复日志记录写入归档日志之前保留恢复日志记录。

输出参数 (output parameter)
MQI 调用的参数，当该调用完成或失败时，队列管理器在该参数中返回信息。

重载 (overloading)
在面向对象程序设计中，这指的是运算符或方法根据上下文具有不同含义的能力。 例如，在 C++ 中，当将函数和标准运算符与类类型配合使用时，用户可以重新定义函数和大多数标准运算符。 方法名称或运算符保持相同，但方法参数的类型和/或数目将不同。 此差别统称为函数或运算符的特征符，并且每个特征符需要不同的实现。

P

页集 (page set)
在 WebSphere MQ for z/OS 将数据（例如，队列和消息）从主存储器中的缓冲区移动到永久后备存储器（DASD）时使用的 VSAM 数据集。

父类 (parent class)
一个类，另一个类从该类继承实例方法、属性和实例变量。 另请参阅抽象类 (abstract class)。

部分存储库 (partial repository)
关于集群中队列管理器的信息的部分集合。 部分存储库由未主管完整存储库的所有集群队列管理器进行维护。 另请参阅完整存储库 (full repository)。

伙伴队列管理器 (partner queue manager)
请参阅远程队列管理器 (remote queue manager)。

PassTicket
在 RACF 安全注册中，指的是一个动态生成的一次性使用的随机替代密码，工作站或其他客户机可使用该密码向主机进行注册，而不是在网络上发送 RACF 密码。

PCF
请参阅可编程命令格式 (programmable command format)。

暂挂事件 (pending event)
由于 CICS 适配器发出连接请求而发生的未安排事件。

渗透 (percolation)
在错误恢复中，指的是沿着预先建立的路径将控制权从一个恢复例程传递给一个更高级别的恢复例程。

性能事件 (performance event)
事件的一个类别，它指示已出现限制条件。

性能跟踪 (performance trace)
一个 WebSphere MQ 跟踪选项，其中的跟踪数据将用于性能分析和调整。

永久动态队列 (permanent dynamic queue)
一个动态队列，仅在显式地请求了删除的情况下，才会在关闭该队列时将其删除。 队列管理器发生故障时将恢复永久动态队列，所以这些队列可以包含持久消息。 另请参阅临时动态队列 (temporary dynamic queue)。

持久消息 (persistent message)
在重新启动队列管理器后仍然存在的消息。 另请参阅非持久消息 (nonpersistent message)。

个人证书 (personal certificate)
您拥有对应专用密钥的证书。 与队列管理器或应用程序关联。

PGM
请参阅实用的常规多点广播 (Pragmatic General Multicast)。

PID
请参阅进程标识 (process ID)。

ping
一个命令，它将因特网控制报文协议 (ICMP) 回波请求包发送到期望接收应答的网关、路由器或主机。

PKCS
公用密钥密码术标准 (Public Key Cryptography Standard)。 一组密码术标准，其中：

7 项针对消息
11 项针对硬件安全模块
12 项针对密钥存储库中使用的文件格式

PKI
请参阅公共密钥基础结构 (public key infrastructure)。

纯文本 (plain text)
请参阅明文 (cleartext)。

恢复点 (point of recovery)
在 WebSphere MQ for z/OS 中的一组 WebSphere MQ for z/OS 页集的备份副本，以及恢复这些页集所需的相应日志数据集。 这些备份副本提供了潜在的重新启动点，以防发生页集丢失事件（例如，页集 I/O 错误）。

有害消息 (poison message)
队列中格式不正确的消息，接收应用程序无法处理这些消息。 可以将该消息重复传递到输入队列，并且应用程序可以重复退回该消息。

多态性 (polymorphism)
面向对象程序设计的一个特征，它允许根据实现方法的类以不同方式执行该方法。 多态性允许子类覆盖继承的方法而不影响父类的方法。 多态性还允许客户机从单个接口访问对象的两个或更多实现。

实用的常规多点广播 (Pragmatic General Multicast，PGM)
一种可靠的多点广播传输协议，它将一个可靠的包序列同时提供给多个接收方。

抢先关闭 (preemptive shutdown)
WebSphere MQ 中不等待连接的应用程序断开连接或当前 MQI 调用完成，就关闭队列管理器的一种关闭方式。 另请参阅立即关闭 (immediate shutdown) 和停顿关闭 (quiesced shutdown)。

首选计算机 (preferred computer)
由在 Microsoft Cluster Server 的控制之下运行的应用程序使用的主计算机。 在故障转移到另一个计算机之后，MSCS 将监视首选计算机，直到它被修复为止，并在它再次正常运行后，立即将应用程序移回到该首选计算机。

主体 (principal)
一个可与其他实体进行安全通信的实体。 主体由它的关联安全上下文标识，该安全上下文定义主体的访问权。

私下定义的对象 (privately defined object)
请参阅本地定义的对象 (locally defined object)。

私有方法和实例数据 (private methods and instance data)
在面向对象程序设计中，指的是仅同一个类的实现可以访问的方法和实例数据。

进程定义对象 (process definition object)
包含 WebSphere MQ 应用程序定义的 WebSphere MQ 对象。 例如，队列管理器在处理触发器消息时使用该定义。

进程标识 (process ID，PID)
表示一个进程的唯一标识。 进程标识是一个正整数，在进程生存期结束之前不能复用。

生产者 (producer)
创建并发送消息的应用程序。 另请参阅消息生产者 (message producer) 和发布者 (publisher)。

可编程命令格式 (programmable command format，PCF)
由以下应用程序使用的一种 WebSphere MQ 消息：用户管理应用程序使用该消息将 PCF 命令放入指定队列管理器的系统命令输入队列；用户管理应用程序使用该消息从指定队列管理器中取出 PCF 命令的结果；队列管理器使用该消息作为事件发生时的通知。 另请参阅 WebSphere MQ 脚本命令。

程序临时性修订 (program temporary fix，PTF)
对于 System i、System p 和 IBM Z Systems 产品，指的是包含一个或多个修订的程序包，该程序包可供所有许可的客户使用。 PTF 用于消除缺陷并可能提供增强功能。

属性
对象的特征，用于描述该对象。 您可以更改或修改属性。 这些属性可以描述对象的名称、类型、值或行为以及其他内容。

受保护方法和实例数据 (protected methods and instance data)
在面向对象程序设计中，指的是仅同一个类或派生类的实现或从朋友类可以访问的方法和实例数据。

代理预订 (proxy subscription)
代理预订是一个队列管理器为另一个队列管理器上发布的主题所作的预订。 代理预订会针对预订所预订的每一个主题字符串在队列管理器间流动。 您无需明确创建代理预订，队列管理器会替您执行此操作。

PTF
请参阅程序临时性修订 (program temporary fix)。

公用密钥密码术
一个使用两种密钥的密码术系统：对每个用户公开的公用密钥和仅对消息接收方公开的专用密钥。 公用密钥和专用密钥在某种程度上相关，即，公用密钥只能用于对消息进行加密，而且只能使用对应的专用密钥对这些消息进行解密。

公共密钥基础结构 (public key infrastructure，PKCS)
由数字证书、认证中心和其他注册中心组成的系统，它们验证并认证网络事务中涉及的每一方的有效性。

公用方法和实例数据 (public methods and instance data)
在面向对象程序设计中，指的是所有类都可以访问的方法和实例数据。

公用/专用密钥密码术 (public-private key cryptography)
请参阅公用密钥密码术 (public key cryptography)。

发布
使与指定主题相关的信息可供发布/预订系统中的队列管理器使用。

发布者
一个应用程序，它使关于指定主题的信息可用于发布/预订系统中的代理程序。 另请参阅生产者 (producer)。

发布/预订
一种消息传递交互，其中，基础结构会将由发布应用程序提供的信息传递给所有已注册为接收该类信息的预订应用程序。

发布/预订集群 (publish/subscribe cluster)
一组完全互相连接的队列管理器，构成发布/预订应用程序的多队列管理器网络的一部分。

put
在消息排队中，这指的是使用 MQPUT 或 MQPUT1 调用将消息放置到队列中。 另请参阅浏览 (browse) 和获取 (get)。

Q

队列
用于为消息排队应用程序存放消息的一个对象。 队列由队列管理器拥有并进行维护。

队列索引 (queue index)
WebSphere MQ for z/OS 中的一个消息标识列表或相关标识列表，可用于加快队列上 MQGET 操作的速度。

队列管理器
消息排队系统的一个组件，为应用程序提供排队服务。 另请参阅通道 (channel) 和多实例队列管理器 (multi-instance queue manager)。

队列管理器事件 (queue manager event)
一个事件，指示发生了与队列管理器使用的资源有关的错误情况（例如，队列不可用），或队列管理器中发生了重要更改（例如，已停止或启动队列管理器）。

队列管理器组 (queue manager group)
客户机通道定义表 (CCDT) 中的一组队列管理器，在与服务器建立连接时，客户机将尝试连接至该组队列管理器。

队列管理器级别安全性 (queue manager level security)
在 WebSphere MQ for z/OS 中，使用特定于队列管理器的 RACF 概要表执行的授权检查。

队列管理器集 (queue manager set)
WebSphere MQ 资源管理器中对队列管理器的一种分组，允许用户对组中的所有队列管理器执行操作。

共享通道恢复
在 WebSphere MQ for z/OS 中同一综合系统中的一组队列管理器，它们可以访问存储在共享库中的一组对象定义和存储在耦合设施中的一组共享队列。 另请参阅共享队列 (shared queue)。

队列共享组级别安全性 (queue-sharing group level security)
在 WebSphere MQ for z/OS 中，使用由队列共享组中所有队列管理器共享的 RACF 概要表执行的授权检查。

停顿 (quiesce)
在允许活动操作正常完成后结束过程或关闭系统。

停顿关闭 (quiesced shutdown)

1. 在 WebSphere MQ 中，允许所有已连接的应用程序断开连接的关闭队列管理器的方式。 另请参阅立即关闭 (immediate shutdown) 和抢先关闭 (preemptive shutdown)。
2. CICS 适配器的一种关闭方式，只有在所有当前活动的任务都已完成后，适配器才与 WebSphere MQ 断开连接。 另请参阅强制关闭 (forced shutdown)。

停顿 (quiescing)
在 WebSphere MQ 中，队列管理器停止前的状态。 处于此状态时，将允许程序完成处理，但不允许启动任何新程序。

定额磁盘 (quorum disk)
只有 Microsoft Cluster Server 才能访问的磁盘，用于存储集群恢复日志并确定服务器是否在运行。 每次只有一个服务器可以拥有定额磁盘。 集群中的服务器可以进行协商以获取所有权。

R

RACF
请参阅资源访问控制设施 (Resource Access Control Facility)。

RAID
请参阅独立磁盘冗余阵列 (Redundant Array of Independent Disks)。

RAS
可靠性，可用性和可维护性。

RBA
请参阅相对字节地址 (relative byte address)。

返回码
请参阅返回码 (return code)。

预读
一个选项，它允许在应用程序请求消息之前将消息发送到客户机。

原因码 (reason code)
一个返回码，它描述消息队列接口 (MQI) 调用失败或部分成功的原因。

接收出口 (receive exit)
一种通道出口程序，在消息通道代理程序 (MCA) 发出“通信接收”后已重新获得控制权并且从通信连接中接收到数据单元之后，就会调用该出口程序。 另请参阅发送出口 (send exit)。

接收方通道 (receiver channel)
消息排队中的一个通道，它响应发送方通道、从通信链路中获取消息并将这些消息放置到本地队列中。

恢复日志 (recovery log)
WebSphere MQ for z/OS 中的数据集，其中包含恢复消息、队列和 WebSphere MQ 子系统所需要的信息。 另请参阅归档日志 (archive log)。

恢复终止管理器 (recovery termination manager，RTM)
一个程序，它通过将控制权传递给与终止功能相关联的恢复例程来处理任务的所有正常终止和异常终止。

独立磁盘冗余阵列 (Redundant Array of Independent Disks，RAID)
包含两个或更多物理磁盘驱动器的集合，这些磁盘驱动器对主机表示为一个或多个逻辑磁盘驱动器。 如果某个物理设备发生故障，那么由于数据冗余性，可以从阵列中的其他磁盘驱动器读取或重新生成数据。

参考消息 (reference message)
一条消息，它引用一段要传输的数据。 参考消息由消息出口程序进行处理，这些程序在该消息中添加或除去数据，所以允许传输这些数据，而不必将它们存储在任何队列中。

注册表 (registry)
一种存储库，包含用户、系统和软件的访问和配置信息。

注册表编辑器 (Registry Editor)
Windows 系统中允许用户编辑注册表的程序。

注册表组织结构 (registry hive)
Windows 系统中数据存储在注册表中的结构。

相对字节地址 (relative byte address，RBA)
数据记录或控制区间与分配给它所属的数据集或文件的存储空间开始位置的偏移。

可靠多点广播消息传递 (reliable multicast messaging，RMM)
用于一对多数据传递或多对多数据交换的高吞吐量、低等待时间的传输光纤网，采用了面向消息的中间件发布/预订方式。 RMM 使用 IP 多点广播基础结构来确保可伸缩的资源保护和及时的信息分发。

远程队列 (remote queue)
属于远程队列管理器的队列。 程序可以将消息放置在远程队列中，但它们无法从远程队列获取消息。 另请参阅本地队列 (local queue)。

远程队列管理器 (remote queue manager)
程序未连接至的队列管理器，即使它正在该程序所在的系统上运行。 另请参阅本地队列管理器 (local queue manager)。

远程队列对象 (remote queue object)
属于本地队列管理器的 WebSphere MQ 对象。 此对象定义另一个队列管理器所拥有的队列的属性。 另外，它还用于队列管理器别名判别和应答队列别名判别。

远程排队 (remote queuing)
在消息排队中，指的是提供服务以允许应用程序将消息放置在属于其他队列管理器的队列中。

应答消息 (reply message)
一种消息，用于应答请求消息。 另请参阅报告消息 (report message) 和请求消息 (request message)。

应答队列 (reply-to queue)
一个队列的名称，发出 MQPUT 调用的程序想要将应答消息或报告消息发送到该队列。

报告消息 (report message)
一种消息，它提供关于另一条消息的信息。 报告消息可以指出消息已传递、已到达其目标、已到期或因为某个原因而无法处理。 另请参阅应答消息 (reply message) 和请求消息 (request message)。

存储库 (repository)
集群中成员队列管理器的相关信息的集合。 此信息包括队列管理器名称、队列管理器位置、队列管理器通道以及它们所主管的队列。

库队列管理器 (repository queue manager)
一个队列管理器，它主管关于集群的信息的完整存储库。

请求者通道 (requester channel)
在消息排队中，指的是可以在本地启动以启动服务器通道操作的通道。 另请参阅服务器通道 (server channel)。

请求消息 (request message)
一种消息，用于从另一个程序请求应答。 另请参阅应答消息 (reply message) 和报告消息 (report message)。

请求/应答 (request/reply)
消息传递应用程序的类型，在该类型中使用请求消息从另一个应用程序请求应答。 另请参阅数据报 (datagram)。

RESLEVEL
WebSphere MQ for z/OS 中的一个选项，控制为确保 API 资源安全性而检查的用户标识数。

解析路径 (resolution path)
一组队列，当应用程序在 MQOPEN 调用的输入中指定别名或远程队列时，将打开这些队列。

资源 (resource)
作业、任务或正在运行的程序所需的计算系统或操作系统的设施。 资源包括主存储器、输入/输出设备、处理单元、数据集、文件、库、文件夹、应用程序服务器以及控制或处理程序。

资源访问控制设施（Resource Access Control Facility，RACF）
一种 IBM 许可程序，它通过以下方法来提供访问控制：向系统识别用户；验证系统的用户；授权对受保护资源的访问权；记录未经授权就试图进入系统的情况；并记录对受保护资源的访问。

资源适配器 (resource adapter)
Java Enterprise Edition 连接器体系结构的一种实施，允许 JMS 应用程序和消息驱动的 bean（在应用程序服务器中运行）访问 WebSphere MQ 队列管理器的资源。

资源管理器 (resource manager)
一个应用程序、程序或事务，管理和控制对共享资源（如内存缓冲区和数据集）的访问。 WebSphere MQ、CICS 和 IMS 都是资源管理器。

资源恢复服务 (Resource Recovery Services，RRS)
z/OS 的一个组件，使用同步点管理器来协调参与资源管理器之间的更改。

响应程序 (responder)
在分布式排队中，这指的是应答来自另一个系统的网络连接请求的程序。 另请参阅起始符、启动程序 (initiator)。

再同步 (resynch)
WebSphere MQ 中的一种选项，它指示通道启动并解析任何处于不确定状态的消息，而无需重新启动消息传送。

返回码 (return code, RC)
由程序返回以指示处理结果的值。 完成代码和原因码是返回码的示例。

返回发送方 (return-to-sender)
一个可用于无法传递消息的 MCA 的选项。 MCA 可以将该消息发送回发起方。

Rivest-Shamir-Adleman 算法 (Rivest-Shamir-Adleman algorithm，RSA)
由 RSA Data Security, Inc 开发的公用密钥加密技术，在 SSL 的 IBM 实现中使用。

RMM
请参阅可靠的多点广播消息传递 (sreliable multicast messaging)。

回滚 (rollback)
请参阅回退 (backout)。

根证书
链中最顶级的证书。 如果这是自签名证书，那么只能用于签署其他证书。 另请参阅自签名证书 (self-signed certificate)。

RRS
请参阅资源恢复服务 (Resource Recovery Services)。

RSA
请参阅 Rivest-Shamir-Adleman 算法 (Rivest-Shamir-Adleman algorithm)。

RTM
请参阅恢复终止管理器 (recovery termination manager)。

规则表 (rules table)
一个控制文件，它包含一个或多个规则，死信队列处理程序将这些规则应用于死信队列 (DLQ) 中的消息。

S

SAF
请参阅存储转发 (store and forward)。

可伸缩并行 2 (Scalable Parallel 2，SP2)
IBM 的并行 UNIX 系统：高速网络中的有效并行 AIX 系统。

SDK
请参阅软件开发包 (software development kit)。

SDWA
请参阅系统诊断工作区 (system diagnostic work area)。

SECMEC
请参阅安全性机制 (security mechanism)。

安全套接字层 (Secure Sockets Layer，SSL)
提供通信隐私的安全性协议。 使用 SSL，客户机/服务器应用程序可使用避免窃听、篡改和伪造消息的方式进行通信。 另请参阅认证中心 (certificate authority)。

安全性启用接口 (security enabling interface，SEI)
一种 WebSphere MQ 接口，客户或供应商编写的用于检查授权、提供用户标识或执行认证的程序必须符合该接口。

安全出口
一个通道出口程序，在通道启动时完成初始数据协商之后，将立即调用该程序。 安全性出口通常成对地工作，并且可以在消息通道和 MQI 通道上调用这些出口。 安全性出口的主要用途是使通道每一端的消息通道代理程序 (MCA) 能够认证其伙伴。

安全标识 (security identifier，SID)
Windows 系统中用户标识的一个附件，它标识定义用户的 Windows 安全性帐户管理器数据库中的完整用户帐户详细信息。

安全机制 (security mechanism，SECMEC)
用于实现安全服务的技术工具或技巧。 机制可以单独使用，也可以与其他机制协作配合使用，以提供特定服务。 安全性机制的示例包括访问控制表、密码术和数字签名。

安全性消息 (security message)
由通道两端调用的安全性出口发送以便互相通信的消息之一。 未定义安全性消息的格式，它由用户确定。

安全服务 (security service)
计算机系统中的服务，用于保护计算机的资源。 访问控制是安全服务的一个示例。

安全支持提供程序接口 (Security Support Provider Interface，SSI)
已联网的应用程序调用若干安全性支持提供程序 (SSP) 之一的方法，以建立已认证的连接并通过那些连接安全地交换数据。 可用于 Windows 系统。

分段
对消息的一种分割，当消息对于队列管理器、队列或应用程序太大时，该消息将分割为一些较小的物理消息，然后由接收队列管理器或应用程序重新组装这些消息。

SEI
请参阅安全性启用接口 (security enabling interface)。

选择器 (selector)
数据项的标识。 在 WebSphere MQ 管理接口 (MQAI) 中，有两种类型的选择器：用户选择器和系统选择器。

自签名证书 (self-signed certificate)
构造方式与数字证书类似，但由其主体签署的证书。 与数字证书不同的是，不能以可信方式将自签名证书用于向其他方认证公用密钥。

信标 (semaphore)
在 UNIX 和 Linux 系统中，这是扩展信号功能的两个进程之间的一般通信方法。

发送方通道 (sender channel)
消息排队中的一个通道，它启动传输，从传输队列中除去消息，并通过通信链路将这些消息移至接收方或请求者通道。

发送出口 (send exit)
一种通道出口程序，在调用该出口程序之后，消息通道代理程序 (MCA) 就会发出“通信发送”以通过通信连接发送数据单元。 另请参阅接收出口 (receive exit)。

顺序分组交换协议 (Sequenced Packet Exchange protocol，SPX)
一个面向会话的网络协议，它在网络中的两个节点之间提供面向连接的服务，主要由客户机/服务器应用程序使用。 它依赖于因特网包交换 (IPX) 协议、提供流量控制和错误恢复并保证物理网络的可靠性。

序号复位值 (sequence number wrap value)
WebSphere MQ 中确保通信链路两端同时复位它们当前消息顺序号的一种方法。 使用序号传输消息将确保接收通道可以在存储消息时重新建立消息顺序。

序列化 (serialization)
在面向对象程序设计中，以顺序方式将程序内存中的数据写入通信介质。

服务器

1. 一种软件程序或计算机，用于向其他软件程序或其他计算机提供服务。 另请参阅客户机 (client)。
2. 一种队列管理器，用于向远程工作站上运行的客户机应用程序提供队列服务。

服务器通道 (server channel)
消息排队中的一个通道，它响应请求者通道，从传输队列中除去消息，并通过通信链路将这些消息移至该请求者通道。 另请参阅请求者通道 (requester channel)。

服务器连接通道类型 (server-connection channel type)
与运行队列管理器的服务器相关联的 MQI 通道定义的类型。 另请参阅客户机连接通道类型 (client-connection channel type)。

服务时间间隔 (service interval)
一个时间间隔，队列管理器在确定是否满足了服务时间间隔事件的条件时，会将该时间间隔与放置或获取和后续获取之间的耗用时间进行比较。 队列的服务时间间隔由队列属性指定。

服务时间间隔事件 (service interval event)
与服务时间间隔相关的事件。

服务对象 (service object)
一种对象，它可以在队列管理器启动时启动其他过程，并可以在队列管理器停止时停止这些过程。

会话 (session)
网络上两个站、软件程序或设备之间的逻辑或虚拟连接，允许两个元素在整个会话的持续时间内进行通信和交换数据。

会话标识 (session ID)
请参阅会话标识 (session identifier)。

会话标识 (session identifier, session ID)
WebSphere MQ for z/OS 中对 CICS 唯一的标识，它定义了消息通道代理程序在将消息从传输队列移至链路时要使用的通信链路。

会话级别认证 (session-level authentication)
在系统网络体系结构 (SNA) 中，指的是会话级别安全性协议，它允许两个逻辑单元 (LU) 在激活会话时互相进行认证。 会话级别认证也称为 LU-LU 验证。

会话级别密码术 (session-level cryptography)
在系统网络体系结构 (SNA) 中，指的是对两个逻辑单元 (LU) 之间的会话中流动的数据进行加密和解密的方法。

共享入站通道 (shared inbound channel)
WebSphere MQ for z/OS 中由侦听器使用组端口启动的通道。 共享通道的通道定义可以存储在页集 0（私有）或共享库（全局）中。

共享出站通道 (shared outbound channel)
WebSphere MQ for z/OS 中从共享传输队列中移动消息的通道。 共享通道的通道定义可以存储在页集 0（私有）或共享库（全局）中。

共享队列 (shared queue)
WebSphere MQ for z/OS 中的一种本地队列。 该队列中的消息存储在耦合设施中，可以由队列共享组中的一个或多个队列管理器访问。 该队列的定义存储在共享库中。 另请参阅队列共享组 (queue-sharing group)。

共享库 (shared repository)
WebSphere MQ for z/OS 中的共享 DB2 数据库，用于保留已被全局定义的对象定义。

共享对话
用于要共享一个通道实例的多个对话的设施，或共享一个通道实例的对话。

shell
用户与操作系统之间的软件接口。 shell 一般属于两个类别的其中一个：命令行 shell 和图形 shell，前者提供操作系统的命令行界面，后者提供图形用户界面 (GUI)。

SID
请参阅安全标识 (security identifier)。

信号 (signal)
一种机制，进程可以通过它得知系统中发生的事件或受到该事件的影响。 这种事件的示例包括硬件异常和进程执行的特定操作。

发信号 (signaling)
WebSphere MQ for z/OS 和 WebSphere MQ for Windows 中的一个功能，它允许操作系统在有预期的消息到达队列时向程序发通知。

特征符 (signature)
与方法相关联的类型的集合。 特征符包括返回值（如果有返回值）的类型以及方法的每个自变量的编号、顺序和类型。

签署者证书 (signer certificate)
用于验证某个证书的颁发者的数字证书。 对于 CA，签署者证书是根 CA 证书。 对于创建自签名证书以用于测试的用户，签署者证书是该用户的个人证书。

单一实例队列管理器 (single instance queue manager)
没有多个实例的队列管理器。 另请参阅多实例队列管理器 (multi-instance queue manager)。

单一记录 (single logging)
记录 WebSphere MQ for z/OS 活动的一种方法，其中每项更改都只记录在一个数据集中。 另请参阅双记录 (dual logging)。

单一阶段回退 (single-phase backout)
一种方法，使用该方法时，不能允许完成正在执行的操作，并且必须撤销属于该操作的所有更改。

单阶段落实 (single-phase commit)
一种方法，使用该方法时，程序可以落实对承诺资源的更新，而不必将这些更新与程序已对另一个资源管理器控制的资源作出的更新进行协调。

SIT
请参阅系统初始化表 (system initialization table)。

SMF
请参阅系统管理设施 (System Management Facilities)。

SNA
请参阅系统网络体系结构 (Systems Network Architecture)。

软件开发包 (software development kit，SDK)
工具、API 和文档的集合，用于辅助用特定计算机语言进行的软件开发或针对特定操作环境进行的软件开发。

源队列管理器 (source queue manager)
请参阅本地队列管理器 (local queue manager)。

SP2
请参阅可伸缩并行 2 (Scalable Parallel 2)。

SPX
请参阅顺序分组交换协议 (Sequenced Packet Exchange protocol)。

SSI
请参阅安全性支持提供程序接口 (Security Support Provider Interface)。

SSL
请参阅安全套接字层 (Secure Sockets Layer)。

SSLPeer
颁发者中的值表示远程个人证书的专有名称。

SSL 或 TLS 客户机 (SSL or TLS client)
启动连接结束。 队列管理器的一个出站通道也是 SSL 或 TLS 客户机。

备用队列管理器实例 (standby queue manager instance)
正在运行的多实例队列管理器的一个实例，它已准备好从活动实例进行接管。 一个多实例队列管理器有一个或多个备用实例。

节 (stanza)
文件中的一组行，它们组合起来形成一个公共功能或定义系统的一个部件。 节通常由空白行或冒号进行分隔，并且每个节都具有一个名称。

星型连接的通信网络 (star-connected communications network)
一种网络，其中所有节点都连接至一个中央节点。

存储类 (storage class)
WebSphere MQ for z/OS 中用于为特定队列保留消息的页集。 存储类在定义队列时指定。

存储转发 (store and forward, SAF)
在将数据网络中的包、消息或帧重新传输到其目标之前临时对其进行存储。

流式方法 (streaming)
在面向对象程序设计中，指的是将类信息和对象实例数据串行化的过程。

预订
请求关于主题的信息。

子系统 (subsystem)
z/OS 中的一种服务提供者，用于执行一个或多个功能，但在被请求之前不会执行任何操作。 例如，每个 WebSphere MQ for z/OS 队列管理器或 DB2 for z/OS 数据库管理系统实例是一个 z/OS 子系统。

管理程序调用 (supervisor call，SVC)
一条指令，它中断正在运行的程序并将控制权传递给管理程序，以便它可以执行该指令所指示的特定服务。

SVC
请参阅管理程序调用 (supervisor call)。

转换 (switchover)
从活动的多实例队列管理器实例更改为备用实例。 转换是因为操作员有意停止活动的多实例队列管理器实例而发生的。

开关概要信息 (switch profile)
WebSphere MQ for z/OS 中的一个 RACF 概要表，在 WebSphere MQ 启动时或发出新的安全命令时使用。 WebSphere MQ 检测到的每个开关概要表都会关闭对指定资源进行的检查。

对称密钥密码术 (symmetric key cryptography)
一个密码术系统，在该系统中，消息的发送方和接收方共享单个公共密钥，该密钥用于对此消息进行加密和解密。 另请参阅非对称密钥密码术 (asymmetric key cryptography)。

症状字符串 (symptom string)
为搜索 IBM 软件支持数据库设计的、以结构化格式显示的诊断信息。

同步消息传递 (synchronous messaging)
一种在程序之间进行通信的方法，使用这种方法时，程序将消息放置在消息队列中，然后在恢复自己的处理之前等待对其消息的应答。 另请参阅异步消息传递 (asynchronous messaging)。

同步点
事务处理期间受保护的资源一致时所处的位置。

综合系统 (sysplex)
通过特定的多系统硬件组件和软件服务彼此通信的一组 z/OS 系统。

系统包 (system bag)
由 MQAI 创建的一种数据包。

系统控制命令 (system control commands)
用于操作特定于平台的实体（如缓冲池、存储类和页集）的命令。

系统诊断工作区 (system diagnostic work area，SDWA)
在 z/OS 环境中，记录到 SYS1.LOGREC 条目中以描述程序或硬件错误的数据。

系统初始化表 (system initialization table，SIT)
包含由 CICS 在启动时使用的参数的表。

系统项 (system item)
由 MQAI 创建的一种数据项。

系统管理设施 (System Management Facilities，SMF)
z/OS 的一个组件，用于收集并记录各种系统信息以及与作业相关的信息。

系统选择器 (system selector)
WebSphere MQ 管理接口（MQAI）中的一个系统项标识，它在数据包创建时包括在数据包中。

系统网络体系结构 (Systems Network Architecture，SNA)
对用于传输信息并控制网络的配置和操作的逻辑结构、格式、协议和操作序列的描述。

T

篡改 (tampering)
通信安全性的一种违规，在该过程中，传送中的信息将被更改或替换，然后发送给接收方。 另请参阅窃听 (eavesdropping) 和冒名 (impersonation)。

目标库高级限定符 (target library high-level qualifier，thlqual)
z/OS 目标数据集名称的高级限定符。

目标队列管理器 (target queue manager)
请参阅远程队列管理器 (remote queue manager)。

任务控制块 (task control block，TCB)
一个 z/OS 控制块，用于传送关于已连接至子系统的地址空间中任务的信息。

任务切换 (task switching)
若干任务之间 I/O 操作和处理的重叠。

TCB
请参阅任务控制块 (task control block)。

TCP
请参阅传输控制协议 (Transmission Control Protocol)。

TCP/IP
请参阅传输控制协议/因特网协议 (Transmission Control Protocol/Internet Protocol)。

技术说明 (technote)
关于单个主题的简短文档。

遥测通道 (telemetry channel)
WebSphere MQ 上的队列管理器与 MQTT 客户机之间的通信链路。 每个通道可能有一个或多个遥测设备与它相连。

遥测服务 (telemetry service)
遥测服务是 WebSphere MQ 服务，用于处理 MQTT 协议的服务器端 (请参阅 *MQTT* 服务器)。 遥测服务托管遥测通道。 它有时称为 MQXR (MQ 扩展访问范围) 服务。

临时动态队列 (temporary dynamic queue)
一个动态队列，关闭该队列时会将其删除。 队列管理器发生故障时不会恢复临时动态队列，所以这些队列只能包含非持久消息。 另请参阅永久动态队列 (permanent dynamic queue)。

太字节空间 (teraspace)
一太字节临时存储器区域，它提供专用于某个进程的存储器。

终止通知 (termination notification)
在 CICS 子系统成功连接至 WebSphere MQ for z/OS 时激活的一个暂挂事件。

thlqual
请参阅目标库高级限定符 (target library high-level qualifier)。

线程 (thread)
控制进程的计算机指令流。 在一些操作系统中，线程即为进程中最小的操作单元。 若干线程可并行运行，执行不同的作业。

TID
请参阅事务标识 (transaction identifier)。

独立于时间的消息传递 (time-independent messaging)
请参阅异步消息传递 (asynchronous messaging)。

TLS
请参阅传输层安全性 (Transport Layer Security)。

TMF
请参阅事务管理器设施 (Transaction Manager Facility)。

TMI
请参阅触发器监视器接口 (trigger monitor interface)。

主题主机路由 (topic host routing)
用于在发布/预订集群中路由发布的一种选项。 利用主题主机路由，只有选中的集群队列管理器才能托管主题定义。 来自非托管队列管理器的发布会通过托管队列管理器路由至集群中带有匹配预订的任何队列管理器。

TP
请参阅事务程序 (transaction program)。

跟踪
计算机程序或事务的处理过程的记录。 从跟踪中收集的信息可以用于评估问题和性能。

事务标识
请参阅事务标识 (transaction identifier)。

事务标识 (transaction identifier，TID, transaction ID, XID)
指定给某一事务且用于标识与该事务相关联的操作的唯一名称。

事务管理器 (transaction manager)
一个软件单元，它通过管理全局事务并对落实或回滚这些全局事务的决策进行协调来协调资源管理器的活动。

事务管理器设施 (Transaction Manager Facility，TMF)
WebSphere MQ for HP NonStop Server 中的一个子系统，用于保护业务事务和数据库完整性。

事务程序 (transaction program，TP)
在 SNA 网络中处理事务的程序。

传输控制协议 (Transmission Control Protocol, TCP)
在因特网中和在遵从因特网工程任务组织 (IETF) 标准的任何网络中用作因特网协议的通信协议。 TCP 在分组交换通信网络和此类网络的互联系统中提供了可靠的主机间协议。 另请参阅因特网协议 (Internet Protocol)。

传输控制协议/因特网协议 (Transmission Control Protocol/Internet Protocol, TCP/IP)
一组业界标准的、非专有的通信协议，通过不同类型的互连网络来为应用程序之间提供可靠的端到端连接。

传输程序 (transmission program)
请参阅消息通道代理程序 (message channel agent)。

传输队列 (transmission queue)
一个本地队列，将发送到远程队列管理器的已准备好的消息临时存储在该队列中。

传输层安全性 (Transport Layer Security)
一组加密规则，使用经验证的证书和加密密钥来保护因特网通信的安全。 TLS 是 SSL 协议的更新。

触发队列 (triggered queue)
一种本地队列（通常是应用程序队列），其中已启用触发，这样在发生触发器事件时将写入一条消息。触发器消息通常会写入启动队列。

触发 事件
导致队列管理器在启动队列中创建触发器消息的事件，如消息到达队列。

触发
WebSphere MQ 中的一种设施，它允许队列管理器在满足队列上预先确定的条件时自动启动应用程序。

触发 消息
一条消息，它包含关于触发器监视器要启动的程序的信息。

触发器监视器
一个连续运行的应用程序，它为一个或多个启动队列提供服务。 当触发器消息到达启动队列时，触发器监视器将检索该消息。 它使用触发器消息中的信息来启动一个进程，该进程为发生触发器事件的队列提供服务。

触发器监视器接口 (trigger monitor interface，TMI)
一种 WebSphere MQ 接口，客户编写或供应商编写的触发器监视器程序必须符合该接口。

信任库 (truststore)
安全性中的一个存储器对象 (文件或硬件加密卡)，其中公用密钥采用可信证书形式存储以用于认证。 在某些应用程序中，这些可信证书会被移入应用程序密钥库以与专用密钥存储在一起。

两阶段落实 (two-phase commit)
这是一个分为两个步骤的过程，将通过该过程来落实可恢复资源和外部子系统。 在第一个步骤中，将轮询数据库管理器子系统以确保它们已作好落实准备。 如果所有子系统都作出肯定响应，那么数据库管理器将指示它们进行落实。

双向认证 (two way authentication)
在此认证方法中，队列管理器与客户机会彼此提供证书。 也称为相互认证。

类型
一个特征，用于指定数据的内部格式并确定可使用数据的方式。

U

UDP
请参阅用户数据报协议 (User Datagram Protocol)。

未经授权的访问 (unauthorized access)
在没有许可权的情况下访问计算机系统中的资源。

未传递的消息队列 (undelivered message queue)
请参阅死信队列 (dead-letter queue)。

撤销/重做记录 (undo/redo record)
恢复过程中使用的日志记录。 记录的重做部分描述了要对 WebSphere MQ 对象进行的更改。 撤销部分描述如何在未落实工作的情况下取消更改。

恢复单元 (unit of recovery)
单个资源管理器（如 DB2 for z/OS 的实例）中的可恢复操作序列。 另请参阅工作单元 (unit of work)。

工作单元 (unit of work，UOW)
由两个一致点之间的应用程序执行的可恢复操作序列。 工作单元在事务启动时或在用户请求的同步点处开始。 它在用户请求的同步点处或事务结束时结束。

UOW
请参阅工作单元 (unit of work)。

用户包 (user bag)
在 MQAI 中，指的是由用户创建的一种数据包。

用户数据报协议 (User Datagram Protocol，UDP)
一个因特网协议，它提供不可靠的无连接数据报服务。 它允许一个机器或进程中的应用程序将数据报发送到另一个机器或进程中的应用程序。

用户项 (user item)
在 MQAI 中，指的是由用户创建的一种数据项。

用户选择器 (user selector)
在 WebSphere MQ 管理接口（MQAI）接口中的标识，它与数据项一起放入数据包以标识该数据项。 WebSphere MQ 为 WebSphere MQ 对象提供预定义的用户选择器。

用户令牌 (user token，UTOKEN)
一种 RACF 安全性令牌，用于封装或表示用户的安全性特征。 RACF 为系统中的每个用户分配一个 UTOKEN。

实用程序 (utility)
WebSphere MQ 中提供的一组程序，它们为系统操作员或系统管理员提供除 WebSphere MQ 命令提供的设施之外的设施。

UTOKEN
请参阅用户令牌 (user token)。

V

值
数据项的内容。 该值可以是整数、字符串或其他数据包的句柄。

虚方法 (virtual method)
在面向对象程序设计中，指的是展示多态性的方法。

W

WebSphere MQ
一系列 IBM 许可程序，它们提供消息排队服务。

WebSphere MQ 管理接口（WebSphere MQ Administration Interface，MQAI）
通过使用数据包在 WebSphere MQ 队列管理器上执行管理任务的编程接口。 数据包允许用户处理 WebSphere MQ 对象的属性（或参数）。

用于 .NET 的 WebSphere MQ 类
一组类，允许用 .NET 编程框架编写的程序作为 WebSphere MQ 客户机连接至 WebSphere MQ 或直接连接至 WebSphere MQ 服务器。

WebSphere MQ for C++ 类（WebSphere MQ classes for C++）
一组用 C++ 编程语言封装 WebSphere MQ 消息队列接口（MQI）的类。

WebSphere MQ Java 类
以 Java 编程语言封装 WebSphere MQ 消息队列接口 (MQI) 的一组类。

WebSphere MQ 完全受管 .NET 客户机 (WebSphere MQ fully managed .NET client)
它是 WebSphere MQ 产品的一部分，无需安装整个队列管理器即可在系统上安装。 WebSphere MQ .NET 客户机由完全受管 .NET 应用程序使用，并与服务器系统上的队列管理器通信。 非完全受管的 .NET 应用程序使用 WebSphere MQ MQI 客户机。 另请参阅 客户机， WebSphere MQ Java 客户机，WebSphere MQ MQI 客户机。

WebSphere MQ Java 客户机
它是 WebSphere MQ 产品的一部分，无需安装整个队列管理器即可在系统上安装。 WebSphere MQ Java 客户机由 Java 应用程序 (WebSphere MQ classes for Java 和 WebSphere MQ classes for JMS) 使用，并与服务器系统上的队列管理器进行通信。 另请参阅客户机 (client)、WebSphere MQ 完全受管 .NET 客户机 (WebSphere MQ fully managed .NET client) 和 WebSphere MQ MQI 客户机 (WebSphere MQ MQI client)。

WebSphere MQ MQI 客户机
它是 WebSphere MQ 产品的一部分，无需安装整个队列管理器即可在系统上安装。 WebSphere MQ MQI 客户机接受来自应用程序的 MQI 调用，并与服务器系统上的队列管理器进行通信。 另请参阅 客户机， WebSphere MQ 完全受管 .NET 客户机， WebSphere MQ Java 客户机。

WebSphere MQ 脚本命令（WebSphere MQ script commands，MQSC）
用于操纵 WebSphere MQ 对象的人可读的命令，在所有平台上一致。 另请参阅可编程命令格式 (programmable command format)。

WebSphere MQ 服务器 (WebSphere MQ server)
为一个或多个客户机提供排队服务的队列管理器。 所有 WebSphere MQ 对象，例如，队列仅存在于队列管理器系统，即 MQI 服务器上。 服务器还可以支持一般的本地 MQI 应用程序。

WebSphere MQ Telemetry
支持可嵌入到多个不同设备平台上运行的智能设备的小型客户机库。 对客户机构建的应用程序使用 MQ Telemetry Transport (MQTT) 协议和 WebSphere MQ 遥测服务来通过 WebSphere MQ 可靠地发布和预订消息。

设备的 WebSphere MQ Telemetry 守护程序 (WebSphere MQ Telemetry daemon for devices)
请参阅设备的 MQTT 守护程序 (MQTT daemon for devices)。

窃听 (wiretapping)
获取对信息的访问权的行为，它通过电话线或通信中使用其他类型的导线进行。 窃听的目的是为了对信息进行未经授权的访问而不被检测到。

X

X509
国际电信联盟关于 PKI 的标准。 指定公用密钥证书和公用密钥密码术的格式。

XCF
请参阅跨系统耦合设施 (cross-system coupling facility)。

XID
请参阅事务标识 (transaction identifier)。

X/Open XA
X/Open 分布式事务处理 XA 接口。 分布式事务通信的建议标准。 该标准在提供对事务中共享资源的访问权的资源管理器之间指定了双向接口，以及在监控和解析事务的事务服务之间指定了双向接口。

IBM MQ 技术概述

使用 IBM MQ 可连接应用程序并管理整个组织中的信息分发。

IBM MQ 使程序能够使用一致的应用程序编程接口在由不同组件 (处理器，操作系统，子系统和通信协议) 组成的网络中相互通信。 使用此接口设计和编写的应用程序称为消息排队应用程序。

使用以下子主题来了解消息排队以及 IBM MQ 提供的其他功能。

相关概念

第 8 页的『IBM MQ 简介』
您可以使用 IBM MQ 使应用程序能够在不同时间和许多不同的计算环境中进行通信。

相关参考

第 126 页的『消息排队的主要功能和优点』
此信息突出显示了消息排队的一些功能和优点。 它描述了诸如消息排队的安全性和数据完整性之类的功能。

相关信息

规划 IBM MQ 体系结构
查找产品需求和更新的支持信息
IBM MQ Managed File Transfer

消息排队简介

IBM MQ 产品支持程序使用一致的应用程序编程接口在不同组件 (处理器，操作系统，子系统和通信协议) 的网络中相互通信。

使用此接口设计和编写的应用程序称为 消息排队 应用程序，因为它们使用 消息传递 和 排队 样式:

消息传递	程序通过在消息中相互发送数据而不是直接相互调用来进行通信。
排队	消息放置在存储器中的队列上，允许程序以不同的速度和时间在不同的位置独立运行，并且没有它们之间的逻辑连接。

消息排队已在数据处理中使用多年。 现在最常用的是电子邮件。 无需排队，长途发送电子消息需要路由上的每个节点都可用于转发消息，并且收件人要登录并意识到您正在尝试向他们发送消息。 在排队系统中，消息存储在中间节点，直到系统准备好转发这些消息为止。 在他们的最终目的地，他们被存储在一个电子邮箱中，直到收件人准备好阅读它们。

即便如此，今天很多复杂的业务交易都是在不排队的情况下处理的。 在大型网络中，系统可能正在维护数千个处于即用即用状态的连接。 如果系统的一个部分迁到问题，那么系统的许多部分将变为不可用。

您可以将消息排队视为营销计划的电子邮件。 在消息排队环境中，构成应用程序套件的一部分的每个程序都会执行明确定义的自包含函数以响应特定请求。 要与另一个程序通信，程序必须将消息放在预定义的队列上。 另一个程序从队列中检索消息，并处理消息中包含的请求和信息。 所以消息排队是一种程序间通信的样式。

排队是在应用程序准备好处理消息之前保留消息的机制。 排队允许您:

- 在程序之间进行通信 (可能每个程序都在不同的环境中运行) ，而不必编写通信代码。
- 选择程序处理消息的顺序。
- 当消息数超过阈值时，通过安排多个程序为队列提供服务来平衡系统上的负载。
- 通过安排备用系统在主系统不可用时为队列提供服务，提高应用程序的可用性。

什么是消息队列?

消息队列 (简称为队列) 是可将消息发送至的指定目标。 消息会在队列上累积，直到服务这些队列的程序检索到这些消息为止。

© Copyright IBM Corp. 2007, 2023

队列驻留在队列管理器中并由其管理 (请参阅 第 128 页的『消息排队术语』)。 队列的物理性质取决于运行队列管理器的操作系统。 队列可以是计算机内存中的易失性缓冲区，也可以是永久存储设备 (例如磁盘) 上的数据集。 队列的物理管理由队列管理器负责，不会对参与的应用程序显示。

程序仅通过队列管理器的外部服务访问队列。 他们可以打开队列，将消息放在队列上，从中获取消息，然后关闭队列。 它们还可以设置和查询队列的属性。

消息排队的不同样式

点到点 (point-to-point)

一条消息放置在队列上，一个应用程序接收该消息。

在点到点消息传递中，发送应用程序必须先知道有关接收应用程序的信息，然后才能向该应用程序发送消息。 例如，发送应用程序可能需要知道要将信息发送到的队列的名称，还可能指定队列管理器名称。

发布/预订

由发布应用程序发布的每条消息的副本将传递到每个感兴趣的应用程序。 可能有许多，一个或没有感兴趣的应用程序。 在发布/预订中，感兴趣的应用程序称为订户，并且消息在由预订标识的队列上排队。

发布/预订消息传递允许您将信息提供者与该信息的使用者分离。 对于要发送和接收的信息，发送申请和接收申请不需要相互了解那么多。 有关更多信息，请参阅 第 168 页的『发布/预订消息传递』。

消息排队对应用程序设计者和开发者的好处

IBM MQ 允许应用程序使用 消息排队 来参与消息驱动的处理。 应用程序可以使用相应的消息排队软件产品在不同平台之间进行通信。 例如， z/OS 应用程序可以通过 IBM MQ for z/OS 进行通信。 应用程序与底层通信的机制相屏蔽。 消息排队的其他优点包括:

- 您可以使用可在许多应用程序之间共享的小程序来设计应用程序。
- 您可以通过复用这些构建块来快速构建新应用程序。
- 为使用消息排队技术而编写的应用程序不受队列管理器工作方式更改的影响。
- 您不需要使用任何通信协议。 队列管理器为您处理通信的所有方面。
- 在向其发送消息时，接收消息的程序无需运行。 这些消息保留在队列上。

设计人员可以降低其应用程序的成本，因为开发速度更快，需要的开发者更少，对编程技能的要求也比不使用消息排队的应用程序低。

IBM MQ 在应用程序运行的任何位置实现称为 消息队列接口 (或 MQI) 的公共应用程序编程接口。 这使您可以更轻松地将应用程序从一个平台移植到另一个平台。

有关 MQI 的详细信息，请参阅 消息队列接口概述。

消息排队的主要功能和优点

此信息突出显示了消息排队的一些功能和优点。 它描述了诸如消息排队的安全性和数据完整性之类的功能。

使用消息排队技术的应用程序的主要功能包括:

- 程序之间没有直接连接。
- 程序之间的通信可以独立于时间。
- 工作可以由小的，自成一体的程序来进行。
- 通信可以由事件驱动。
- 应用程序可以为消息分配优先级。
- 安全。
- 数据完整性。
- 恢复支持。

程序之间没有直接连接

消息排队是一种用于间接程序间通信的技术。 它可以在程序相互通信的任何应用程序中使用。 通信由一个将消息放入队列 (由队列管理器拥有) 的程序和另一个从队列中获取消息的程序进行。

程序可以获取由其他程序放入队列中的消息。 其他程序可以连接到与接收程序相同的队列管理器，也可以连接到其他队列管理器。 此另一个队列管理器可能位于另一个系统上，另一个计算机系统上，甚至位于另一个业务或企业内。

使用消息队列进行通信的程序之间没有物理连接。 程序将消息发送到队列管理器拥有的队列，而另一个程序从队列中检索消息 (请参阅 第 127 页的图 1)。

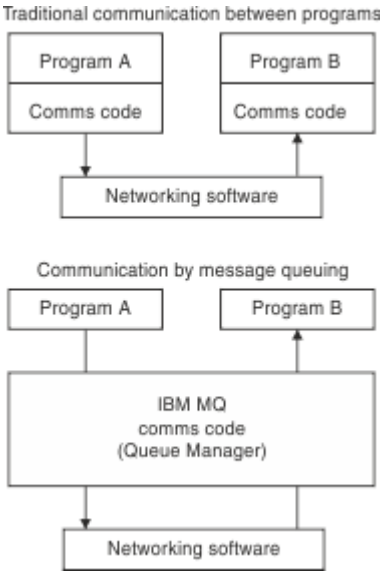


图 *1:* 与传统通信相比的消息排队

与电子邮件一样，作为事务一部分的个别消息通过商店上的网络传输。基础。 如果节点之间的链接失败，那么将保留该消息，直到恢复该链接，或者操作员或程序重定向该消息。

在程序中隐藏消息从队列移动到队列的机制。 因此，程序更简单。

独立于时间的通信

请求他人执行工作的程序不必等待对请求的应答。 他们可以执行其他工作，并在应答到达时或稍后处理该应答。 在编写消息传递应用程序时，当程序发送消息时，或者当目标能够接收消息时，您无需知道 (或关心)。 此消息不会丢失; 队列管理器将保留此消息，直到目标准备好对其进行处理为止。 消息将保留在队列中，直到程序将其除去。 这意味着发送和接收应用程序已解耦; 发送方可以继续处理，而无需等待接收方确认接收消息。 发送消息时，目标应用程序甚至不必正在运行。 它可以在消息启动后检索该消息。

小程序

消息排队允许您使用小型自包含程序的优点。 您可以将作业分布在几个较小的独立程序上，而不是单个大型程序按顺序执行作业的所有部分。 请求程序将消息发送到每个单独的程序，要求它们执行其功能; 当每个程序完成时，结果将作为一条或多条消息发送回。

消息驱动的处理

当消息到达队列时，它们可以使用 触发自动启动应用程序。 如果需要，可以在处理消息 (或消息) 时停止应用程序。

事件驱动的处理

可以根据队列的状态来控制程序。 例如，您可以安排程序在消息到达队列时立即启动，或者您可以指定程序在队列上具有高于特定优先级的 10 条消息或队列上具有任何优先级的 10 条消息之前不启动。

消息优先级

当程序将消息放入队列中时，它可以将优先级分配给消息。 这将确定添加新消息的队列中的位置。

程序可以按消息在队列中的顺序或通过获取特定消息从队列中获取消息。 (如果程序正在查找对其先前发送的请求的应答，那么该程序可能希望获取特定消息。)

安全性
提供了安全设施，包括当应用程序使用队列管理器时对其进行认证，当它们使用资源 (例如队列管理器上的队列) 时对其进行授权检查，以及当消息数据在网络上传输时对其进行加密，以及当消息数据驻留在队列上时对其进行加密。 有关安全性的更多信息，请参阅 安全性概述。

数据完整性
数据完整性由工作单元提供。 在每个 MQGET 或 MQPUT 上，完全支持将工作单元的开始和结束同步作为选项，从而允许落实或回滚工作单元的结果。 同步点支持在内部或外部针对 IBM MQ 运行，具体取决于为应用程序选择的同步点协调形式。

恢复支持
为了实现恢复，将记录所有持久 IBM MQ 更新。 如果需要恢复，那么将复原所有持久消息，回滚所有正在进行的事务，并以控制中的同步点管理器的正常方式处理任何同步点落实和回退。 有关持久消息的更多信息，请参阅 消息持久性。

注: 在考虑 IBM MQ 客户机和服务器时，您不必更改服务器应用程序以支持新平台上的其他 IBM MQ MQI clients 。 同样， IBM MQ MQI client 可以在不进行更改的情况下使用其他类型的服务器。

消息排队术语

此信息提供了对消息排队中使用的一些术语的洞察。

其中包括:

- 消息
- 消息描述符
- 队列
- 队列管理器
- 通道
- 消息通道代理程序
- 集群
- z/OS 共享队列
- z/OS 队列共享组
- z/OS 组内排队
- IBM MQ MQI client
- 点到点
- 发布/预订
- 主题
- 预订

消息

在消息排队中，消息是由一个程序发送并用于另一个程序的数据集合。 请参阅 IBM MQ 消息。 有关消息类型的信息，请参阅 消息类型。

消息描述符

IBM MQ 消息由控制信息和应用程序数据组成。

控制信息在消息描述符结构 (MQMD) 中定义，并包含如下内容:

- 消息类型
- 消息的标识
- 消息传递的优先级

应用程序数据的结构和内容由参与程序确定，而不是由 IBM MQ 确定。

队列

可将消息发送到的指定目标。 消息会在队列上累积，直到服务这些队列的程序检索到这些消息为止。

队列管理器

队列管理器 是向应用程序提供排队服务的系统程序。

它提供了一个应用程序编程接口，以便程序可以将消息放入队列并从队列中获取消息。 队列管理器提供了其他功能，以便管理员可以创建新队列，更改现有队列的属性以及控制队列管理器的操作。

要使 IBM MQ 消息排队服务在系统上可用，必须有一个正在运行的队列管理器。 您可以在单个系统上运行多个队列管理器 (例如，将测试系统与 实时 系统分开)。 对于应用程序，每个队列管理器由 连接句柄 (*Hconn*) 标识。

许多不同的应用程序可以同时使用队列管理器的服务，而这些应用程序可能完全不相关。 要使程序使用队列管理器的服务，它必须建立与该队列管理器的连接。

要使应用程序向连接到其他队列管理器的应用程序发送消息，队列管理器必须能够在它们之间进行通信。 IBM MQ 实现 *store-and-forward* 协议，以确保在此类应用程序之间安全地传递消息。

通道

通道 是提供从一个队列管理器到另一个队列管理器的通信路径的对象。 通道在分布式排队中用于将消息从一个队列管理器移动到另一个队列管理器，它们保护应用程序不受底层通信协议的保护。 队列管理器可能存在于相同或不同的平台上。

消息通道代理程序

消息通道代理程序是通道的一端。 一对消息通道代理 (一个发送和一个接收) 组成一个通道并将消息从一个队列管理器移动到另一个队列管理器。

集群

集群 是以某种方式逻辑关联的队列管理器网络。

在不使用集群的分布式排队的 IBM MQ 网络中，每个队列管理器都是独立的。 如果一个队列管理器需要将消息发送到另一个队列管理器，那么它必须已定义传输队列和到远程队列管理器的通道。

使用集群有两种不同的原因: 减少系统管理和提高可用性和工作负载均衡。

一旦建立了即使是最小的集群，您也会从简化的系统管理中获益。 属于集群的队列管理器需要较少的定义，因此减少了在定义中发生错误的风险。

有关集群的更多信息，请参阅 集群。

z/OS 共享队列

共享队列 是一种本地队列，其消息可由综合系统中的一个或多个队列管理器访问。 这与多个应用程序使用同一队列管理器共享的队列不同。 这仅适用于 IBM MQ for z/OS。

z/OS 队列共享组

可以访问同一组共享队列的队列管理器构成一个称为 队列共享组 (QSG) 的组。 它们通过用于存储共享队列的耦合设施 (CF) 相互通信。 这仅适用于 IBM MQ for z/OS。 有关队列共享组的更多信息，请参阅 第 291 页的『共享队列和队列共享组』 。

z/OS 组内排队

队列共享组中的队列管理器可以使用正常通道进行通信，也可以使用称为 组内排队 (IGQ) 的方法，这使您能够在不定义通道的情况下执行快速消息传输。 这仅适用于 IBM MQ for z/OS。

IBM MQ MQI client

IBM MQ MQI 客户机 是 IBM MQ 的独立可安装组件。 MQI 客户机允许您使用通信协议运行 IBM MQ 应用程序，与其他平台上的一个或多个消息队列接口 (MQI) 服务器进行交互并连接到其队列管理器。

有关如何安装和使用 IBM MQ MQI client 组件的完整详细信息，请参阅 安装 IBM MQ MQI client 和 配置服务器与客户机之间的连接。

点到点消息传递

在点到点消息传递中，每条消息从一个生产应用程序传输到一个消费应用程序。 通过将消息放入队列中的生产应用程序来传输消息，并且使用应用程序从该队列中获取消息。

发布/预订消息传递

在发布/预订消息传递中，由发布应用程序发布的每条消息的副本将传递到每个感兴趣的应用程序。 可能有许多，一个或不感兴趣的应用程序。 在发布/预订中，感兴趣的应用程序称为订户，并且消息在由预订标识的队列上排队。 有关更多信息，请参阅 第 168 页的『发布/预订消息传递』。

Topic

主题是描述在发布/预订消息中所发布信息的主题的字符串。

主题是发布/预订系统中成功发送消息的关键。 发布者为每条消息分配一个主题，而不是在每条消息中包含一个特定目标地址。 队列管理器使该主题与一组已预订该主题的订户匹配，并将消息传递至其中每个订户。

预订

发布/预订应用程序可以注册对有关特定主题的消息的兴趣。 当应用程序执行此操作时，它称为订户，术语预订定义匹配消息排队等待处理的方式。

预订包含有关订户的身份以及要将发布放置到的目标队列的身份的信息。 它还包含有关如何将发布放在目标队列上的信息。

消息和队列

消息和队列是消息排队系统的基本组件。

什么是消息?

消息 是对使用它的应用程序有意义的字节字符串。 消息用于将信息从一个应用程序传输到另一个应用程序(或者在同一应用程序的不同部分之间)。 应用程序可以在同一平台上运行，也可以在不同平台上运行。

IBM MQ 消息由以下内容组成:

- 应用程序数据。应用程序数据的内容和结构由使用它的应用程序定义。
- 消息描述符。消息描述符标识消息并包含其他控制信息，例如消息类型以及发送应用程序分配给消息的优先级。

  消息描述符的格式由 IBM MQ 定义。 有关消息描述符的完整描述，请参阅 MQMD-消息描述符。
- 消息属性。 有关消息的元数据。 消息属性的内容由使用它们的应用程序定义。 有关更多信息，请参阅 消息属性。

消息长度

缺省最大消息长度为 4 MB ，尽管您可以将其增大到最大长度 100 MB (其中 1 MB 等于 1 048 576 字节)。 在实践中，消息长度可能受以下限制:

- 为接收队列定义的最大消息长度
- 为队列管理器定义的最大消息长度
- 队列定义的最大消息长度
- 发送或接收应用程序定义的最大消息长度
- 可用于消息的存储量

可能需要多条消息才能发送应用程序所需的所有信息。

应用程序如何发送和接收消息?

应用程序使用 **MQI 调用**发送和接收消息。

例如，要将消息放入队列，应用程序:

1. 通过发出 MQI MQOPEN 调用打开所需队列
2. 发出 MQI MQPUT 调用以将消息放入队列

另一个应用程序可以通过发出 MQI MQGET 调用从同一队列中检索消息

有关 MQI 调用的更多信息，请参阅 MQI 调用。

什么是队列?

队列 是用于存储消息的数据结构。

每个队列都由 队列管理器拥有。 队列管理器负责维护它所拥有的队列，并负责将它接收到的所有消息存储到适当的队列中。 这些消息可由应用程序或队列管理器作为其正常操作的一部分放在队列上。

预定义队列和动态队列

可以通过创建队列的方式来描述这些队列:

- **预定义队列** 由管理员使用相应的 MQSC 或 PCF 命令创建。预定义队列是永久队列; 它们独立于使用它们的应用程序而存在，并且在 IBM MQ 重新启动后仍存在。
- **动态队列** 是在应用程序创建时创建的 发出 MQOPEN 请求，指定 模型队列的名称。 创建的队列基于称为模型队列的 模板队列定义。您可以使用 MQSC 命令 DEFINE QMODEL 来创建模型队列。 模型队列的属性 (例如，可存储在其上的最大消息数) 由从中创建的任何动态队列继承。

  模型队列具有一个属性，该属性指定动态队列是永久队列还是临时队列。 永久队列在应用程序和队列管理器重新启动后仍然存在; 临时队列在重新启动时丢失。

从队列中检索消息

适当授权的应用程序可以根据以下检索算法从队列中检索消息:

- 先进先出 (FIFO)。
- 消息优先级，如消息描述符中所定义。 将根据 FIFO 检索具有相同优先级的消息。
- 针对特定消息的程序请求。

来自应用程序的 MQGET 请求确定所使用的方法。

IBM MQ 对象

队列管理器定义 IBM MQ 对象的属性。 这些属性的值会影响 IBM MQ 处理这些对象的方式。 您可以使用 IBM MQ 命令和界面来创建和管理对象。 在应用程序中，使用消息队列接口 (MQI) 来控制对象。 从程序寻址时，对象由 IBM MQ 对象描述符 (MQOD) 标识。

对象管理包括以下任务:

- 启动和停止队列管理器。
- 为应用程序创建对象 (尤其是队列)。
- 显示或改变对象的属性。
- 正在删除对象。
- 使用通道来创建到其他 (远程) 系统上的队列管理器的通信路径。
- 创建队列管理器的 集群 以简化整体管理过程并平衡工作负载。

除了动态队列之外，必须先向队列管理器定义对象，然后才能使用这些对象。

使用 IBM MQ 命令执行对象管理操作时，队列管理器将检查您是否具有执行该操作所需的权限级别。 同样，当应用程序使用 MQOPEN 调用打开对象时，队列管理器将检查应用程序是否具有所需的权限级别，然后才能访问此对象。 针对要打开的对象的名称执行检查。

您可以使用以下方法来定义和管理对象:

- 可编程命令格式参考 和 自动化管理任务 中描述的 PCF 命令
- MQSC 命令 中描述的 MQSC 命令
- z/OS IBM MQ for z/OS 操作和控制面板，如 操作 IBM MQ for z/OS 中所述
- distributed IBM MQ Explorer (仅适用于 Intel 系统的 Windows 和 Linux)

您还可以使用以下方法来管理对象:

- 从键盘输入的控制命令。 请参阅 控制命令。
- IBM MQ 程序中的管理接口 (MQAI) 调用。 请参阅 IBM MQ 管理接口 (MQAI)。
- Windows 仅限 IBM MQ for Windows：
  - 程序中的 MQAI 组件对象模型 (COM) 调用
  - Windows 缺省配置应用程序

z/OS 对于您定期使用的 IBM MQ for z/OS 命令序列，您可以编写用于创建包含命令的消息并将这些消息放入系统命令输入队列的管理程序。 队列管理器处理此队列上的消息的方式与处理从命令行或操作和控制面板中输入的命令的方式相同。 此技术在 编写程序以管理 IBM MQ 中进行了描述，并在随 IBM MQ for z/OS 提供的 Mail Manager 样本应用程序中进行了演示。 有关此样本的描述，请参阅 IBM MQ for z/OS 的样本程序 。

IBM i 对于定期使用的 IBM MQ for IBMi 命令序列，可以编写 CL 程序。

distributed 对于 Windows UNIX 和 Linux 系统上的 IBM MQ 命令序列，您可以使用 MQSC 工具来运行保存在文件中的一系列命令。

对象类型

许多管理任务涉及处理各种类型的 IBM MQ 对象。

有关命名 IBM MQ 对象的信息，请参阅 第 146 页的『对 IBM MQ 对象进行命名』。

有关在队列管理器上创建的缺省对象的信息，请参阅 第 152 页的『系统缺省对象』。

有关不同类型的 IBM MQ 对象的信息，请参阅以下子主题:

相关概念

第 125 页的『消息排队简介』
IBM MQ 产品支持程序使用一致的应用程序编程接口在不同组件 (处理器，操作系统，子系统和通信协议) 的网络中相互通信。

第 151 页的『对象属性』
对象的属性由其属性定义。 某些可以指定，其他只能查看。

相关信息

MQSC 命令

队列

IBM MQ 队列和队列属性简介。

IBM MQ 队列 是一个指定的对象，应用程序可以在该对象上放置消息，并且应用程序可以从中获取消息。

消息存储在队列上，因此，如果放置应用程序期望对其消息进行应答，那么在等待该应答时可以执行其他工作。 应用程序使用消息队列接口 (MQI) 访问队列，如 消息队列接口概述中所述。

必须先创建队列，然后才能将消息放入队列中。 队列由队列管理器拥有，并且该队列管理器可以拥有许多队列。 但是，每个队列必须具有在该队列管理器中唯一的名称。

通过队列管理器维护队列。 在大多数情况下，每个队列都由其队列管理器进行物理管理，但这对于应用程序并不明显。 IBM MQ for z/OS 共享队列可以由队列共享组中的任何队列管理器管理。

要创建队列，可以使用 IBM MQ 命令 (MQSC)， PCF 命令或特定于平台的接口。 例如， IBM MQ for z/OS 操作和控制面板。

您可以从应用程序 动态 为临时作业创建本地队列。 例如，您可以创建 *reply-to* 队列 (在应用程序结束后不需要这些队列)。 有关更多信息，请参阅第 136 页的『动态队列和模型队列』。

在使用队列之前，必须打开队列，并指定要对其执行的操作。 例如，您可以打开以下对象的队列:

- 仅浏览消息 (不检索消息)
- 检索消息 (并与其他程序共享访问权或独占访问权)
- 将消息放入队列
- 查询队列的属性
- 设置队列的属性

要获取打开队列时可以指定的选项的完整列表，请参阅 MQOPEN-Open object。

队列的属性

队列的某些属性是在定义队列时指定的，之后无法更改 (例如，队列的类型)。 可以将队列的其他属性分组到可以更改的属性中:

- 在队列处理期间由队列管理器执行 (例如，队列的当前深度)
- 仅通过命令 (例如，队列的文本描述)
- 由应用程序使用 MQSET 调用 (例如，是否允许在队列上执行 put 操作)

您可以使用 MQINQ 调用来查找所有属性的值。

对于多种类型的队列通用的属性包括:

QName
队列的名称

QType
队列的类型

QDesc
队列的文本描述

InhibitGet
是否允许程序从队列获取消息 (尽管您永远无法从远程队列获取消息)

InhibitPut
是否允许程序将消息放入队列

DefPriority
放入队列的消息的缺省优先级

DefPersistence
放入队列中的消息的缺省持久性

Scope (在 z/OS 上不受支持)
控制此队列的条目是否也存在于名称服务中

有关这些属性的完整描述，请参阅 队列的属性。

相关概念

z/OS 共享队列
共享队列 是一种本地队列类型。 该队列上的消息可由综合系统中的一个或多个队列管理器访问。

第 165 页的『集群队列』
集群队列是由集群队列管理器托管并可供集群中其他队列管理器使用的队列。

第 158 页的『死信队列』
死信队列 (或未传递的消息队列) 是无法将消息路由到其正确目标时将消息发送到的队列。 每个队列管理器通常都有一个死信队列。

相关参考

第 165 页的『共享队列与集群队列之间的比较』
此信息旨在帮助您比较共享队列和集群队列，并确定哪些可能更适合您的系统。

相关信息

MQSC 命令

开发应用程序参考

本地队列

传输，启动，死信，命令，缺省，通道和事件队列是本地队列的类型。

如果队列归程序连接到的队列管理器所有，那么该队列对程序而言是本地队列。 您可以从本地队列中获取消息及将消息放在本地队列上。

队列定义对象保存队列的定义信息以及放在该队列上的物理消息。

每个队列管理器都可以有一些用于特殊目的的本地队列:

传输队列

当应用程序将消息发送到远程队列时，本地队列管理器将消息存储在称为 传输队列的特殊本地队列中。

消息通道代理程序 是与传输队列相关联的通道程序，它将消息传递到其下一个目标。 下一个目标是消息通道所连接到的队列管理器。 它不一定是与消息的最终目标相同的队列管理器。 将消息传递到其下一个目标时，将从传输队列中删除该消息。 该消息可能必须在到达其最终目标的过程中通过许多队列管理器。 您必须在路由的每个队列管理器上定义一个传输队列，每个队列管理器都保存等待传输到下一个目标的消息。 正常传输队列保存下一个目标的消息，尽管这些消息可能具有不同的最终目标。 集群传输队列保存多个目标的消息。 每条消息的 `correlID` 标识放置该消息以将其传输到其下一个目标的通道。

您可以在队列管理器上定义多个传输队列。 您可以为同一目标定义多个传输队列，每个传输队列用于不同的服务类。 例如，您可能想要为发送到同一目标的小消息和大消息创建不同的传输队列。 然后，您可以使用不同的消息通道来传输消息，以便大型消息不会容纳较小的消息。 缺省情况下，集群队列或集群主题的所有消息都放置在单集群传输队列 `SYSTEM.CLUSTER.TRANSMIT.QUEUE` 上。 作为一个选项，您可以更改缺省值，并将流向不同集群队列管理器的消息流量分隔到不同的集群传输队列上。 如果将队列管理器属性 `DEFCLXQ` 设置为 `CHANNEL`，那么每个集群发送方通道都会创建一个单独的集群传输队列。 作为替代方法，您可以手动定义集群传输队列以供集群发送方通道使用。

传输队列可以触发消息通道代理程序以继续发送消息; 请参阅 使用触发器启动 IBM MQ 应用程序。

z/OS 在 IBM MQ for z/OS 上，如果使用组内排队，那么传输队列由 组内排队代理提供服务。 在 IBM MQ for z/OS 上使用组内排队时，将使用共享传输队列。

启动队列

启动队列 是一个本地队列，当应用程序队列上发生触发器事件时，队列管理器在该队列上放置触发器消息。

触发器事件是旨在使程序开始处理队列的事件。 例如，事件可能超过 10 条消息到达。 有关触发如何工作的更多信息，请参阅 使用触发器启动 IBM MQ 应用程序。

死信（未传递的消息）队列

死信 (未传递的消息) 队列 是队列管理器将无法传递的消息放入其中的本地队列。

当队列管理器将消息放入死信队列时，它会向消息添加头。 头信息包括队列管理器将消息放入死信队列的原因。 它还包含原始消息的目标，日期以及队列管理器将消息放入死信队列的时间。

应用程序还可以将队列用于它们无法传递的消息。 有关更多信息，请参阅 使用死信 (未传递的消息) 队列。

系统命令队列

系统命令队列 是适当授权的应用程序可以向其发送 IBM MQ 命令的队列。 这些队列接收平台上支持的 PCF， MQSC 和 CL 命令，以便队列管理器能够对它们进行操作。

z/OS 在 IBM MQ for z/OS 上，队列称为 SYSTEM.COMMAND.INPUT ; 在其他平台上称为 SYSTEM.ADMIN.COMMAND.QUEUE。 接受的命令因平台而异。 请参阅可编程命令格式参考以获取详细信息。

系统缺省队列

系统缺省队列 包含系统的队列的初始定义。 创建队列定义时，队列管理器将从相应的系统缺省队列复制该定义。 创建队列定义与创建动态队列不同。 动态队列的定义基于您选择作为动态队列模板的模型队列。

事件队列

事件队列 用于保存事件消息。 这些消息由队列管理器或通道报告。

远程队列

对于程序，如果队列由与程序所连接的队列管理器不同的队列管理器拥有，那么该队列是 远程 队列。

在已建立通信链路的情况下，程序可以向远程队列发送消息。 程序永远无法从远程队列获取消息。

定义远程队列时创建的队列定义对象仅保存本地队列管理器查找您希望消息进入的队列所需的信息。 此对象称为 远程队列的本地定义。 远程队列的所有属性都由拥有它的队列管理器挂起，因为它是该队列管理器的本地队列。

打开远程队列时，要标识该队列，必须指定下列其中一项:

- 定义远程队列的本地定义的名称。

  要创建远程队列的本地定义，请使用 DEFINE QREMOTE 命令; 在 IBM MQ for IBM i 上，使用 CRTMQMQ 命令。

  从应用程序的角度来看，这与打开本地队列相同。 应用程序不需要知道队列是本地队列还是远程队列。
- 远程队列管理器的名称以及该远程队列管理器已知的队列的名称。

除了 第 133 页的『队列的属性』中描述的公共属性外，远程队列的本地定义还具有三个属性。 这些是 *RemoteQName* (队列的拥有队列管理器知道它的名称)， *RemoteQMgrName* (拥有队列管理器的名称) 和 *XmitQName* (将消息转发到其他队列管理器时使用的本地传输队列的名称)。 有关这些属性的更完整描述，请参阅 队列的属性。

如果对远程队列的本地定义使用 MQINQ 调用，那么队列管理器仅返回本地定义的属性，即远程队列名称，远程队列管理器名称和传输队列名称，而不是远程系统中匹配的本地队列的属性。

另请参阅 传输队列 (Transmission queue)。

别名队列

别名队列 是可用于访问另一个队列或主题的 IBM MQ 对象。 这意味着多个程序可以使用同一个队列，使用不同的名称访问该队列。

解析别名所生成的队列 (称为基本队列) 可以是平台支持的以下任何类型的队列:

- 本地队列
- 远程队列的本地定义。
- z/OS 它还可以是共享队列 (仅在 IBM MQ for z/OS 上可用的本地队列类型)。
- 预定义队列
- 动态队列

别名也可以解析为主题。 如果应用程序当前将消息放入队列，那么可以通过使队列名称成为主题的别名来将其发布到主题。 无需更改应用程序代码。

注: 别名不能直接解析为同一队列管理器上的另一个别名。

使用别名队列的一个示例是系统管理员对基本队列名称 (即，别名解析到的队列) 和别名队列名称授予不同的访问权限。 这意味着可以授权程序或用户使用别名队列，但不能使用基本队列。

或者，可以设置授权以禁止对别名执行放置操作，但允许对基本队列执行这些操作。

在某些应用程序中，使用别名队列意味着系统管理员可以轻松更改别名队列对象的定义，而不必更改应用程序。

当程序尝试使用别名时， IBM MQ 会针对别名进行授权检查。 它不会检查程序是否有权访问别名解析为的名称。 因此，可以授权程序访问别名队列名称，但不能访问已解析的队列名称。

除了 第 132 页的『队列』中描述的常规队列属性外，别名队列还具有 *BaseQName* 属性。 这是别名解析到的基本队列的名称。 有关此属性的更完整描述，请参阅 BaseQName (MQCHAR48)。

InhibitGet 和 *InhibitPut* 属性 (请参阅 第 132 页的『队列』) 别名队列属于别名。 例如，如果别名队列名称 ALIAS1 解析为基本队列名称 BASE，那么对 ALIAS1 的禁止仅影响 ALIAS1，并且不会禁止 BASE。 但是，禁止 BASE 也会影响 ALIAS1。

DefPriority 和 *DefPersistence* 属性也属于别名。 例如，您可以将不同的缺省优先级分配给同一基本队列的不同别名。 此外，您可以更改这些优先级，而不必更改使用别名的应用程序。

动态队列和模型队列

此信息可深入了解动态队列，临时和永久动态队列的属性，动态队列的使用，使用动态队列时的一些注意事项以及模型队列。

当应用程序发出 MQOPEN 调用以打开模型队列时，队列管理器会动态创建与模型队列具有相同属性的本地队列实例。 根据模型队列的 *DefinitionType* 字段的值，队列管理器将创建临时或永久动态队列 (请参阅 创建动态队列)。

临时动态队列的属性

临时动态队列 具有以下属性:

- 它们不能是共享队列，可从队列共享组中的队列管理器访问。

  请注意，队列共享组仅在 IBM MQ for z/OS 上可用。
- 它们仅保存非持久消息。
- 它们是不可恢复的。
- 当队列管理器启动时，将删除它们。
- 当发出创建队列的 MQOPEN 调用的应用程序关闭队列或终止时，将删除这些消息。
  - 如果队列上存在任何已落实的消息，那么将删除这些消息。
  - 如果此时有任何未落实的 MQGET， MQPUT 或 MQPUT1 调用未针对队列执行，那么该队列将标记为正在逻辑上删除，并且仅在落实这些调用后作为关闭处理的一部分进行物理删除，或者在应用程序终止时进行物理删除。
  - 如果此时该队列正在使用 (由创建或另一个应用程序)，那么该队列将标记为正在逻辑上被删除，并且仅当最后一个使用该队列的应用程序关闭时才会物理上被删除。
  - 尝试访问逻辑删除的队列 (而不是关闭该队列) 失败，原因码为 MQRC_Q_DELETED。
  - 在针对创建队列的相应 MQOPEN 调用的 MQCLOSE 调用上指定 MQCO_NONE 时， MQCO_DELETE 和 MQCO_DELETE_PURGE 都将被视为 MQCO_NONE。

永久动态队列的属性

永久动态队列 具有以下属性:

- 它们保存持久或非持久消息。
- 在发生系统故障时，它们是可恢复的。
- 当应用程序 (不一定是发出了创建队列的 MQOPEN 调用的应用程序) 使用 MQCO_DELETE 或 MQCO_DELETE_PURGE 选项成功关闭队列时，将删除这些消息。
  - 如果队列上仍有任何消息 (已落实或未落实)，那么带有 MQCO_DELETE 选项的关闭请求将失败。 带有 MQCO_DELETE_PURGE 选项的关闭请求成功，即使队列上存在已落实的消息 (要在关闭过程中删

除的消息)，但如果针对队列有任何未落实的 MQGET， MQPUT 或 MQPUT1 调用未完成，那么将失败。

  - 如果删除请求成功，但该队列恰好正在使用 (由创建应用程序或其他应用程序)，那么该队列将标记为逻辑上已删除，并且仅当最后一个使用该队列的应用程序关闭时才会实际删除该队列。

- 如果未被授权删除队列的应用程序关闭，那么不会将其删除，除非关闭应用程序发出了创建队列的 MQOPEN 调用。 将对用于验证相应 MQOPEN 调用的用户标识 (或备用用户标识 (如果指定了 MQOO_ALTERNATE_USER_AUTHORITY)) 执行授权检查。
- 可以使用与正常队列相同的方式删除这些队列。

动态队列的使用

您可以将动态队列用于:

- 在应用程序终止后不需要保留队列的应用程序。
- 需要由另一个应用程序处理对消息的回复的应用程序。 此类应用程序可以通过打开模型队列来动态创建应答队列。 例如，客户机应用程序可以:

  1. 创建动态队列。
  2. 在请求消息的消息描述符结构的 *ReplyToQ* 字段中提供其名称。
  3. 将请求放在服务器正在处理的队列上。

然后，服务器可以将应答消息放在应答队列上。 最后，客户机可以处理应答，并使用删除选项关闭应答队列。

使用动态队列时的注意事项

使用动态队列时，请考虑以下几点:

- 在客户机/服务器模型中，每个客户机都必须创建并使用其自己的动态应答队列。 如果在多个客户机之间共享动态应答队列，那么可能会延迟删除应答队列，因为针对该队列存在未落实的活动，或者该队列正由另一个客户机使用。 此外，该队列可能被标记为正在逻辑上被删除，并且对于后续 API 请求 (MQCLOSE 除外) 不可访问。
- 如果应用程序环境要求必须在应用程序之间共享动态队列，请确保仅当已落实针对队列的所有活动时才关闭该队列 (使用删除选项)。 这应该是最后一个用户。 这将确保不会延迟删除该队列，并将由于该队列已标记为已被逻辑删除而无法访问的时间段最小化。

模型队列

模型队列 是您在创建动态队列时使用的队列定义的模板。

您可以从 IBM MQ 程序动态创建本地队列，命名要用作队列属性模板的模型队列。 此时，您可以更改新队列的某些属性。 但是，您无法更改 *DefinitionType*。 例如，如果需要永久队列，请选择定义类型设置为永久的模型队列。 某些会话式应用程序可以使用动态队列来保存对其查询的应答，因为在处理这些应答之后，它们可能不需要维护这些队列。

在 MQOPEN 调用的 对象描述符 (MQOD) 中指定模型队列的名称。 通过使用模型队列的属性，队列管理器会动态地为您创建本地队列。

您可以指定动态队列的名称 (完整) 或名称的主干 (例如， ABC)，并让队列管理器为此添加唯一部分，也可以让队列管理器为您分配完整的唯一名称。 如果队列管理器指定名称，那么它会将其放入 MQOD 结构中。

不能直接向模型队列发出 MQPUT1 调用，但可以向已通过打开模型队列创建的动态队列发出 MQPUT1 。

无法对模型队列发出 MQSET 和 MQINQ。 使用 MQOO_INQUIRE 或 MQOO_SET 打开模型队列会导致对动态创建的队列进行后续 MQINQ 和 MQSET 调用。

模型队列的属性是本地队列的属性的子集。 有关更完整的描述，请参阅 队列的属性。

定义队列

您可以使用 MQSC 命令 DEFINE 或 PCF "创建队列" 命令将队列定义到 IBM MQ 。

这些命令指定队列的类型及其属性。 例如，本地队列对象具有一些属性，这些属性指定应用程序在 MQI 调用中引用该队列时发生的情况。 属性的示例包括:

- 应用程序是否可以从队列中检索消息 (已启用 GET)
- 应用程序是否可以将消息放入队列 (已启用 PUT)
- 对队列的访问是由一个应用程序独占还是在应用程序之间共享
- 可同时存储在队列上的最大消息数 (最大队列深度)
- 可以放到队列中的消息的最大长度

有关定义队列对象的更多详细信息，请参阅 脚本 (MQSC) 命令。

IBM MQ 使用的队列

IBM MQ 将某些本地队列用于与其操作相关的特定用途。

您必须先定义这些队列，然后 IBM MQ 才能使用这些队列。

启动队列

启动队列是在触发中使用的队列。 当发生触发器事件时，队列管理器将触发器消息放在启动队列上。 触发器事件是队列管理器检测到的条件的逻辑组合。 例如，当队列上的消息数达到预定义的深度时，可能会生成触发器事件。 此事件会导致队列管理器将触发器消息放在指定的启动队列上。 此触发器消息由触发器监视器(用于监视启动队列的特殊应用程序) 检索。 然后，触发器监视器启动触发器消息中指定的应用程序。

如果队列管理器要使用触发，那么必须至少为该队列管理器定义一个启动队列。 请参阅 管理用于触发的对象， runmqtrm 和 使用触发器启动 IBM MQ 应用程序

传输队列

传输队列是临时存储发往远程队列管理器的消息的队列。 必须为本地队列管理器要直接向其发送消息的每个远程队列管理器定义至少一个传输队列。 这些队列也用于远程管理; 请参阅 来自本地队列管理器的远程管理。 有关在分布式排队中使用传输队列的信息，请参阅 IBM MQ 分布式排队技术。

每个队列管理器都可以具有缺省传输队列。 如果不属于集群的队列管理器将消息放入远程队列，那么缺省操作是使用缺省传输队列。 如果存在与目标队列管理器同名的传输队列，那么消息将放置在该传输队列上。 如果存在队列管理器别名定义，其中 **RQMNAME** 参数与目标队列管理器匹配，并且指定了 **XMITQ** 参数，那么会将消息放在由 **XMITQ** 指定的传输队列上。 如果没有 **XMITQ** 参数，那么会将消息放在消息中指定的本地队列上。

集群传输队列

集群中的每个队列管理器都有一个名为 SYSTEM.CLUSTER.TRANSMIT.QUEUE 的集群传输队列和一个模型集群传输队列 SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE。 缺省情况下，在定义队列管理器时将创建这些队列的定义。 如果队列管理器属性 **DEFCLXQ** 设置为 CHANNEL，那么将自动为创建的每个集群发送方通道创建永久动态集群传输队列。 这些队列称为 SYSTEM.CLUSTER.TRANSMIT.*ChannelName*。 您还可以手动定义集群传输队列。

属于集群的队列管理器将其中一个队列上的消息发送到同一集群中的其他队列管理器。

在名称解析期间，集群传输队列优先于缺省传输队列，而特定集群传输队列优先于 SYSTEM.CLUSTER.TRANSMIT.QUEUE。

死信队列

死信 (undelivered-message) 队列是存储无法路由到其正确目标的消息的队列。 例如，当目标队列已满时，无法路由消息。 提供的死信队列称为 SYSTEM.DEAD.LETTER.QUEUE。

对于分布式排队，请在涉及的每个队列管理器上定义死信队列。

命令队列

命令队列 SYSTEM.ADMIN.COMMAND.QUEUE 是一个本地队列，适当授权的应用程序可以将 MQSC 命令发送到该队列以进行处理。 然后，这些命令将由称为命令服务器的 IBM MQ 组件检索。 命令服务器验证命令，传递有效的命令以供队列管理器处理，并将任何响应返回到相应的应答队列。

创建队列管理器时，将自动为每个队列管理器创建命令队列。

应答队列

当应用程序发送请求消息时，接收消息的应用程序可以将应答消息发送回发送应用程序。 此消息放在队列上，称为应答队列，通常是发送应用程序的本地队列。 应答队列的名称由发送应用程序指定为消息描述符的一部分。

事件队列

检测事件可用于独立于 MQI 应用程序监视队列管理器。

发生检测事件时，队列管理器会将事件消息放入事件队列中。 然后，监视应用程序可以读取此消息，这可能会通知管理员或在事件指示问题时启动一些补救操作。

注: 触发器事件与检测事件不同。 触发器事件不是由相同的条件引起的，并且不会生成事件消息。

有关检测事件的更多信息，请参阅 检测事件。

IBM MQ 队列管理器

队列管理器 及其提供给应用程序的排队服务简介。

程序必须具有与队列管理器的连接，然后才能使用该队列管理器的服务。 程序可以显式建立此连接 (使用 MQCONN 或 MQCONNX 调用) ，也可以隐式建立此连接 (这取决于运行程序的平台和环境)。

队列管理器向应用程序提供排队服务，并管理属于它们的队列。 队列管理器确保执行以下操作:

- 根据接收到的命令更改对象属性。
- 满足相应条件时，将生成特殊事件 (例如，触发器事件或检测事件)。
- 根据发出 MQPUT 调用的应用程序的请求，将消息放入正确的队列中。 如果无法执行此操作，那么将通知应用程序，并提供相应的原因码。

每个队列都属于单个队列管理器，并且被认为是该队列管理器的 本地队列 。 应用程序所连接的队列管理器被认为是该应用程序的 本地队列管理器 。 对于应用程序，属于其本地队列管理器的队列是本地队列。

远程队列 是属于另一个队列管理器的队列。 远程队列管理器 是除本地队列管理器以外的任何队列管理器。 远程队列管理器可以存在于网络中的远程机器上，也可以存在于本地队列管理器所在的机器上。 IBM MQ 支持同一机器上的多个队列管理器。

可以在某些 MQI 调用中使用队列管理器对象。 例如，您可以使用 MQI 调用 MQINQ 来查询队列管理器对象的属性。

队列管理器的属性

与每个队列管理器关联的是一组用于定义其特征的属性 (或属性)。队列管理器的某些属性在创建时是固定的; 您可以使用 IBM MQ 命令来更改其他属性。 您可以使用 MQINQ 调用来查询所有属性的值，但用于安全套接字层 (SSL) 加密的属性除外。

fixed 属性包括:

- 队列管理器的名称
- 运行队列管理器的平台 (例如， Windows)
- 队列管理器支持的系统控制命令的级别
- 可以分配给队列管理器处理的消息的最大优先级
- 程序可以向其发送 IBM MQ 命令的队列的名称
- 队列管理器可处理的最大消息长度 z/OS (仅在 IBM MQ for z/OS 中修正)
- 当程序放入和获取消息时，队列管理器是否支持同步

可更改 属性包括:

- 队列管理器的文本描述
- 队列管理器在处理 MQI 调用时用于字符串的字符集的标识
- 队列管理器用于限制触发器消息数的时间间隔
- z/OS 队列管理器用于确定队列扫描到期消息的频率的时间间隔 (仅限 IBM MQ for z/OS)
- 队列管理器的死信 (未传递的消息) 队列的名称
- 队列管理器的缺省传输队列的名称
- 任何一个连接的最大打开句柄数

- 启用和禁用各种类别的事件报告
- 工作单元中未落实的最大消息数

队列管理器和工作负载管理

您可以设置具有相同队列的多个定义的队列管理器集群 (例如，集群中的队列管理器可以相互克隆)。 特定队列的消息可由主管队列实例的任何队列管理器处理。 工作负载管理算法决定哪个队列管理器处理消息，从而在队列管理器之间传播工作负载; 请参阅 集群工作负载管理算法 以获取更多信息。

进程定义

进程定义对象允许通过定义应用程序的属性以供队列管理器使用来启动应用程序，而无需操作员干预。

进程定义对象定义用于响应 IBM MQ 队列管理器上的触发器事件而启动的应用程序。 进程定义属性包括应用程序标识，应用程序类型以及特定于应用程序的数据。 有关更多信息，请参阅 第 138 页的『IBM MQ 使用的队列』 下的 "启动队列" 条目。

要允许在不需要操作员干预的情况下启动应用程序 (如 使用触发器启动 IBM MQ 应用程序 中所述)，队列管理器必须知道应用程序的属性。 这些属性是在 进程定义对象中定义的。

ProcessName 属性在创建对象时是固定的; 您可以使用 IBM MQ 命令 z/OS 或 IBM MQ for z/OS 操作和控制面板来更改其他属性。

您可以使用 MQINQ-Inquire 对象属性来查询 所有 属性的值。

有关进程定义的属性的完整描述，请参阅 进程定义的属性。

名称列表

名称列表 是一个 IBM MQ 对象，其中包含集群名称，队列名称或认证信息对象名称的列表。 在集群中，它可用于标识队列管理器为其保存存储库的集群的列表。

名称列表是包含其他 IBM MQ 对象列表的 IBM MQ 对象。 通常，名称列表由诸如触发器监视器之类的应用程序使用，在那些应用程序中，名称列表用来标识一组队列。 使用名称列表的优点是它独立于应用程序进行维护; 可以在不停止使用它的任何应用程序的情况下进行更新。 此外，如果一个应用程序失败，那么名称列表不受影响，其他应用程序可以继续使用该名称列表。

名称列表还与队列管理器集群配合使用，以维护由多个 IBM MQ 对象引用的集群列表。

您可以使用 z/OS IBM MQ for z/OS 或 MQSC 命令的操作和控制面板来定义和修改名称列表。

程序可以使用 MQI 来查找这些名称列表中包含的队列。 名称列表的组织由应用程序设计者和系统管理员负责。

有关名称列表属性的完整描述，请参阅 名称列表的属性。

认证信息对象

队列管理器认证信息对象的简介以及指向更多信息的链接。

队列管理器认证信息对象构成对安全套接字层 (SSL) 和传输层安全性 (TLS) 的 IBM MQ 支持的一部分。 它提供了检查已撤销证书所需的定义。 认证中心撤销不再可信的证书。

本节描述如何将 `setmqaut`， `dspmqaut`， `dmpmqaut`， `rcrmqobj`， `rcdmqimg` 和 `dspmqfls` 命令与认证信息对象配合使用。 有关 SSL 和 TLS 以及使用认证信息对象的概述，请参阅 IBM MQ 中的 SSL 和 TLS 安全协议。

有关 SSL 和 TLS 的更多信息，请参阅 安全套接字层 (SSL) 和传输层安全性 (TLS) 概念。

认证信息对象提供执行证书撤销检查所需的定义。

有关认证信息对象的属性的完整描述，请参阅 认证信息对象。

通信信息对象

IBM MQ 多点广播提供低等待时间、高扇出和可靠的多点广播消息传递。 需要通信信息 (COMMINFO) 对象才能使用多点广播传输。

COMMINFO 对象是包含与多点广播传输关联的属性的 IBM MQ 对象。 有关这些属性的更多信息，请参阅 DEFINE COMMINFO。 有关创建 COMMINFO 对象的更多信息，请参阅 多点广播入门。

相关概念

第 205 页的『IBM MQ 多点广播』
IBM MQ 多点广播提供具有低等待时间和高扇出的可靠多点广播消息传递。

通道

通道 是分布式队列管理器使用的通信链路。

通道 是提供从一个队列管理器到另一个队列管理器的通信路径的对象。 通道在分布式排队中用于将消息从一个队列管理器移动到另一个队列管理器，它们保护应用程序不受底层通信协议的保护。 队列管理器可能存在于相同或不同的平台上。

要使队列管理器相互通信，必须在要发送消息的队列管理器上定义一个通道对象，在要接收消息的队列管理器上定义另一个补充对象。

IBM MQ 中有三类通道:

- 消息 通道 (单向) ，将消息从一个队列管理器传输到另一个队列管理器; 请参阅 通道-出口调用和数据结构 以获取更多信息。
- *MQI* 通道 (双向) ，用于将 MQI 调用从 IBM MQ MQI client 传输到队列管理器，并将响应从队列管理器传输到 IBM MQ 客户机; 请参阅 第 141 页的『什么是通道？』 以获取更多信息。
- *AMQP* 通道，这是双向通道，用于将 AMQP 客户机连接到服务器上的队列管理器。 IBM MQ 使用 AMQP 通道在 AMQP 应用程序和队列管理器之间传输 AMQP 调用和响应

通道定义

请参阅 第 142 页的『通道定义』 以获取每种类型的通道的描述。

相关概念

第 152 页的『分布式排队和集群』
分布式排队意味着将消息从一个队列管理器发送到另一个队列管理器。 接收队列管理器可以在同一台机器上，也可以在另一台机器上; 可以在附近，也可以在世界的另一侧。 它可以在与本地队列管理器相同的平台上运行，也可以在 IBM MQ 支持的任何平台上运行。 您可以手动定义分布式排队环境中的所有连接，也可以创建集群并让 IBM MQ 为您定义许多连接详细信息。

相关参考

第 144 页的『通信』
IBM MQ MQI clients 使用 MQI 通道与服务器通信。

相关信息

管理远程 IBM MQ 对象
通道出口调用和数据结构

什么是通道?

通道是 IBM MQ MQI client 与 IBM MQ 服务器之间或两个 IBM MQ 服务器之间的逻辑通信链路。

通道有两个定义: 连接的每个端都有一个定义。 在连接的每一端都必须使用相同的 通道名称 ，并且所使用的通道类型 必须兼容。

IBM MQ 中有两个类别的通道，这些类别中有不同的通道类型:

消息通道
消息通道是单向链接。 它使用 消息通道代理程序 (MCA) 连接两个队列管理器。

消息通道的用途是将消息从一个队列管理器传输到另一个队列管理器。 客户机服务器环境不需要消息通道。

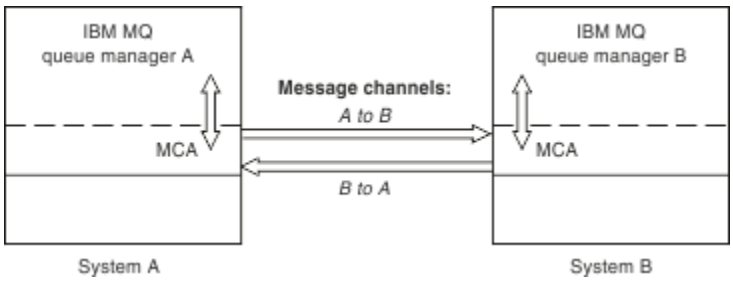


图 *2:* 两个队列管理器之间的消息通道

MQI 通道
消息队列接口 (MQI) 通道将 IBM MQ MQI client 连接到服务器上的队列管理器，并在您从 IBM MQ MQI client 应用程序发出 MQCONN 或 MQCONNX 调用时建立。

它是双向链接，仅用于传输 MQI 调用和响应，包括包含消息数据的 MQPUT 调用和导致返回消息数据的 MQGET 调用。 有不同的方法来创建和使用通道定义 (请参阅 定义 MQI 通道)。

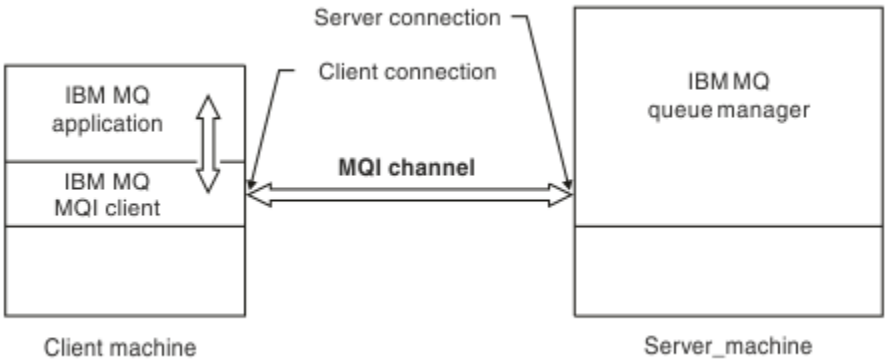


图 *3: MQI* 通道上的客户机连接和服务器连接

z/OS MQI 通道可用于将客户机连接到单个队列管理器，或连接到属于队列共享组的队列管理器 (请参阅 将客户机连接到队列共享组)。

MQI 通道定义有两种通道类型。 它们定义双向 MQI 通道。

客户机连接通道
此类型适用于 IBM MQ MQI client。

服务器连接通道
此类型适用于运行队列管理器的服务器，在 IBM MQ MQI client 环境中运行的 IBM MQ 应用程序将与该服务器进行通信。

相关信息
停止 MQI 通道
消息队列接口概述

通道定义
描述 IBM MQ 使用的不同类型的消息通道和 MQI 通道的表。

当提到消息通道时，通道一词常用作通道定义的同义词。 通常从上下文中能清楚地分辨出我们正在谈论的是一个完整的通道（带有两端）还是一个通道定义（只有一端）。

消息通道

消息通道定义可以是下列其中一个类型：

消息通道定义类型	描述
发送方	发送方通道是队列管理器用来将消息发送至其他队列管理器的消息通道。 要使用发送方通道来发送消息，还必须在另一队列管理器上创建与此发送方通道同名的接收方通道。 如果要实现“回叫”机制，那么还可以将发送方通道与请求方通道配合使用。
服务器	服务器通道是队列管理器用来将消息发送至其他队列管理器的消息通道。 要使用服务器通道来发送消息，还必须在另一队列管理器上创建与此服务器通道同名的接收方通道。 您还可以将服务器通道与请求方通道配合使用。 在此情况下，在通道另一端的请求方通道定义将请求要启动的服务器通道定义。 服务器会发送消息给请求方。 只要服务器知道对方通道的连接名称，它还可启动通信。
接收方	接收方通道是队列管理器用来从其他队列管理器接收消息的消息通道。 要使用接收方通道来接收消息，还必须在另一队列管理器上创建与此接收方通道同名的发送方通道或服务器通道。
请求者	请求方通道是队列管理器用来从其他队列管理器接收消息的消息通道。 请求方通道可以请求启动在远端定义的伙伴通道。 如果伙伴通道是服务器通道，那么服务器通道会接受启动请求，并开始从服务器通道定义中标识的传输队列向请求方通道发送消息。 如果伙伴通道是发送方通道，那么发送方通道会接受启动请求，然后关闭与请求方的连接。 然后，发送方通道开始与伙伴请求方通道协商会话，并开始从发送方通道定义中标识的传输队列发送消息。 后一种情况实质上提供了一种回调机制，其中请求方通道可以请求发送方通道进行回调。
集群发送方	集群发送方 (CLUSSDR) 通道定义用来定义通道发送端，集群队列管理器可通过它将集群信息发送至其中一个完整存储库。 集群发送方通道用于告知存储库有关队列管理器状态的任何更改，例如，添加或除去队列。 它还用于传输消息。 完整存储库队列管理器自身拥有指向彼此的集群发送方通道。 它们使用这些通道以就集群状态的更改进行相互通信。 队列管理器的 CLUSSDR 通道定义指向哪个完整存储库不是很重要。 在进行了最初的接触之后，会根据需要自动定义更多的集群队列管理器对象，以便队列管理器可将集群信息发送至每个完整存储库并将消息发送至每个队列管理器。
集群接收方	集群接收方 (CLUSRCVR) 通道定义用来定义通道接收端，集群队列管理器可通过它从集群中的其他队列管理器接收消息。 集群接收方通道还可传送有关集群的信息（发往存储库）。 通过定义集群接收方通道，该队列管理器对其他集群队列管理器表示它可用于接收消息。 每个集群队列管理器至少需要有一个集群接收方通道。

对于每个通道，您必须定义两端以便获取通道每一端的通道定义。 通道的两端必须是兼容类型。

您可使用下列通道定义的组合：

- 发送方-接收方
- 服务器-接收方
- 请求方-服务器
- 请求方-发送方（回调）

- 集群发送方-集群接收方

消息通道代理程序

您创建的每个通道定义都属于特定队列管理器。 队列管理器可具有同一类型或不同类型的几个通道。 通道的每一端是一个程序，即消息通道代理程序 (MCA)。 在通道的一端，调用方 MCA 从传输队列获取消息并通过通道发送它们。 在通道的另一端，响应方 MCA 接收这些消息并将它们传递至远程队列管理器。

调用方 MCA 可与发送方通道、服务器通道或请求方通道关联。 响应方 MCA 可与任何类型的消息通道关联。

IBM MQ 在连接的两端支持通道类型的以下组合：

调用方		**消息流方向**	**响应方**	
通道类型	**需要侦听器吗?**		**需要侦听器吗?**	**通道类型**
发送方	否	从调用方至响应方	Yes	接收方
服务器	否	从调用方至响应方	Yes	接收方
服务器	否	从调用方至响应方	Yes	请求者
请求者	否	从响应方至调用方	Yes	服务器
请求者	Yes	从响应方至调用方	Yes	发送方

MQI 通道

MQI 通道可以是下列其中一个类型：

MQI 通道类型	**描述**
服务器连接	服务器连接通道是双向 MQI 通道，用于将 IBM MQ 客户机连接至 IBM MQ 服务器。 服务器连接通道是通道的服务器端。
客户机连接	客户机连接通道是双向 MQI 通道，用于将 IBM MQ 客户机连接至 IBM MQ 服务器。 MQ Explorer 还使用客户机连接来连接至远程队列管理器。 客户机连接通道是通道的客户机端。 当您创建客户机连接通道时，在主管队列管理器的计算机上创建一个文件。 然后，您必须将该客户机连接文件复制到 IBM MQ 客户端计算机。

通信

IBM MQ MQI clients 使用 MQI 通道与服务器通信。

必须在连接的 IBM MQ MQI client 和服务器端创建通道定义。 定义 MQI 通道中说明了如何创建通道定义。

下表显示了可能的传输协议：

表 *6: MQI* 通道的传输协议				
客户机平台	**LU 6.2**	**TCP/IP**	**NetBIOS**	**SPX**
IBM i		Yes		
UNIX 和 Linux 系统	是 [1]	Yes		
Windows	Yes	Yes	Yes	Yes

注:

1. LU6.2 在 Linux (POWER 平台)， Linux (x86-64 平台)， Linux (zSeries s390x 平台) 或 Solaris (x86-64 平台) 上不受支持

传输协议- IBM MQ MQI client 和服务器平台的组合 显示了使用这些传输协议的 IBM MQ MQI client 和服务器平台的可能组合。

IBM MQ MQI client 上的 IBM MQ 应用程序可以使用与队列管理器为本地时相同的所有 MQI 调用。 **MQCONN** 或 **MQCONNX** 将 IBM MQ 应用程序与所选队列管理器相关联，从而创建 连接句柄。 然后，连接的队列管理器将处理使用该连接句柄的其他调用。 IBM MQ MQI client 通信需要客户机与服务器之间的活动连接，而队列管理器之间的通信是独立于连接和独立于时间的。

传输协议是使用通道定义指定的，不会影响应用程序。 例如， Windows 应用程序可以通过 TCP/IP 连接到一个队列管理器，并通过 NetBIOS 连接到另一个队列管理器。

性能注意事项

您使用的传输协议可能会影响 IBM MQ 客户机和服务器系统的性能。 对于通过慢速电话线的拨号支持，建议使用 IBM MQ 通道压缩。

客户机连接通道

客户机连接通道对象简介。

客户机连接通道 是提供从 IBM MQ MQI client 到队列管理器的通信路径的对象。 客户机连接通道用于分布式排队，以在队列管理器和客户机之间移动消息。 它们保护应用程序免受底层通信协议的保护。 客户机可能存在于与队列管理器相同的或不同的平台上。

存储类

存储类 将一个或多个队列映射到页集。

这意味着该队列的消息将存储在该页集上 (可进行缓冲)。 **仅在 IBM MQ for z/OS 上受支持**。

有关存储类的更多信息，请参阅 规划 z/OS。

侦听器

侦听器 是接受来自其他队列管理器或客户机应用程序的网络请求并启动关联通道的进程。

侦听器 是接受来自其他队列管理器或客户机应用程序的网络请求并启动关联通道的进程。 可以使用 `runmqlsr` 控制命令来启动侦听器进程。

侦听器对象 是 IBM MQ 对象，允许您在队列管理器的作用域内管理侦听器进程的启动和停止。 通过定义侦听器对象的属性，可以执行以下操作:

- 配置侦听器进程。
- 指定当队列管理器启动和停止时，侦听器进程是否自动启动和停止。

注: z/OS **侦听器对象在 IBM MQ for z/OS 上不受支持**。 有关 IBM MQ for z/OS 如何通过使用通道启动程序实现侦听的更多信息，请参阅 第 287 页的『z/OS 上的通道启动程序』。

服务

服务 对象是定义在队列管理器启动或停止时要运行的程序的一种方法。

z/OS **在 IBM MQ for z/OS 上不受支持**。 程序可以是下列其中一种类型:

服务器
服务器 是将参数 SERVTYPE 指定为 SERVER 的服务对象。 服务器服务对象是将在启动指定队列管理器时执行的程序的定义。 只能同时执行服务器进程的一个实例。 运行时，可以使用 MQSC 命令 DISPLAY SVSTATUS 来监视服务器进程的状态。 通常，服务器服务对象是程序 (例如，死信处理程序或触发器监视器) 的定义，但是可以运行的程序不限于 IBM MQ 随附的程序。 此外，可以定义服务器服务对象以包含在关闭指定队列管理器以结束程序时将运行的命令。

命令
命令 是将参数 SERVTYPE 指定为 COMMAND 的服务对象。 命令服务对象是将在启动或停止指定队列管理器时执行的程序的定义。 可以同时执行命令进程的多个实例。 命令服务对象与服务器服务对象不同，

因为一旦执行了程序，队列管理器将不会监视该程序。 通常，命令服务对象是生命周期较短的程序的定义，并且将执行特定任务 (例如，启动一个或多个其他任务)。

相关信息

使用服务

Topic 对象

主题对象 是一个 IBM MQ 对象，允许您将特定非缺省属性分配给主题。

主题 由发布或预订特定 主题字符串的应用程序定义。 主题字符串可以通过使用正斜杠字符 (/) 分隔主题来指定主题层次结构。 这可以通过 主题树进行可视化。 例如，如果应用程序发布到主题字符串 `/Sport/American Football` 和 `/Sport/Soccer`，那么将创建一个主题树，其中包含具有两个子代 `American Football` 和 `Soccer` 的父节点 `Sport` 。

主题从其主题树中找到的第一个父管理节点继承其属性。 如果特定主题树中没有管理主题节点，那么所有主题都将从基本主题对象 SYSTEM.BASE.TOPIC。

您可以在主题树中的任何节点上创建主题对象，方法是在主题对象的 TOPICSTR 属性中指定该节点的主题字符串。 您还可以为管理主题节点定义其他属性。 有关这些属性的更多信息，请参阅 MQSC 命令或 自动化管理任务。 缺省情况下，每个主题对象都将从其最接近的父管理主题节点继承其属性。

主题对象还可用于向应用程序开发者隐藏完整主题树。 如果为主题 `/Sport/American Football` 创建了名为 `FOOTBALL.US` 的主题对象，那么应用程序可以发布或预订名为 `FOOTBALL.US` 的对象，而不是具有相同结果的字符串 `/Sport/American Football` 。

如果在主题对象的主题字符串中输入 `#` ， `+` ，`/`或 `*` 字符，那么该字符将被视为字符串中的正常字符，并被视为与主题对象关联的主题字符串的一部分。

有关主题对象的更多信息，请参阅 第 168 页的『发布/预订消息传递』。

对 IBM MQ 对象进行命名

针对 IBM MQ 对象采用的命名约定取决于对象。 与 IBM MQ 配合使用的机器的名称和用户标识也受到一些命名限制。

队列管理器的每个实例都由其名称识别。 此名称在相互连接的队列管理器的网络中必须唯一，以便一个队列管理器可以毫不含糊地标识将任何给定消息发送到的目标队列管理器。

对于其他类型的对象，每个对象都有一个与其关联的名称，可以通过该名称引用该名称。 这些名称在一个队列管理器和对象类型中必须唯一。 例如，可以有一个队列和一个同名的进程，但不能有两个同名的队列。

在 IBM MQ 中，名称最多可以包含 48 个字符，但最多包含 20 个字符的 通道 除外。 有关命名 IBM MQ 对象的更多信息，请参阅 第 147 页的『用于命名 IBM MQ 对象的规则』。

与 IBM MQ 配合使用的机器的名称和用户标识也受到一些命名限制:

- 确保机器名不包含任何空格。 IBM MQ 不支持包含空格的机器名。 如果在此类机器上安装 IBM MQ ，那么无法创建任何队列管理器。
- 对于 IBM MQ 权限，用户标识和组的名称不得超过 20 个字符 (不允许使用空格)。
- 如果客户机以包含 @ 字符的用户标识 (例如， abc@d.) 运行，那么 IBM MQ for Windows 服务器不支持连接 Windows 客户机。

相关概念

第 149 页的『了解 IBM MQ 文件名』
每个 IBM MQ 队列管理器，队列，进程定义，名称列表，通道，客户机连接通道，侦听器，服务和认证信息对象都由一个文件表示。 由于对象名不一定是有效的文件名，因此队列管理器会在必要时将对象名转换为有效的文件名。

相关参考

第 147 页的『用于命名 IBM MQ 对象的规则』
IBM MQ 对象名具有最大长度并且区分大小写。 并非每个对象类型都支持所有字符，并且许多对象都有关于名称唯一性的规则。

用于命名 IBM MQ 对象的规则

IBM MQ 对象名具有最大长度并且区分大小写。 并非每个对象类型都支持所有字符，并且许多对象都有关于名称唯一性的规则。

有许多不同类型的 IBM MQ 对象，并且每种类型的对象都可以具有相同的名称，因为它们存在于不同的对象名称空间中: 例如，本地队列和发送方通道都可以具有相同的名称。 但是，一个对象不能与同一名称空间中的另一个对象具有相同的名称：例如，本地队列不能与模型队列具有相同的名称，发送方通道不能与接收方通道具有相同的名称。

以下 IBM MQ 对象存在于单独的对象名称空间中:

- 认证信息
- 通道
- 客户机通道
- 侦听器
- 名称列表
- 进程
- 队列
- 服务
- 存储类
- 预订
- Topic

对象名称的字符长度

通常， IBM MQ 对象名的长度最多可以为 48 个字符。 此规则适用于以下对象:

- 认证信息
- 集群
- 侦听器
- 名称列表
- 进程定义
- 队列
- 队列管理器
- 服务
- 预订
- Topic

存在以下限制:

1. z/OS 在 z/OS 系统上，队列管理器必须最多为 4 个字符，并且必须仅为大写字符和数字字符。
2. 通道对象名和客户机连接通道名的最大长度为 20 个字符。 请参阅 定义通道 以获取有关通道的更多信息。
3. 主题字符串最多可以为 10240 个字节。 所有 IBM MQ 对象名都区分大小写。
4. 存储类名的最大长度为 8 个字符。
5. CF 结构名称的最大长度为 12 个字符。

对象名称中的字符

IBM MQ 对象名的有效字符为:

个字符之后	限制
大写 A-Z	• None
小写 a-z	• 在 MQSC 脚本中，必须将带有小写字符的名称括在单引号中。 这将防止将小写字符转换为大写。 • 使用 EBCDIC 片假名的系统不能在对象名中使用小写 a-z 字符。 • z/OS 在 z/OS 系统上使用小写字符时可能存在限制，例如，队列管理器名称不能包含小写字符。 • IBM i 在 IBM i 系统上使用 CL 命令时，必须将带有小写字符的名称括在单引号中。 这将防止将小写字符转换为大写。
数字 0-9	• None
句点 (.)	• None
下划线 (_)	• IBM i distributed None • z/OS 避免使用带有前导或尾部下划线的名称，因为 IBM MQ for z/OS 操作和控制面板无法处理这些名称。
正斜杠 (/)	• Windows 在 Windows 系统上，队列管理器名称的第一个字符不能是正斜杠。 • IBM i 在 IBM i 系统上使用 CL 命令时，必须将包含正斜杠的名称括在单引号中。 • z/OS None
百分号 (%)	• distributed None • z/OS 如果要将 RACF 用作 IBM MQ for z/OS 的外部安全性管理器，请不要在对象名中使用%，因为在使用 RACF 通用概要文件时，这些名称不会包含在安全性检查中。 • IBM i 在 IBM i 系统上使用 CL 命令时，必须将包含百分号的名称括在单引号中。

还有一些关于对象名上的字符的一般规则:

1. 不允许前导空白或嵌入空白。
2. 不允许使用本地语言字符。
3. 任何小于完整字段长度的名称都可以用空格填充到右边。 队列管理器返回的所有短名称总是用空格填充到右边。

队列名称

队列的名称有两个部分:

- 队列管理器的名称

- 队列管理器已知的队列的局部名

队列名称的每个部分都有 48 个字符长。

要引用本地队列，可以省略队列管理器的名称 (通过将其替换为空白字符或使用前导空字符)。 但是，IBM MQ 返回到程序的所有队列名称都包含队列管理器的名称。

z/OS 共享队列 (可供其队列共享组中的任何队列管理器访问) 不能与同一队列共享组中的任何非共享本地队列同名。 此限制避免了应用程序在打算打开本地队列时误打开共享队列的可能性，反之亦然。 共享队列和队列共享组仅在 IBM MQ for z/OS 上可用。

要引用远程队列，程序必须在完整队列名称中包含队列管理器的名称，或者必须存在远程队列的本地定义。

当应用程序使用队列名称时，该名称可以是本地队列的名称 (或一个队列的别名) ，也可以是远程队列的本地定义的名称，但应用程序不需要知道哪个，除非它需要从队列中获取消息 (当队列必须是本地队列时)。 当应用程序打开队列对象时， MQOPEN 调用将执行名称解析函数以确定要在哪个队列上执行后续操作。 这一点的重要意义在于，应用程序不依赖于在队列管理器网络中的特定位置定义的特定队列。 因此，如果系统管理员在网络中重新定位队列并更改其定义，那么不需要更改使用这些队列的应用程序。

z/OS **保留对象名**

在 IBM MQ for z/OS 上，保留耦合设施应用程序结构名称 CSQSYSAPPL。

保留对象名

以 SYSTEM 开头的对象名。 为队列管理器定义的对象保留。 您可以使用 **Alter**， **Define** 和 **Replace** 命令来更改这些对象定义以适合您的安装。 为 IBM MQ 定义的名称在 队列名称中完整列出。

z/OS 在 IBM MQ for z/OS 上，保留耦合设施应用程序结构名称 CSQSYSAPPL。

相关信息

选择安装名称

了解 IBM MQ 文件名

每个 IBM MQ 队列管理器，队列，进程定义，名称列表，通道，客户机连接通道，侦听器，服务和认证信息对象都由一个文件表示。 由于对象名不一定是有效的文件名，因此队列管理器会在必要时将对象名转换为有效的文件名。

队列管理器目录的缺省路径如下所示:

- 在 IBM MQ 配置信息中定义的前缀:
  - Windows 在 Windows 32 位系统上，缺省前缀为 `C:\Program Files (x86) \WebSphere MQ`。 在 Windows 64 位系统上，缺省前缀为 `C:\Program Files\IBM\WebSphere MQ`。 对于 32 位和 64 位安装，数据目录将安装到 `C:\ProgramData\IBM\MQ` 中。 这是在 mqs.ini 配置文件的 `DefaultPrefix` 节中配置的。
  - UNIX 在 UNIX 和 Linux 系统上，缺省前缀为 `/var/mqm`。 这是在 mqs.ini 配置文件的 `DefaultPrefix` 节中配置的。

  如果可用，可以使用 IBM MQ Explorer 中的 IBM MQ 属性页面来更改前缀，否则请手动编辑 `mqs.ini` 配置文件。
- 队列管理器名称将变换为有效的目录名称。 例如，队列管理器:

```
queue.manager
```

将表示为:

```
queue!manager
```

此过程称为 名称变换。

在 IBM MQ 中，可以为队列管理器提供最多包含 48 个字符的名称。

例如，可以命名队列管理器:

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

但是，每个队列管理器都由一个文件表示，并且对文件名的最大长度以及名称中可以使用的字符都有限制。因此，将自动变换表示对象的文件的名称以满足文件系统的要求。

管理队列管理器名称变换的规则如下所示:

1. 变换单个字符:
   - 从。 到!
   - From / to &
2. 如果名称仍然无效:
   a. 将其截断为 8 个字符
   b. 附加三字符数字后缀

例如，假定缺省前缀和名为 `queue.manager` 的队列管理器:

- Windows 在具有 NTFS 或 FAT32 的 IBM MQ for Windows 中，队列管理器名称变为:

  ```
  C:\Program Files\IBM\WebSphere MQ\qmgrs\queue!manager
  ```

- Windows 在具有 FAT 的 IBM MQ for Windows 中，队列管理器名称变为:

  ```
  C:\Program Files\IBM\WebSphere MQ\qmgrs\queue!ma
  ```

- Linux UNIX 在 IBM MQ for UNIX 和 Linux 系统中，队列管理器名称变为:

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

变换算法还区分仅在不区分大小写的文件系统上不同的名称。

对象名变换

对象名不一定是有效的文件系统名。 您可能需要变换对象名。 所使用的方法与队列管理器名称的方法不同，因为虽然每台机器上只有几个队列管理器名称，但每个队列管理器可能有大量其他对象。 队列，进程定义，名称列表，通道，客户机连接通道，侦听器，服务和认证信息对象在文件系统中表示。

当变换过程生成新名称时，与原始对象名没有简单关系。 您可以使用 `dspmqfls` 命令在实对象名和已变换对象名之间进行转换。

了解 IBM MQ for IBM i 队列管理器库名

使用此信息可了解库名， IFS 目录和名称变换。

创建队列管理器时， IBM MQ 会将队列管理器库与其相关联。 此队列管理器库具有唯一名称，长度不超过 10 个字符，这在很大程度上取决于用户定义的队列管理器名称。 队列管理器和队列管理器库都放在一个目录中，该目录也基于具有前缀 `/QIBM/UserData/mqm` 的队列管理器名称。 以下是队列管理器，队列管理器库和目录的示例:

队列管理器名称	橙色
队列管理器库名	Qmorange

目录	/QIBM/UserData/mqm/ORANGE

所有队列管理器名称和队列管理器库名都将写入文件 /QIBM/UserData/mqm/mqs.ini 中的节。

了解 IBM MQ IFS 目录和文件

IBM i Integrated File System (IFS) 由 IBM MQ 广泛用于存储数据。 有关 IFS 的更多信息，请参阅 *Integrated File System* 简介。

每个 IBM MQ 对象 (例如: 通道或队列管理器) 都由一个文件表示。 由于对象名不一定是有效的文件名，因此队列管理器会在必要时将对象名转换为有效的文件名。

队列管理器目录的路径由以下内容构成:

- 在队列管理器配置文件 qm.ini 中定义的前缀。 缺省前缀为 /QIBM/UserData/mqm。
- 文字 qmgrs。
- 编码的队列管理器名称，即转换为有效目录名的队列管理器名称。 例如，队列管理器 queue/manager 由 queue&manager 表示。

此过程称为名称变换。

IFS 队列管理器名称变换

在 IBM MQ 中，可以为队列管理器提供最多包含 48 个字符的名称。

例如，可以将队列管理器命名为 QUEUE/MANAGER/ACCOUNTING/SERVICES。 与为每个队列管理器创建库的方式相同，每个队列管理器也由一个文件表示。 由于 EBCDIC 中的变体代码点，因此可以在名称中使用的字符存在限制。 因此，表示对象的 IFS 文件的名称会自动变换以满足文件系统的要求。

使用名称为 queue/manager 的队列管理器示例，将字符 / 转换为 &，并采用缺省前缀， IBM MQ for IBM i 中的队列管理器名称将变为 /QIBM/UserData/mqm/qmgrs/queue&manager。

对象名变换

对象名不一定是有效的文件系统名，因此可能需要变换对象名。 所使用的方法与队列管理器名称的方法不同，因为虽然每台机器只有几个队列管理器名称，但每个队列管理器可能有大量其他对象。 只有进程定义，队列和名称列表在文件系统中表示; 通道不受这些注意事项影响。

当变换过程生成新名称时，与原始对象名没有简单关系。 可以使用 DSPMQMOBJN 命令来查看 IBM MQ 对象的已变换名称。

对象属性

对象的属性由其属性定义。 某些可以指定，其他只能查看。

例如，队列可容纳的最大消息长度由其 *MaxMsgLength* 属性定义; 您可以在创建队列时指定此属性。 *DefinitionType* 属性指定如何创建队列; 您只能显示此属性。

在 IBM MQ 中，有两种引用属性的方法:

- 使用其 PCF 名称，例如 *MaxMsgLength*。
- 使用其 MQSC 命令名，例如 MAXMSGL。

本指南主要描述了如何使用 MQSC 命令指定属性，因此它引用了大多数使用其 MQSC 命令名称的属性，而不是其 PCF 名称。

队列共享组

可访问同一组共享队列的队列管理器构成一个称为 队列共享组 (QSG) 的组，它们使用存储共享队列的耦合设施 (CF) 相互通信。

共享队列 是具有可由队列共享组中的一个或多个队列管理器访问的消息的本地队列类型。 **仅在 IBM MQ for z/OS 上受支持**。 (这与多个应用程序使用同一队列管理器 共享 的队列不同。)

队列共享组具有最多为四个字符的名称。 该名称在网络中必须是唯一的，并且必须与所有队列管理器名称不同。

队列共享组不是严格的对象，而是在此处提及以方便使用。

有关更多信息，请参阅第 291 页的『共享队列和队列共享组』。

系统缺省对象

系统缺省对象的简介以及指向更多信息的链接。

系统缺省对象 是在创建队列管理器时自动创建的一组对象定义。 您可以复制和修改这些对象定义中的任何对象定义，以便在安装时在应用程序中使用。

缺省对象名具有主干 SYSTEM; 例如，缺省本地队列为 SYSTEM.DEFAULT.LOCAL.QUEUE，缺省接收方通道为 SYSTEM.DEF.RECEIVER。 您无法重命名这些对象; 需要这些名称的缺省对象。

定义对象时，将从相应的缺省对象复制未显式指定的任何属性。 例如，如果定义本地队列，那么将从缺省队列 SYSTEM.DEFAULT.LOCAL.QUEUE。

有关系统缺省值的更多信息，请参阅 系统和缺省对象 。

分布式排队和集群

分布式排队意味着将消息从一个队列管理器发送到另一个队列管理器。 接收队列管理器可以在同一台机器上，也可以在另一台机器上; 可以在附近，也可以在世界的另一侧。 它可以在与本地队列管理器相同的平台上运行，也可以在 IBM MQ 支持的任何平台上运行。 您可以手动定义分布式排队环境中的所有连接，也可以创建集群并让 IBM MQ 为您定义许多连接详细信息。

分布式 排队

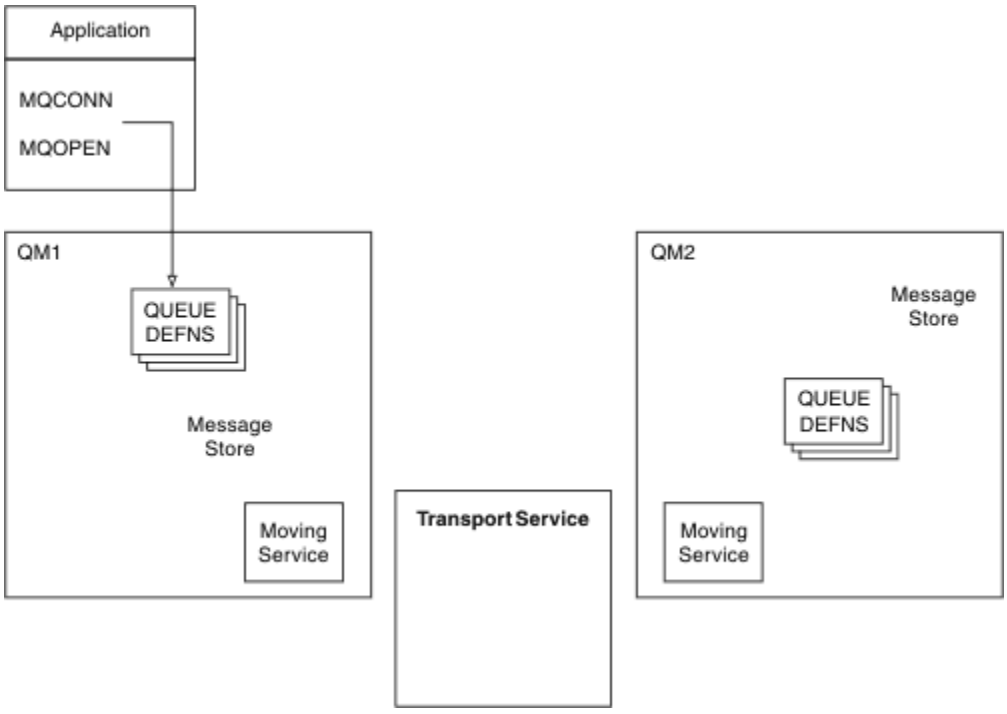


图 *4:* 分布式排队组件概述

在上图中:

- 应用程序使用 MQCONN 调用来连接到队列管理器。 然后，应用程序使用 MQOPEN 调用来打开队列，以便它可以将消息放入队列中。
- 每个队列管理器都有其每个队列的定义。 它可以具有 本地队列 (即，由此队列管理器托管) 的定义以及 远程队列 (即，由其他队列管理器托管) 的定义。

- 如果消息以远程队列为目标，那么本地队列管理器会将它们保存在 传输队列上，该队列会将它们持久存储在消息存储器中，直到可以将它们转发到远程队列管理器为止。
- 每个队列管理器都包含通信软件 (称为 移动服务) ，队列管理器使用这些软件与其他队列管理器进行通信。
- 传输服务 独立于队列管理器，可以是下列任何一项 (取决于平台):
  - 系统网络体系结构高级程序间通信 (SNA APPC)
  - 传输控制协议/因特网协议 (Transmission Control Protocol/Internet Protocol, TCP/IP)
  - 网络基本输入/输出系统 (NetBIOS)
  - 顺序包交换 (SPX)

发送消息所需的组件

如果要将消息发送到远程队列管理器，那么本地队列管理器需要 传输队列 和 通道的定义。 通道是两个队列管理器之间的单向通信链路。 它可以携带发往远程队列管理器中任意数量的队列的消息。

通道的每个端都具有单独的定义，例如，将其定义为发送端或接收端。 简单通道由本地队列管理器上的 发送方 通道定义和远程队列管理器上的 接收方 通道定义组成。 这两个定义必须具有相同的名称，并且它们共同构成一个通道。

用于处理消息发送和接收的软件称为 消息通道代理程序 (MCA)。 在通道的每一端都有一个 消息通道代理程序 (MCA)。

每个队列管理器都应该具有 死信队列 (也称为 未传递的消息队列)。 如果无法将消息传递到其目标，那么会将这些消息放入此队列。

下图显示了队列管理器，传输队列，通道和 MCA 之间的关系:

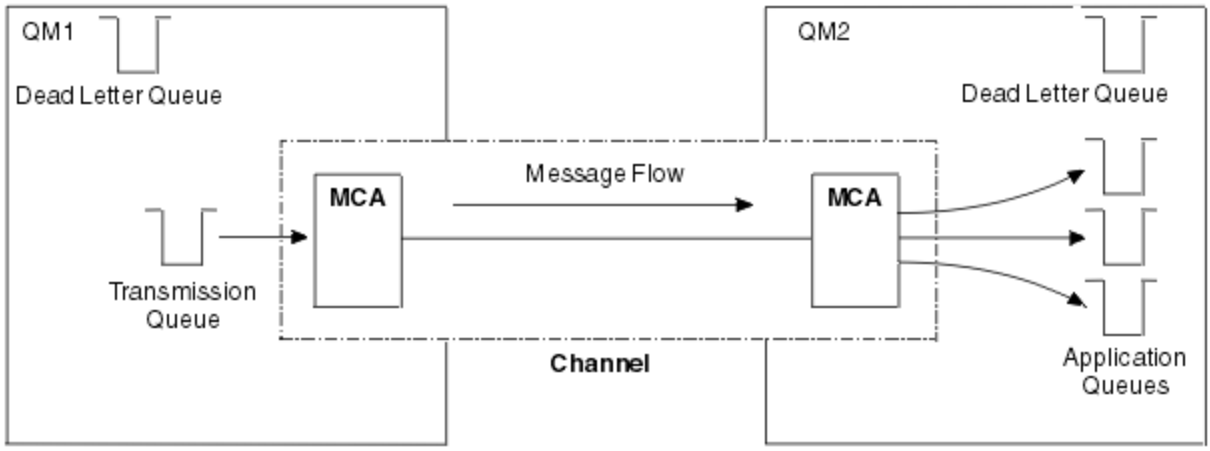


图 *5:* 发送消息

返回消息所需的组件

如果应用程序要求从远程队列管理器返回消息，那么需要定义另一个通道，以便在队列管理器之间以相反的方向运行，如下图所示:

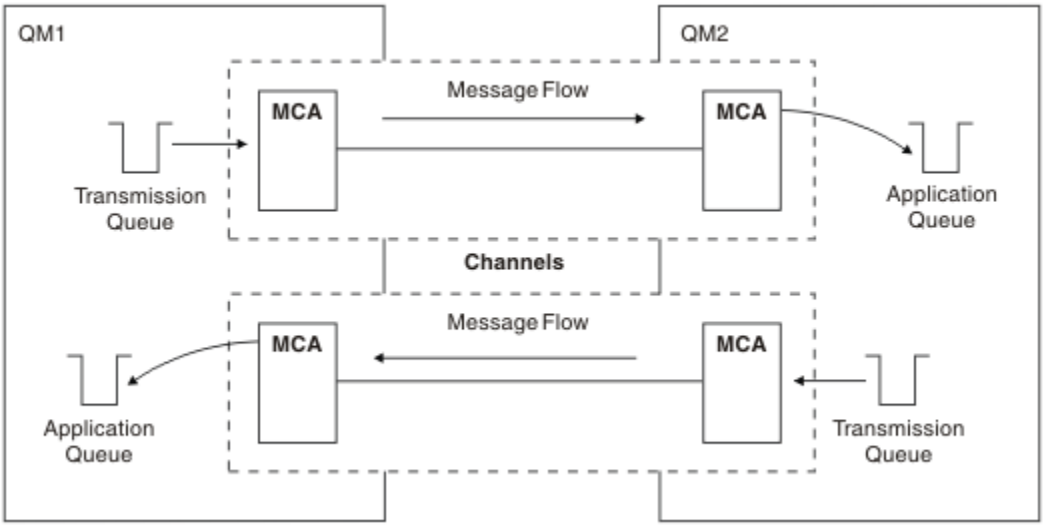


图 *6:* 双向发送消息

集群

您可以在集群中对一组队列管理器进行分组，而不是手动定义分布式排队环境中的所有连接。 执行此操作时，队列管理器可以使其托管的队列可供集群中的其他队列管理器使用，而无需针对每个目标的显式通道定义，远程队列定义或传输队列。 集群中的每个队列管理器都有一个传输队列，用于将消息传输到集群中的任何其他队列管理器。 对于每个队列管理器，您只需要定义一个集群接收方通道和一个集群发送方通道; 任何其他通道都由集群自动管理。

IBM MQ 客户机可以连接到属于集群的队列管理器，就像它可以连接到任何其他队列管理器一样。 与手动配置的分布式排队一样，您使用 MQPUT 调用将消息放入任何队列管理器中的队列。 您可以使用 MQGET 调用从本地队列中检索消息。

支持集群的平台上的队列管理器不必是集群的一部分。 您可以继续手动配置分布式排队，也可以使用集群来代替分布式排队。

使用集群的优点

集群提供了两个关键优势:

- 集群简化了 IBM MQ 网络的管理，通常需要为要配置的通道，传输队列和远程队列提供许多对象定义。 在许多队列管理器需要互连的大型网络中，尤其存在这种情况。 这种架构特别难以配置和积极维护。
- 集群可用于在集群中的队列和队列管理器之间分配消息流量的工作负载。 此类分发允许将单个队列的消息工作负载分布到位于多个队列管理器上的该队列的等效实例中。 工作负载的这种分布可用于实现对系统故障的更大弹性，以及提高系统中特别活跃的消息流的缩放性能。 在这种环境中，分布式队列的每个实例都具有处理消息的使用应用程序。 有关更多信息，请参阅 使用集群进行工作负载管理。

如何在集群中路由消息

您可以将集群视为由良心系统管理员维护的队列管理器网络。 无论何时定义集群队列，系统管理员都会根据需要在其他队列管理器上自动创建相应的远程队列定义。

您不需要进行传输队列定义，因为 IBM MQ 在集群中的每个队列管理器上提供了传输队列。 此单个传输队列可用于将消息传递到集群中的任何其他队列管理器。 您不限于使用单个传输队列。 队列管理器可以使用多个传输队列来分隔发送到集群中每个队列管理器的消息。 通常，队列管理器使用单个集群传输队列。 您可以更改队列管理器属性 DEFCLXQ，以便队列管理器对集群中的每个队列管理器使用不同的集群传输队列。 您还可以手动定义集群传输队列。

所有加入集群的队列管理器都同意以这种方式工作。 他们发送有关自己和他们托管的队列的信息，并接收有关集群的其他成员的信息。

要确保在队列管理器变为不可用时不会丢失任何信息，请在集群中指定两个队列管理器以充当 完整存储库。 这些队列管理器存储有关集群中所有队列管理器和队列的完整信息集。 集群中的所有其他队列管理器仅存储

有关这些队列管理器及其交换消息的队列的信息。 这些队列管理器称为 部分存储库。 有关更多信息，请参阅第 164 页的『集群存储库』。

要成为集群的一部分，队列管理器必须具有两个通道: 集群发送方通道和集群接收方通道:

- 集群发送方通道是类似于发送方通道的通信通道。 必须在队列管理器上手动创建一个集群发送方通道，以将其连接到已是集群成员的完整存储库。
- 集群接收方通道是类似于接收方通道的通信通道。 必须手动创建一个集群接收方通道。 通道充当队列管理器接收集群通信的机制。

然后，将自动创建此队列管理器与集群的其他成员之间的通信所需的所有其他通道。

下图显示了名为 CLUSTER 的集群的组件:

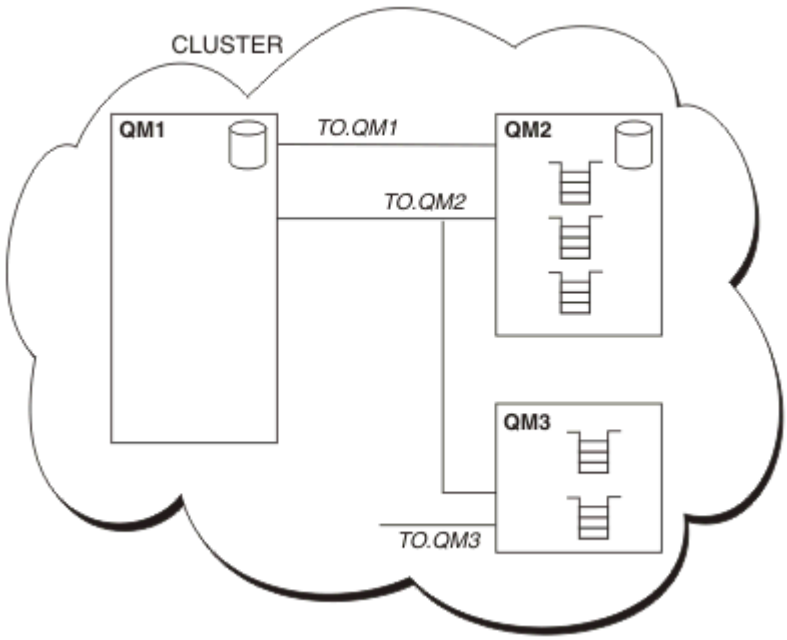


图 7: 队列管理器集群

- CLUSTER 包含三个队列管理器: QM1， QM2 和 QM3。
- QM1 和 QM2 托管有关集群中的队列管理器和队列的信息的完整存储库。
- QM2 和 QM3 托管一些集群队列，即可供集群中任何其他队列管理器访问的队列。
- 每个队列管理器都有一个名为 TO.qmgr 的集群接收方通道，可以在该通道上接收消息。
- 每个队列管理器还具有一个集群发送方通道，在该通道上，它可以将信息发送到其中一个存储库队列管理器。
- QM1 和 QM3 发送到 QM2 上的存储库， QM2 发送到 QM1 上的存储库。

分布式排队组件

分布式排队的组件包括消息通道，消息通道代理程序，传输队列，通道启动器和侦听器以及通道出口程序。 消息通道的每个端的定义可以是多种类型之一。

消息通道是将消息从一个队列管理器传递到另一个队列管理器的通道。 请勿将消息通道与 MQI 通道混淆。 有两种类型的 MQI 通道: 服务器连接 (SVRCONN) 和客户机连接 (CLNTCONN)。 有关更多信息，请参阅 MQI 通道。

消息通道的每个端的定义可以是下列其中一种类型:

- 发件人 (SDR)
- 接收器 (RCVR)

- 服务器 (SVR)
- 请求者 (RQSTR)
- 集群发送方 (CLUSSDR)
- 集群接收方 (CLUSRCVR)

使用在一端定义的其中一种类型和在另一端定义的兼容类型来定义消息通道。 可能的组合包括:

- 发送方-接收方
- 请求方-服务器
- 请求方-发送方（回调）
- 服务器-接收方
- 集群发送方-集群接收方

有关创建发送方/接收方通道的详细指示信息包含在 定义通道中。 有关设置发送方/接收方通道所需的参数的示例，请参阅适用于您的平台的 示例配置信息 。 有关定义任何类型的通道所需的参数，请参阅 DEFINE CHANNEL。

发送方-接收方通道

一个系统中的发送方启动通道，以便可以将消息发送到另一个系统。 发送方请求通道另一端的接收方启动。 发送方将消息从其传输队列发送到接收方。 接收方将消息放在目标队列上。 第 156 页的图 8 对此进行了说明。

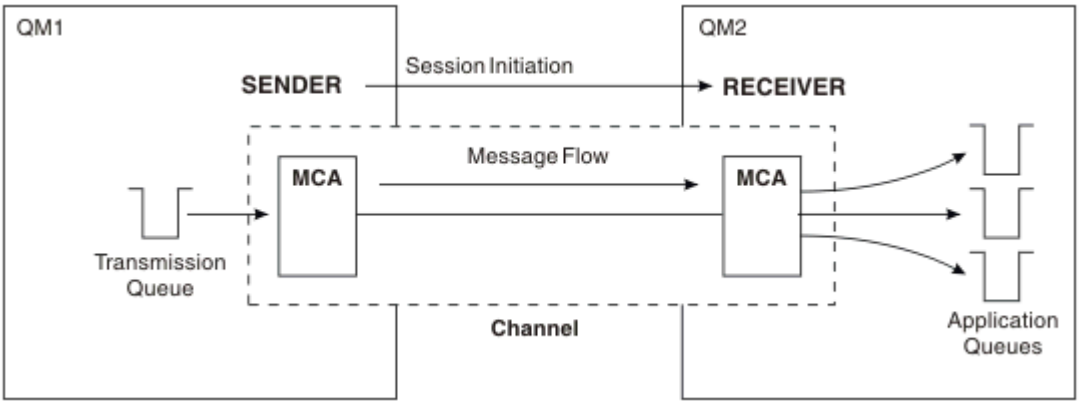


图 *8:* 一种发送方-接收方通道

请求方-服务器通道

一个系统中的请求者启动通道，以便它可以从另一个系统接收消息。 请求者请求通道另一端的服务器启动。 服务器从其通道定义中定义的传输队列向请求者发送消息。

服务器通道还可以启动通信并向请求者发送消息。 这仅适用于 标准 服务器，即在通道定义中指定了伙伴的连接名称的服务器通道。 标准服务器可以由请求者启动，也可以启动与请求者的通信。

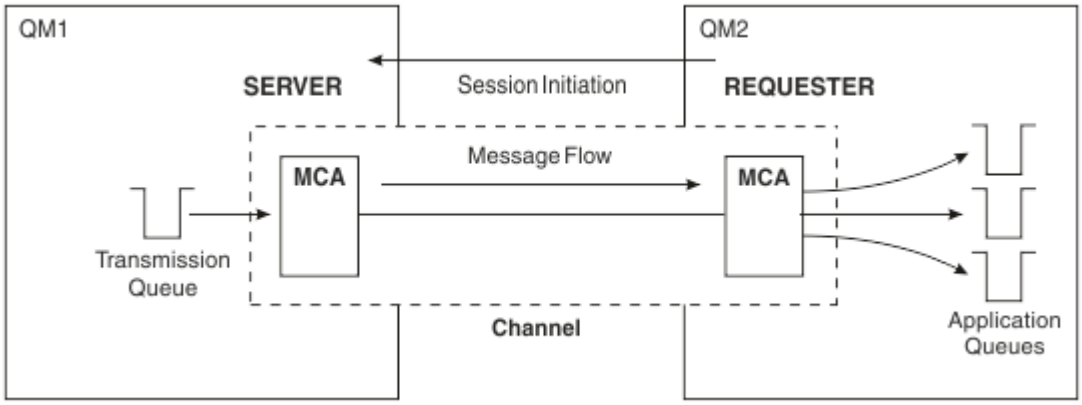


图 *9:* 请求者-服务器通道

请求方-发送方通道

请求者启动通道，发送方终止呼叫。 然后，发送方根据其通道定义 (称为 回调) 中的信息重新启动通信。 它将消息从传输队列发送到请求者。

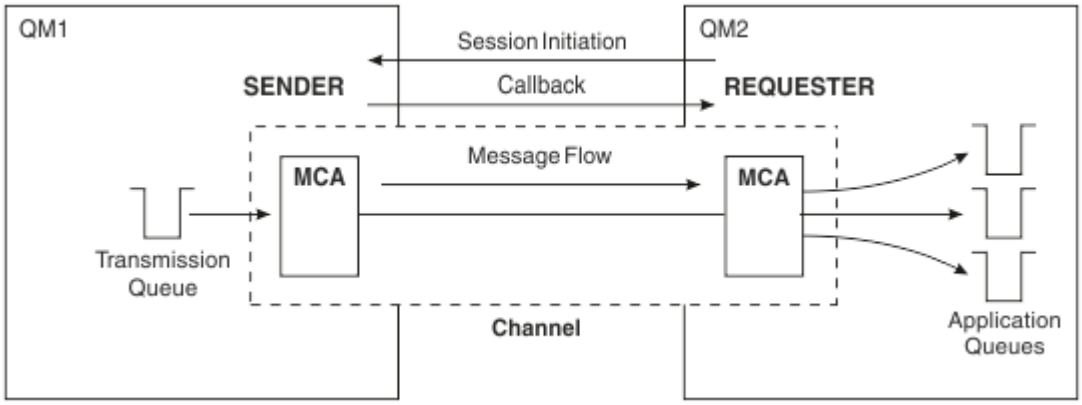


图 *10:* 请求者-发送方通道

服务器-接收方通道

这类似于发送方/接收方，但仅适用于 标准 服务器，即具有在通道定义中指定的伙伴的连接名称的服务器通道。 必须在链路的服务器端启动通道启动。 此图的插图与 第 156 页的图 8 中的插图类似。

集群发送方通道

在集群中，每个队列管理器都有一个集群发送方通道，在该通道上可以将集群信息发送到其中一个完整的存储库队列管理器。 队列管理器还可以将消息发送到集群发送方通道上的其他队列管理器。

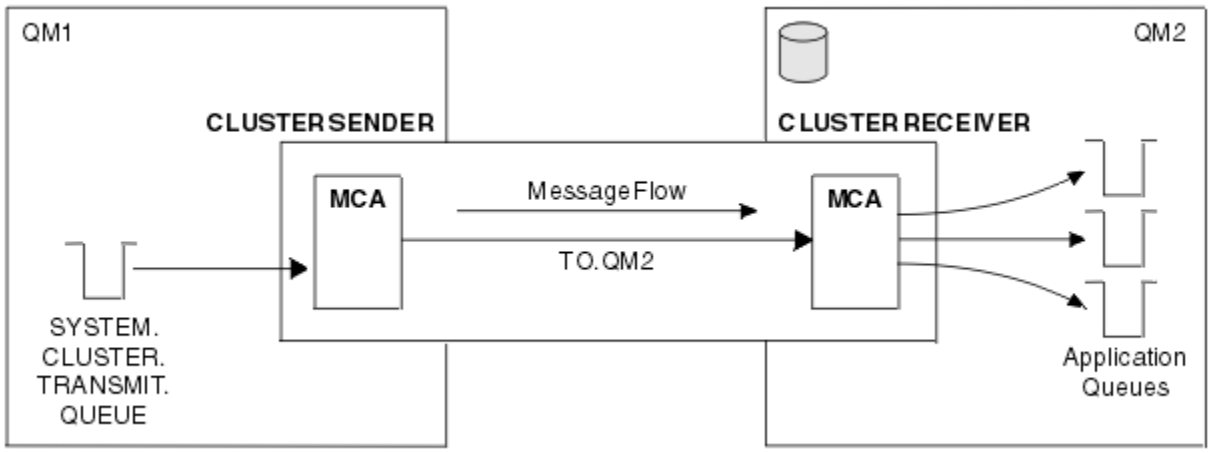


图 *11:* 集群发送方通道

集群接收方通道

在集群中，每个队列管理器都有一个集群接收方通道，在该通道上可以接收有关集群的消息和信息。 此图的插图与 第 158 页的图 11 中的插图类似。

死信队列

死信队列 (或未传递的消息队列) 是无法将消息路由到其正确目标时将消息发送到的队列。 每个队列管理器通常都有一个死信队列。

死信队列 (DLQ) (有时称为 未传递的消息队列) 是无法传递到其目标队列的消息的保留队列，例如，由于该队列不存在或已满。 对于数据转换错误，在通道的发送端也会使用死信队列。 网络中的每个队列管理器通常都有一个要用作死信队列的本地队列，以便无法传递到其正确目标的消息可以存储以供以后检索。

消息可由队列管理器，消息通道代理程序 (MCA) 和应用程序放在 DLQ 上。 死信队列上的所有消息都必须以 死信头 结构 MQDLH 作为前缀。 MQDLH 结构的 原因 字段包含标识消息在 DLQ 上的原因的原因码。

通常应该为每个队列管理器定义死信队列。 如果没有，并且 MCA 无法放入消息，那么会将其保留在传输队列上并停止通道。 此外，如果是快速的非持久消息 (请参阅 快速的非持久消息) 无法传递，并且目标系统上不存在死信队列，将废弃这些消息。

但是，使用死信队列可能会影响传递消息的顺序，因此您可能选择不使用这些消息。

远程队列定义

远程队列定义是另一个队列管理器所拥有的队列的定义。

而应用程序只能从本地队列检索消息，它们可以将消息放在本地队列或远程队列上。 因此，除了其每个本地队列的定义外，队列管理器还可以具有 远程队列定义。 远程队列定义的优点是使应用程序能够将消息放入远程队列，而不必指定远程队列或远程队列管理器的名称或传输队列的名称。 远程队列定义使您具有位置独立性。

远程队列定义还有其他用途，稍后将进行描述。

如何访问远程队列管理器

您可能并非总是在每个源队列管理器与目标队列管理器之间具有一个通道。 两者之间还有很多其他的链接方式，包括多跳，共享通道，使用不同的通道和集群。

多跳跃

如果源队列管理器与目标队列管理器之间没有直接通信链路，那么可以在到达目标队列管理器的路上通过一个或多个 中间队列管理器 。 这称为 多跳。

您需要定义所有队列管理器之间的通道，以及中间队列管理器上的传输队列。 第 159 页的图 12 对此作了说明。

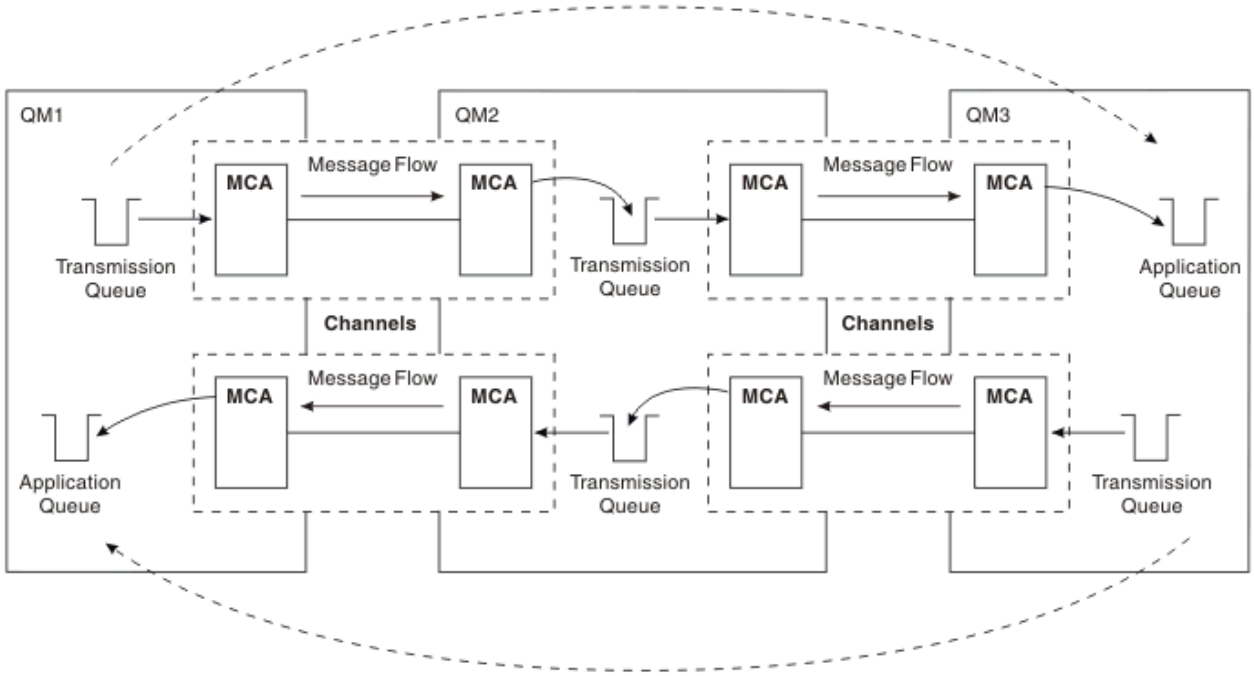


图 *12:* 通过中间队列管理器

共享通道

作为应用程序设计者，您可以选择强制应用程序指定远程队列管理器名称以及队列名称，或者为每个远程队列创建 远程队列定义 。 此定义保存远程队列管理器名称，队列名称和传输队列的名称。 无论哪种方式，来自同一远程位置的所有寻址队列的应用程序的所有消息都通过同一传输队列发送其消息。 第 159 页的图 13 对此作了说明。

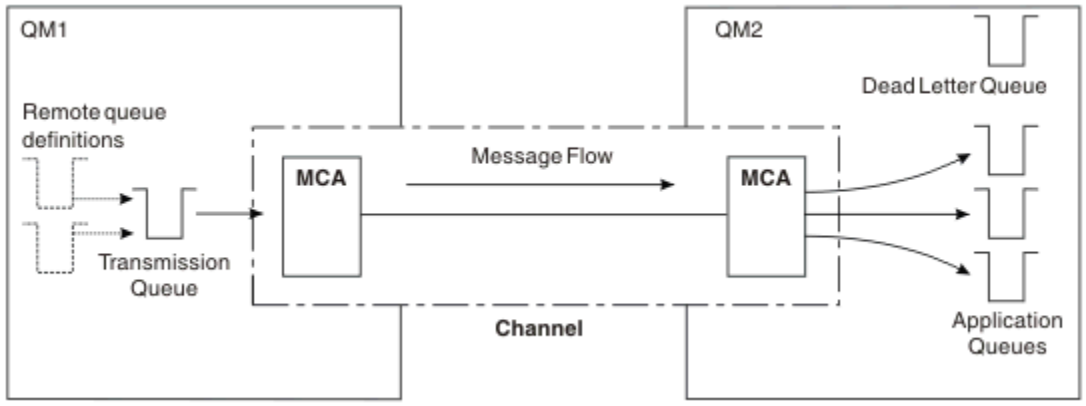


图 *13:* 共享传输队列

第 159 页的图 13 说明从多个应用程序到多个远程队列的消息可以使用同一通道。

使用不同的通道

如果要在两个队列管理器之间发送不同类型的消息，那么可以在这两个队列管理器之间定义多个通道。 有时，您需要备用通道 (可能出于安全目的) ，或者需要将交付速度与大量消息流量进行比较。

要设置第二个通道，您需要定义另一个通道和另一个传输队列，并创建指定位置和传输队列名称的远程队列定义。 然后，应用程序可以使用任一通道，但消息仍会传递到相同的目标队列。 第 160 页的图 14 对此作了说明。

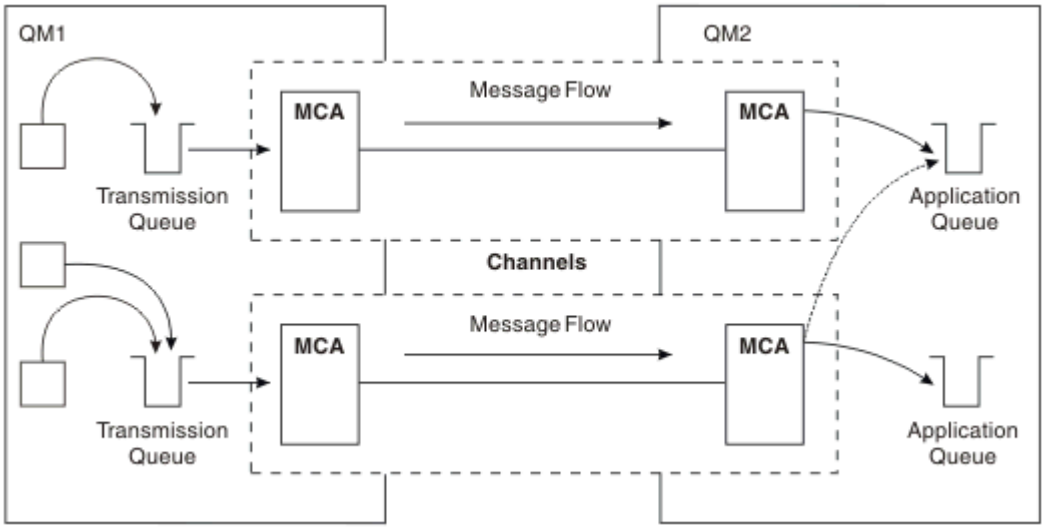


图 *14:* 使用多个通道

当您使用远程队列定义来指定传输队列时，应用程序必须 **不** 指定位置 (即目标队列管理器) 本身。 如果这样做，那么队列管理器不会使用远程队列定义。 远程队列定义使您具有位置独立性。 应用程序可以在不知道队列所在位置的情况下将消息放入 逻辑 队列，并且您可以更改 物理 队列，而不必更改应用程序。

使用集群

集群中的每个队列管理器都定义一个集群接收方通道。 当另一个队列管理器想要向该队列管理器发送消息时，它会自动定义相应的集群发送方通道。 例如，如果集群中有多个队列实例，那么可以向托管该队列的任何队列管理器定义集群发送方通道。 IBM MQ 使用工作负载管理算法，该算法使用循环法例程来选择要将消息路由到的可用队列管理器。 有关更多信息，请参阅 集群。

寻址信息

当应用程序将发往远程队列管理器的消息放入时，本地队列管理器会在将这些消息放入传输队列之前向它们添加传输头。 此头包含目标队列和队列管理器的名称，即 寻址信息。

在单队列管理器环境中，当应用程序打开要将消息放入的队列时，将建立目标队列的地址。 由于目标队列位于同一队列管理器上，因此不需要任何寻址信息。

在分布式排队环境中，队列管理器不仅需要知道目标队列名称，还需要知道该队列的位置 (即，队列管理器名称) 以及到该远程位置的路径 (即，传输队列)。 此寻址信息包含在传输头中。 接收通道除去传输头并使用其中的信息来查找目标队列。

如果使用远程队列定义，那么可以避免应用程序需要指定目标队列管理器的名称。 此定义指定远程队列的名称，将消息发送至的远程队列管理器的名称以及用于传输消息的传输队列的名称。

什么是别名?

别名用于为消息提供服务质量。 队列管理器别名使系统管理员能够更改目标队列管理器的名称，而不必更改应用程序。 它还使系统管理员能够更改到目标队列管理器的路由，或者设置涉及通过多个其他队列管理器 (多跳跃) 的路由。 应答队列别名为应答提供服务质量。

队列管理器别名和应答队列别名是使用具有空白 RNAME 的远程队列定义创建的。 这些定义不定义实际队列; 队列管理器使用这些定义来解析物理队列名称，队列管理器名称和传输队列。

别名定义的特征是具有空白 RNAME。

队列名称解析

每次打开队列时，都会在每个队列管理器上解析队列名称。 其目的是标识目标队列，目标队列管理器 (可能是本地队列) 以及到该队列管理器的路径 (可能为空)。 解析的名称有三个部分: 队列管理器名称，队列名称以及传输队列 (如果队列管理器是远程的)。

当存在远程队列定义时，不会引用任何别名定义。 应用程序提供的队列名称解析为目标队列，远程队列管理器和远程队列定义中指定的传输队列的名称。 有关队列名称解析的更多详细信息，请参阅 队列名称解析。

如果没有远程队列定义，并且指定了队列管理器名称或由名称服务解析，那么队列管理器将查看是否存在与提供的队列管理器名称匹配的队列管理器别名定义。 如果存在，那么其中的信息将用于将队列管理器名称解析为目标队列管理器的名称。 队列管理器别名定义还可用于确定到目标队列管理器的传输队列。

如果解析的队列名称不是本地队列，那么队列管理器名称和队列名称都包含在应用程序放入传输队列的每条消息的传输头中。

除非由远程队列定义或队列管理器别名定义更改，否则所使用的传输队列通常具有与已解析的队列管理器相同的名称。 如果您尚未定义此类传输队列，但已定义缺省传输队列，那么将使用此传输队列。

z/OS 在 z/OS 上运行的队列管理器的名称限制为四个字符。

队列管理器别名定义

当打开队列以放入消息的应用程序指定队列名称 **和** 队列管理器名称时，队列管理器别名定义适用。

队列管理器别名定义有三种用途:

- 发送消息时，重新映射队列管理器名称
- 发送消息时，更改或指定传输队列
- 接收消息时，确定本地队列管理器是否是这些消息的预期目标

出站消息-重新映射队列管理器名称

队列管理器别名定义可用于重新映射 MQOPEN 调用中指定的队列管理器名称。 例如， MQOPEN 调用指定队列名称 THISQ 和队列管理器名称 YOURQM。 在本地队列管理器上，存在类似于以下示例的队列管理器别名定义:

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

这表明当应用程序将消息放入队列管理器 YOURQM 时要使用的实际队列管理器是 REALQM。 如果本地队列管理器为 REALQM，那么它会将消息放入队列 THISQ(这是本地队列)。 如果本地队列管理器未被称为 REALQM，那么它会将消息路由到名为 REALQM 的传输队列。 队列管理器将传输头更改为 REALQM 而不是 YOURQM。

出站消息-更改或指定传输队列

第 162 页的图 15 显示了消息到达队列管理器 QM1 的场景，其中传输头显示队列管理器 QM3 上的队列名称。 在此场景中，可通过 QM2 进行多跳操作来访问 QM3 。

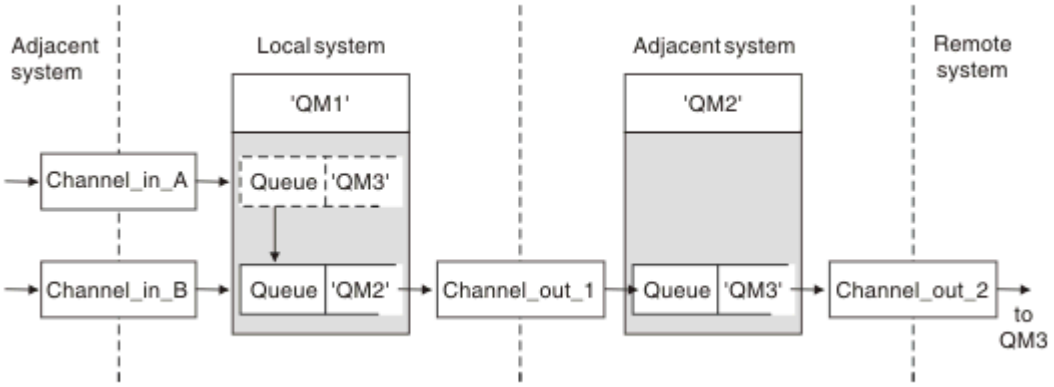


图 *15:* 队列管理器别名

QM3 的所有消息都是在 QM1 上使用队列管理器别名捕获的。 队列管理器别名名为 QM3 ， 包含通过传输队列 QM2 的定义 QM3 。 该定义类似于以下示例:

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

队列管理器将消息放在传输队列 QM2 上，但不会改变传输队列头，因为目标队列管理器 QM3 的名称不会改变。

所有到达 QM1 并在 QM2 上显示包含队列名称的传输头的消息也将放在 QM2 传输队列上。 通过这种方式，将具有不同目标的消息收集到公共传输队列中到相应的相邻系统，以便继续传输到它们的目标。

入站消息-确定目标

接收 MCA 打开传输头中引用的队列。 如果存在与所引用的队列管理器同名的队列管理器别名定义，那么在传输头中接收到的队列管理器名称将替换为该定义中的 RQMNAME。

此过程有两种用途:

- 将消息定向到另一个队列管理器
- 将队列管理器名称更改为与本地队列管理器相同

应答队列别名定义

应答队列别名定义指定消息描述符中应答信息的备用名称。 这样做的优点是可以更改队列或队列管理器的名称，而不必更改应用程序。

队列名称解析

当应用程序应答消息时，它使用它接收到的消息的 消息描述符 中的数据来找出要应答的队列的名称。 发送应用程序指示将应答发送到的位置，并将此信息附加到其消息。 此概念必须作为应用程序设计的一部分进行协调。

在将消息放入队列之前，将在应用程序的发送端进行队列名称解析。 因此，在与要向其发送消息的远程应用程序进行交互之前，将进行队列名称解析。 这是在未打开队列的情况下进行名称解析的唯一情况。

使用队列管理器别名解析队列名称

通常，应用程序指定应答队列，并将应答队列管理器名称留空。 队列管理器在放入时完成其自己的名称。 除非您希望将备用通道用于应答 (例如，使用传输队列 QM1_relief 而不是使用传输队列 QM1 的缺省返回通道) ，否则此方法正常工作。 在此情况下，在传输队列头中指定的队列管理器名称与 "实际" 队列管理器名称不匹配，而是使用队列管理器别名定义重新指定。 为了在备用路由上返回应答，还需要使用应答队列别名定义来映射应答队列数据。

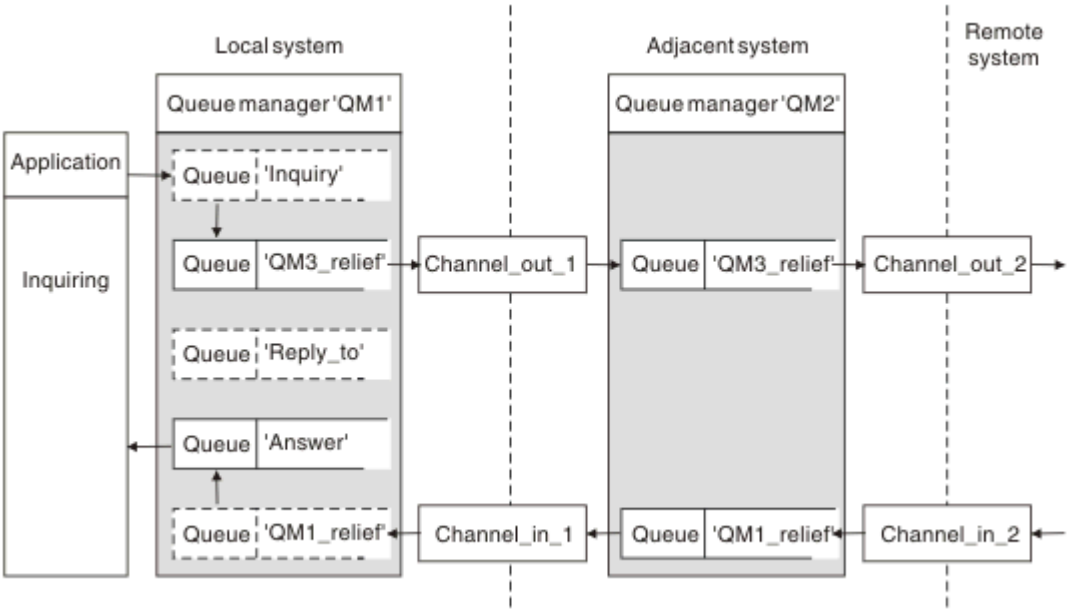


图 *16:* 用于更改应答位置的应答队列别名

在 第 163 页的图 16 中的示例中:

1. 应用程序使用 MQPUT 调用并在消息描述符中指定以下信息来放置消息:

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

   ReplyTo 队列管理器必须为空才能使用应答队列别名。
2. 创建名为 `Reply_to` 的应答队列别名定义，其中包含名称 `Answer` 和队列管理器名称 `QM1_relief`。

```
DEFINE QREMOTE ('Reply_to') RNAME ('Answer')
RQMNAME ('QM1_relief')
```

3. 将使用显示 `ReplyToQ='Answer'` 和 `ReplyToQMgr='QM1_relief'`的消息描述符来发送消息。
4. 应用程序规范必须包含将在队列 `Answer` 而不是 `Reply_to` 中找到应答的信息。

要准备应答，必须创建并行返回通道，请定义:

- 在 QM2 上，名为 `QM1_relief` 的传输队列

```
DEFINE QLOCAL ('QM1_relief') USAGE(XMITQ)
```

- 在 QM1 上，队列管理器别名为 `QM1_relief`

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

  此队列管理器别名将终止并行返回通道链，并捕获 QM1 的消息。

如果您认为您可能希望在将来某个时间执行此操作，请确保应用程序从一开始就使用别名。 现在，这是应答队列的正常队列别名，但稍后可以将其更改为队列管理器别名。

应答队列名称

命名应答队列时需要小心。 应用程序在消息中放入应答队列名称的原因是它可以指定将其应答发送到的队列。 使用此名称创建应答队列别名定义时，不能使用同名的实际应答队列 (即本地队列定义)。 因此，应答队列别名定义必须包含新的队列名称以及队列管理器名称，并且应用程序规范必须包含在此其他队列中找到其应答的信息。

应用程序现在必须从另一个队列中检索消息，而该队列在它们放置原始消息时被指定为应答队列。

集群组件

集群由队列管理器，集群存储库，集群通道和集群队列组成。

请参阅以下子主题以获取有关每个集群组件的信息:

相关信息

配置队列管理器集群
集群与分布式排队的比较
设置新集群

集群存储库

存储库是有关作为集群成员的队列管理器的信息集合。

存储库信息包括队列管理器名称，它们的位置，它们的通道，它们托管的队列以及其他信息。 该信息以消息形式存储在名为 SYSTEM.CLUSTER.REPOSITORY.QUEUE 的队列上。 队列是其中一个缺省对象。 IBM i distributed 它是在创建 IBM MQ 队列管理器时定义的。 z/OS 在 IBM MQ for z/OS 上，它定义为队列管理器定制的一部分。

通常，集群中的两个队列管理器保存完整存储库。 其余队列管理器都持有部分存储库。

完整存储库和部分存储库

托管集群中每个队列管理器的完整信息集的队列管理器具有完整的存储库。 集群中的其他队列管理器具有部分存储库，其中包含完整存储库中的一部分信息。

部分存储库仅包含有关队列管理器需要与之交换消息的那些队列管理器的信息。 队列管理器请求更新它们需要的信息，因此如果它发生更改，那么完整存储库队列管理器将向它们发送新信息。 在大部分时间内，部分存储库包含队列管理器在集群中需要执行的所有信息。 当队列管理器需要某些其他信息时，它可查询完整存储库并更新其部分存储库。 队列管理器使用名为 SYSTEM.CLUSTER.COMMAND.QUEUE 的队列来请求和接收对存储库的更新。 此队列是其中一个缺省对象 IBM i distributed 它是在创建 IBM MQ 队列管理器时定义的。 z/OS 在 IBM MQ for z/OS 上，它定义为队列管理器定制的一部分。

集群队列管理器

集群队列管理器是属于集群的队列管理器。

一个队列管理器可以隶属于多个集群。 每个集群队列管理器都必须具有在其所属的所有集群中唯一的名称。

集群队列管理器可以托管队列，它会将这些队列通告给集群中的其他队列管理器。 但是，它不必这样做。 它可以改为将消息馈送到集群中其他位置托管的队列中，并且仅接收显式定向到该队列的响应。

z/OS 在 IBM MQ for z/OS 中，集群队列管理器可以是队列共享组的成员。 在这种情况下，它与同一队列共享组中的其他队列管理器共享其队列定义。

集群队列管理器是自主的。 他们可以完全控制他们定义的队列和通道。 其他队列管理器 (同一队列共享组中的队列管理器除外) 无法修改其定义。 存储库队列管理器不控制集群中其他队列管理器中的定义。 它们包含所有定义的完整集合，以便在需要时使用。 集群是队列管理器的联合。

在集群队列管理器上创建或更改定义后，会将信息发送到完整存储库队列管理器。 稍后将更新集群中的其他存储库。

完整存储库队列管理器

完整存储库队列管理器是一个集群队列管理器，用于保存集群资源的完整表示。 要确保可用性，请在每个集群中设置两个或更多完整存储库队列管理器。 完整存储库队列管理器接收集群中其他队列管理器发送的信息并更新其存储库。 它们相互发送消息以确保它们都与有关集群的新信息保持一致。

队列管理器和存储库

每个集群都有至少一个 (最好是两个) 队列管理器，这些队列管理器保存有关集群中的队列管理器，队列和通道的完整信息存储库。 这些存储库还包含来自集群中其他队列管理器的请求，用于更新信息。

其他队列管理器各自保存部分存储库，其中包含有关需要与其通信的队列和队列管理器的子集的信息。 队列管理器通过在首次需要访问另一个队列或队列管理器时进行查询来构建其部分存储库。 他们请求将有关该队列或队列管理器的任何新信息通知他们。

每个队列管理器都将其存储库信息存储在名为 SYSTEM.CLUSTER.REPOSITORY.QUEUE 的队列上的消息中。 队列管理器在名为 SYSTEM.CLUSTER.COMMAND.QUEUE 的队列上的消息中交换存储库信息。

连接集群的每个队列管理器都定义了一个集群发送方 CLUSSDR 通道，用于连接到其中一个存储库。 它会立即了解集群中的哪些其他队列管理器保存完整存储库。 从此，队列管理器可以从任何存储库请求信息。 当队列管理器将信息发送到所选存储库时，它还会将信息发送到另一个存储库 (如果有)。

当托管完整存储库的队列管理器从与其链接的其中一个队列管理器接收新信息时，将更新完整存储库。 还会将新信息发送到另一个存储库，以降低在存储库队列管理器无法使用时将其延迟的风险。 由于将两次发送所有信息，因此存储库必须废弃重复项。 每个信息项都带有一个序号，存储库使用该序号来标识重复项。 通过交换消息使所有存储库保持一致。

集群队列

集群队列是由集群队列管理器托管并可供集群中其他队列管理器使用的队列。

集群队列定义将播发给集群中的其他队列管理器。 集群中的其他队列管理器无需相应的远程队列定义即可将消息放入集群队列。 可以使用集群名称列表在多个集群中播发集群队列。

在播发队列时，集群中的任何队列管理器都可以将消息放入该队列中。 要放入消息，队列管理器必须从完整存储库中查明托管该队列的位置。 然后，它会将一些路由信息添加到消息，并将消息放到集群传输队列上。

z/OS 集群队列可以是 IBM MQ for z/OS 中队列共享组的成员所共享的队列。

相关信息

定义集群队列

共享队列与集群队列之间的比较

此信息旨在帮助您比较共享队列和集群队列，并确定哪些可能更适合您的系统。

通道启动程序成本

在集群队列中，消息由通道发送，因此除了应用程序成本外，还允许通道启动程序成本。 由于通道获取和放置消息，因此网络中存在成本。 共享队列不存在这些成本，因此，在队列共享组中的队列管理器之间移动消息时，共享队列使用的处理能力低于集群队列。

消息的可用性

放入队列时，集群队列会将消息发送到其中一个具有连接到队列管理器的活动通道的队列管理器。 在远程队列管理器上，如果用于处理消息的应用程序不工作，那么不会处理消息并等待应用程序启动。 同样，如果队列管理器已关闭，那么在队列管理器重新启动之前，队列管理器上的任何消息都不可用。 这些实例显示的消息可用性低于使用共享队列时的可用性。

使用共享队列时，队列共享组中的任何应用程序都可以获取发送的消息。 如果关闭队列共享组中的一个队列管理器，那么消息可供其他队列管理器使用，从而提供比使用集群队列时更高的消息可用性。

容量

耦合设施比磁盘更昂贵; 因此，在本地队列中存储 1,000,000 条消息的成本低于具有足够容量来存储相同数量的消息的耦合设施。

发送到其他队列管理器

共享队列消息仅在队列共享组中可用。 如果要在队列共享组外部使用队列管理器，那么必须使用通道。 您可以使用集群在多个远程分布式队列管理器之间实现工作负载均衡。

工作负载均衡

您可以使用集群来赋予哪些通道和队列管理器获取所发送消息的比例的权重。 例如，可以将 60% 的消息发送到一个队列管理器，将 40% 的消息发送到另一个队列管理器。 此实例不依赖于远程队列管理器处理工作的能力。 具有第一个队列管理器的系统可能超负荷，具有第二个队列管理器的系统可能处于空闲状态，但大多数消息仍转至第一个队列管理器。

通过共享队列，两个 CICS 系统可以获取消息。 如果一个系统超负荷，那么另一个系统将接管大部分工作负载。

集群通道

在每个完整存储库上，手动定义一个集群接收方通道和一组集群发送方通道，以连接到集群中的每个其他完整存储库。 添加部分存储库时，请手动定义集群接收方通道以及连接到其中一个完整存储库的单个集群发送方通道。 需要时，集群会自动定义其他集群发送方通道。 自动定义的集群发送方通道从接收队列管理器上的相应集群接收方通道定义中获取其属性。

集群接收方通道: CLUSRCVR

CLUSRCVR 通道定义定义了一个通道的末尾，集群队列管理器可以在该通道上接收来自集群中其他队列管理器的消息。

必须为每个集群队列管理器至少定义一个 CLUSRCVR 通道。 通过定义 CLUSRCVR 通道，队列管理器将显示它可用于接收消息的其他集群队列管理器。

CLUSRCVR 通道定义还使其他队列管理器能够自动定义相应的集群发送方通道定义。 请参阅本文的 第 166 页的『自动定义的集群发送方通道』 部分。

集群发送方通道: CLUSSDR

您可以手动定义从每个完整存储库队列管理器到集群中每个其他完整存储库队列管理器的 CLUSSDR 通道。 由完整存储库交换的所有更新仅在这些通道上流动。 通过手动定义这些通道，可以显式地控制完整存储库的网络。

将部分存储库队列管理器添加到集群时，请手动定义单个 CLUSSDR 通道以连接到其中一个完整存储库。 这与您选择的完整存储库差别不大，因为在进行初始联系之后，将根据需要自动定义队列管理器的更多集群队列管理器对象 (包括 CLUSSDR 通道)。 这使队列管理器能够将集群信息发送到任何完整存储库，并将消息发送到集群中的任何队列管理器。

如本文部分所述，自动定义的发送方通道基于集群接收方通道的配置。 因此，您在集群通道上设置的任何通道属性都应该在匹配的 CLUSSDR 和集群接收方通道上进行相同的设置，或者仅在集群接收方通道上设置。

由于先前描述的原因，您应该仅手动定义 CLUSSDR 通道。 即，最初将部分存储库连接到完整存储库，或者将两个完整存储库连接到一起。 手动配置连接到部分存储库或不在集群中的队列管理器的 CLUSSDR 通道会导致发出错误消息，例如 AMQ9427 和 AMQ9428 。 虽然这有时可能作为临时情况不可避免，例如在修改完整存储库的位置时，应该尽快删除手动定义。

自动定义的集群发送方通道

通常，将部分存储库队列管理器添加到集群时，仅手动定义队列管理器上的两个集群通道:

- 集群发送方 (CLUSSDR) 到集群的完整存储库队列管理器的通道。

- 集群接收方 (CLUSRCVR) 通道。

您定义的 CLUSSDR 通道允许队列管理器与集群进行初始联系。 进行初始联系后，集群会在需要时自动定义更多 CLUSSDR 通道。

自动定义的 CLUSSDR 通道从接收队列管理器上相应的 CLUSRCVR 通道定义中获取其属性。 即使存在手动定义的 CLUSSDR 通道，也会使用自动定义的 CLUSSDR 通道中的属性。 例如，假设您定义 CLUSRCVR 通道而不在 CONNAME 参数中指定端口号，并手动定义指定端口号的 CLUSSDR 通道。 当自动定义的 CLUSSDR 通道替换手动定义的通道时，端口号 (取自 CLUSRCVR 通道) 将变为空白。 将使用缺省端口号，并且通道将发生故障。

如果手动定义的 CLUSSDR 通道与相应的 CLUSRCVR 通道定义之间存在配置差异，那么某些差异将立即生效 (例如，工作负载均衡参数) ，而某些差异仅在通道重新启动 (例如， SSL 配置) 时生效。

为避免混淆，请尽可能遵守以下准则:

- 仅手动定义 CLUSSDR 通道以指向完整存储库。
- 如果您已手动定义 CLUSSDR 通道，请将其配置为与接收队列管理器上相应的 CLUSRCVR 通道定义完全匹配。

另请参阅 使用自动定义的通道。

相关信息

设置新集群
将队列管理器添加至集群
使用自动定义的通道
使用集群传输队列和集群发送方通道

集群主题

集群主题是定义了 **cluster** 属性的管理主题。 有关集群主题的信息会推送给集群的所有成员，并与本地主题相结合以创建横跨多个队列管理器的主题空间的一些部分。 这支持将一个队列管理器中的某个主题上发布的消息传递给集群中其他队列管理器的预订。

在队列管理器上定义集群主题时，会将集群主题定义发送到完整存储库队列管理器。 然后，完整存储库会将集群主题定义传播到集群中的所有队列管理器，从而使相同的集群主题可用于集群中任何队列管理器处的发布者和订户。 您在其中创建集群主题的队列管理器称为集群主题主机。 集群主题可供集群中的任何队列管理器使用，但是对集群主题的任何修改必须在定义该主题的队列管理器（主机）上执行，此时会通过完整存储库将修改传播到集群的所有成员。

有关配置集群主题以使用 直接路由 或 主题主机路由的信息，以及有关集群主题继承和通配符预订的信息，请参阅 定义集群主题。

有关用于显示集群主题的命令的信息，请参阅相关信息。

相关信息

处理管理主题
使用预订
DISPLAY TOPIC
DISPLAY TPSTATUS
显示子项

缺省集群对象

在分布式平台和 IBM i 平台上，缺省集群对象包含在定义队列管理器时自动创建的缺省对象集中。

z/OS 在 z/OS 上，可以在定制样本中找到缺省集群对象定义。

注: 您可以通过运行 MQSC 或 PCF 命令，以与任何其他通道定义相同的方式更改缺省通道定义。 请勿更改缺省队列定义， SYSTEM.CLUSTER.HISTORY.QUEUE 除外。

SYSTEM.CLUSTER.COMMAND.QUEUE
集群中的每个队列管理器都有一个名为 SYSTEM.CLUSTER.COMMAND.QUEUE 的本地队列，用于将消息传输到完整存储库。 此消息包含有关队列管理器的任何新信息或已更改的信息，或者有关其他队列管理器的任何信息请求。 SYSTEM.CLUSTER.COMMAND.QUEUE 通常为空。

SYSTEM.CLUSTER.HISTORY.QUEUE
集群中的每个队列管理器都有一个名为 SYSTEM.CLUSTER.HISTORY.QUEUE 的本地队列。 SYSTEM.CLUSTER.HISTORY.QUEUE 用于存储集群状态信息的历史记录以用于服务目的。

在缺省对象设置中， SYSTEM.CLUSTER.HISTORY.QUEUE 设置为 PUT (ENABLED)。 要禁止历史记录收集，请将设置更改为 PUT (DISABLED)。

SYSTEM.CLUSTER.REPOSITORY.QUEUE
集群中的每个队列管理器都有一个名为 SYSTEM.CLUSTER.REPOSITORY.QUEUE 的本地队列。 此队列用于存储所有完整存储库信息。 此队列通常不为空。

SYSTEM.CLUSTER.TRANSMIT.QUEUE
每个队列管理器都有一个名为 SYSTEM.CLUSTER.TRANSMIT.QUEUE 的本地队列的定义。 SYSTEM.CLUSTER.TRANSMIT.QUEUE 是所有消息到集群中所有队列和队列管理器的缺省传输队列。 您可以通过更改队列管理器属性 DEFCLXQ 将每个集群发送方通道的缺省传输队列更改为 SYSTEM.CLUSTER.TRANSMIT. *ChannelName*。 无法删除 SYSTEM.CLUSTER.TRANSMIT.QUEUE。 它还用于定义授权检查所使用的缺省传输队列是 SYSTEM.CLUSTER.TRANSMIT.QUEUE 还是 SYSTEM.CLUSTER.TRANSMIT. *ChannelName*。

SYSTEM.DEF.CLUSRCVR
每个集群都有一个名为 SYSTEM.DEF.CLUSRCVR 的缺省 CLUSRCVR 通道定义。 SYSTEM.DEF.CLUSRCVR 用于为您在集群中的队列管理器上创建集群接收方通道时未指定的任何属性提供缺省值。

SYSTEM.DEF.CLUSSDR
每个集群都有一个名为 SYSTEM.DEF.CLUSSDR 的缺省 CLUSSDR 通道定义。 SYSTEM.DEF.CLUSSDR 用于为您在集群中的队列管理器上创建集群发送方通道时未指定的任何属性提供缺省值。

相关信息
使用缺省集群对象

发布/预订消息传递

发布/预订消息传递允许您使信息的提供者与该信息的使用者分离开来。 发送应用程序和接收应用程序不需要相互了解即可发送和接收信息。

在点到点 IBM MQ 应用程序可以向另一个应用程序发送消息之前，它需要了解有关该应用程序的一些信息。 例如，它需要知道要向其发送信息的队列的名称，并且还可以指定队列管理器名称。

IBM MQ 发布/预订使您的应用程序无需了解有关目标应用程序的任何信息。 发送应用程序必须执行以下操作:

- 放置 包含应用程序需要的信息的 IBM MQ 消息。
- 将消息分配给表示信息主题的主题。
- 让 IBM MQ 处理该信息的分发。

类似地，目标应用程序不必知道有关其接收的信息源的任何信息。

下图显示了最简单的发布/预订系统。 有一个发布程序，一个队列管理器和一个订户。 预订由队列管理器上的订户进行，发布从发布者发送到队列管理器，然后由队列管理器将发布转发给订户。

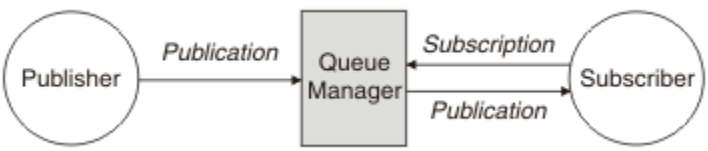


图 *17:* 简单发布/预订配置

典型的发布/预订系统在许多不同的主题上具有多个发布者和多个订户，并且通常具有多个队列管理器。 应用程序可以是发布者和订户。

发布/预订消息传递与点到点之间的另一个显着差异是，发送到点到点队列的消息仅由单个使用应用程序处理。 发布到发布/预订主题的消息 (其中多个订户已注册兴趣) 由每个感兴趣的订户处理。

发布/预订组件

发布/订阅是订阅者以消息形式从发布者接收信息的机制。 发布者与订户之间的交互由队列管理器使用标准 IBM MQ 工具进行控制。

典型的发布/预订系统在许多不同的主题上具有多个发布者和多个订户，并且通常具有多个队列管理器。 应用程序可以是发布者和订户。

信息的提供者称为发布者。 发布者提供关于主题的信息，而不必了解任何关于对该信息有兴趣的应用程序的情况。 发布者以消息形式生成此信息，称为 发布 ，他们要发布这些消息并定义这些消息的主题。

信息的使用者称为订户。 订户创建描述订户感兴趣的主题的 预订 。 因此，订阅确定了转发给订阅者的发布内容。 订户可以创建多个预订，并可以接收来自许多不同发布者的信息。

已发布的信息将在 IBM MQ 消息中发送，并且该信息的主题由其 主题标识。 发布者在发布信息时指定主题，订户指定要接收发布的主题。 仅向订户发送有关其预订的主题的信息。

它是存在的主题，允许信息的提供者和使用者在发布/预订消息传递中解耦，方法是根据点到点消息传递中的要求，消除在每条消息中包含特定目标的需求。

发布者和订户之间的交互都由队列管理器控制。 队列管理器从发布者接收消息，并从订户接收预订 (针对一系列主题)。 队列管理器的作业是将已发布的消息路由到已在消息主题中注册兴趣的订户。

标准 IBM MQ 工具用于分发消息，因此应用程序可以使用可供现有 IBM MQ 应用程序使用的所有功能。 这意味着您可以使用持久消息来获取仅一次有保证的传递，并且您的消息可以是事务性工作单元的一部分，以确保仅当消息由发布者落实时才将其传递给订户。

出版商和出版物

在 IBM MQ 发布/预订中，发布者是一个应用程序，它以称为发布的标准 IBM MQ 消息的形式向队列管理器提供有关指定主题的信息。 发布者可以发布有关多个主题的信息。

发布者使用 MQPUT 动词将消息放入先前打开的主题，此消息是发布。 然后，本地队列管理器会将发布路由到预订该发布主题的任何订户。 已发布的消息可以由多个订户使用。

除了将发布分发给具有相应预订的所有本地订户外，队列管理器还可以直接或通过具有主题订户的队列管理器网络将发布分发给与其连接的任何其他队列管理器。

在 IBM MQ 发布/预订网络中，发布应用程序也可以是订户。

同步点下的出版物

发布者可以在同步点中发出 MQPUT 或 MQPUT1 调用，以在工作单元中包含传递给订户的所有消息。 如果指定了值为 ALL 或 ALLDUR 的 MQPMO_RETAIN 选项或主题交付选项 NPMSGDLV 或 PMSGDLV ，那么队列管理器将在发布程序 MQPUT 或 MQPUT1 调用的作用域内同步点中使用内部 MQPUT 或 MQPUT1 调用。

状态和事件信息

可以将出版物分类为状态出版物 (例如，库存的当前价格) 或事件出版物 (例如，该库存的贸易)。

状态发布

状态发布 包含有关某事物的当前状态的信息，例如，库存价格或足球比赛中的当前分数。 当发生某件事（如股票价格更改或足球比分更改）时，将不再需要先前的状态信息，因为它已被新的信息取代。

订户将希望在启动时接收当前版本的状态信息，并在状态更改时被发送新信息。

如果发布内容包含了状态信息，其通常作为保留发布内容被发布。 新订户通常希望立即获取当前状态信息; 订户不希望等待导致重新发布信息的事件。 除非预订程序使用 MQSO_PUBLIC ICATIONS_ON_REQUEST 或 MQSO_NEW_PUBLIC ICATIONS_ONLY 选项，否则预订程序在预订时将自动接收主题的保留发布。

事件发布

事件发布 包含有关发生的个别事件的信息，例如某个库存的交易或特定目标的评分。 每个事件独立于其他事件。

订户将希望在事件发生时接收有关这些事件的信息。

保留的发布内容

缺省情况下，在将发布发送到所有感兴趣的订户之后，将废弃该发布。 但是，发布者可以指定保留发布的副本，以便可以将其发送给在主题中注册兴趣的未来订户。

在将发布发送给所有感兴趣的订户后删除这些发布适用于事件信息，但并不总是适用于状态信息。 通过保留消息，新订阅者不必等待再次发布信息即可接收到初始状态信息。 例如，预订股票价格的订户将直接收到当前价格，而无需等待股票价格更改 (并因此重新发布)。

队列管理器只能为每个主题保留一个发布，因此当新的保留发布到达队列管理器时，将删除主题的现有保留发布。 但是，删除现有发布可能不会在新保留发布到达时同步发生。 因此，只要有可能，就有不超过一个发布者发送有关任何主题的保留发布。

订户可以使用 MQSO_NEW_publicATIONS_ONLY 预订选项指定他们不希望接收保留发布。 现有订户可以要求向其发送保留发布的重复副本。

有时，您可能不希望保留发布，即使是针对状态信息:

- 如果在对某个主题进行任何发布之前对该主题进行了所有预订，并且您不期望或不允许新预订，那么不需要保留发布，因为这些发布在第一次发布时交付到完整的订户集。
- 如果发布频繁发生 (例如每秒) ，那么新订户 (或从故障中恢复的订户) 几乎在其初始预订之后立即接收到当前状态，因此不需要保留这些发布。
- 如果发布内容较大，那么最终可能需要大量存储空间来存储每个主题的保留发布内容。 在多队列管理器环境中，保留发布由具有匹配预订的网络中的所有队列管理器存储。

在决定是否使用保留发布时，请考虑预订应用程序如何从故障中恢复。 如果发布者不使用保留发布，那么订户应用程序可能需要在本地存储其当前状态。

要确保保留发布，请使用 MQPMO_RETAIN put-message 选项。 如果使用此选项并且无法保留发布，那么将不会发布消息，并且调用将失败并返回 MQRC_PUT_NOT_保留。

如果消息是保留发布，那么此消息由 MQIsRetained 消息属性指示。 消息的持久性与最初发布消息时一样。

相关信息

发布/预订集群中保留的发布的设计注意事项

同步点下的出版物

在 IBM MQ 发布/预订中，同步点可以由发布者使用，也可以由队列管理器在内部使用。

发布程序在使用 MQPMO_SYNCPOINT 选项发出 MQPUT/MQPUT1 调用时使用同步点。 传递到订户的所有消息将计入 work.The MAXUMSGS 队列管理器属性指定此限制。 如果达到限制，那么发布程序将接收到 2024 (07E8) (RC2024) :MQRC_SYNCPOINT_LIMIT_已达到 原因码。

当发布者使用带有 MQPMO_RETAIN 选项的 MQPMO_NO_SYNCPOINT 或带有值 ALL 或 ALLDUR 的主题传递选项 NPMSGDV/PMSGDV 发出 MQPUT/MQPUT1 调用时，队列管理器将使用内部同步点来保证按请求传递消息。 如果在发布程序 MQPUT/MQPUT1 调用范围内达到限制，那么发布程序可以接收 2024 (07E8) (RC2024): MQRC_SYNCPOINT_LIMIT_已达到 原因码。

订户和预订

在 IBM MQ 发布/预订中，订户是从发布/预订网络中的队列管理器请求有关特定主题的信息的应用程序。 订户可以从多个发布程序接收关于相同或不同主题的消息。

可以使用 MQSC 命令手动创建预订，也可以由应用程序手动创建预订。 这些预订将向本地队列管理器发出，并包含有关订户要接收的发布的信息:

- 订户感兴趣的主题; 如果使用通配符，那么这可以解析为多个主题。
- 要应用于已发布消息的可选的选择字符串。
- 队列 (称为 订户队列) 的句柄，应该将所选发布放在该队列上，以及可选的 CorrelId。

本地队列管理器存储预订信息，当它接收到发布时，扫描该信息以确定是否存在与发布的主题和选择字符串相匹配的预订。 对于每个匹配的预订，队列管理器会将发布定向到订户的订户队列。 可以使用 DIS SUB 和 DIS SBSTATUS 命令来查看队列管理器存储的有关预订的信息。

仅当发生下列其中一个事件时，才会删除预订:

- 订户使用 MQCLOSE 调用取消预订 (如果以非持久方式进行预订)。
- 预订将到期。
- 系统管理员使用 DELETE SUB 命令删除预订。
- 订户应用程序结束 (如果预订以非持久方式进行)。
- 队列管理器将停止或重新启动 (如果预订以非持久方式进行)。

获取消息时，请在 MQGET 调用上使用相应的选项。 如果应用程序仅处理一个预订的消息，那么至少应使用 `get-by-correlid`，如 C 样本程序 `amqssbxa.c` 和 非受管 MQ 订户中所示。 要使用的 **CorrelId** 从 MQSD 中的 MQSUB 返回。 **SubCorrelId** 字段。

相关信息

克隆和共享的预订

如何定义 sharedSubscription 属性的示例

受管队列和发布/预订

创建预订时，可以选择使用受管排队。 如果使用受管排队，那么在创建预订时将自动创建预订队列。 将根据预订的耐久性自动对受管队列进行分层。 使用受管队列意味着您不必担心创建队列以接收发布，如果非持久预订连接已关闭，那么将自动从订户队列中除去任何未使用的发布。

如果应用程序不需要将特定队列用作其订户队列 (它接收的发布的目标) ，那么可以使用 MQSO_MANAGED 预订选项来使用 受管预订 。 如果您创建受管预订，那么队列管理器会将对象句柄返回给订户队列的订户，队列管理器将在该订户队列中创建将接收发布的对象句柄。 将返回队列的对象句柄，允许您浏览，获取或查询队列 (除非已显式授予您对临时动态队列的访问权，否则无法放入或设置受管队列的属性)。

预订的耐久性确定在预订应用程序与队列管理器的连接中断时受管队列是否保留。

与非持久预订配合使用时，受管预订特别有用，因为当应用程序的连接结束时，未使用的消息将无限期地保留在订户队列上占用队列管理器中的空间。 如果您正在使用受管预订，那么该受管队列将是临时动态队列，因此，当由于以下任何原因导致连接中断时，将删除该受管队列以及任何未使用的消息:

- 使用带有 MQCO_REMOVE_SUB 的 MQCLOSE ，并关闭受管 Hobj。
- 与使用非持久预订 (MQSO_NON_耐久性) 的应用程序的连接丢失。
- 由于预订已到期并且受管 Hobj 已关闭，因此将除去该预订。

受管预订也可以与持久预订配合使用，但您可能希望在订户队列中保留未使用的消息，以便在重新打开连接时可以检索这些消息。 因此，持久预订的受管队列采用永久动态队列形式，并将在预订应用程序与队列管理器的连接中断时保留。

如果要使用永久动态受管队列，那么可以在预订上设置到期时间，以便尽管该队列在连接中断后仍将存在，但它不会无限期地继续存在。

如果删除受管队列，那么将接收到错误消息。

创建的受管队列在结束时使用数字 (时间戳记) 进行命名，因此每个队列都是唯一的。

预订耐久性

可以将预订配置为持久或非持久。 预订耐久性确定在预订应用程序与队列管理器断开连接时对预订执行的操作。

持久预订

持久预订在预订应用程序与队列管理器之间的连接关闭后继续存在。 如果预订是持久的，那么当预订应用程序断开连接时，预订将保留在原位置，并且可以由预订应用程序在使用创建预订时返回的 **SubName** 重新连接请求预订时使用。

持久预订时，预订名称 (**SubName**) 是必需的。 预订名称在队列管理器中必须唯一，以便可用于标识预订。 如果您有意关闭与预订的连接 (使用 MQCO_KEEP_SUB 选项) 或者已与队列管理器断开连接，那么在指定要恢复的预订时，必须使用此标识方法。 您可以使用带有 MQSO_RESUME 选项的 MQSUB 调用来恢复现有预订。 如果将 DISPLAY SBSTATUS 命令与 SUBTYPE ALL 或 ADMIN 配合使用，那么还会显示预订名称。

当应用程序不再需要持久预订时，可以使用带有 MQCO_REMOVE_SUB 选项的 MQCLOSE 函数调用将其除去，也可以使用 MQSC 命令 DELETE SUB 手动将其删除。

您可以使用 **DURSUB** 主题属性来指定是否可以对主题进行持久预订。

使用 MQSO_RESUME 选项从 MQSUB 调用返回时，预订到期将设置为预订的原始到期时间，而不是剩余到期时间。

队列管理器继续发送发布以满足持久预订，即使该订户应用程序未连接也是如此。 这将导致在订户队列上构建消息。 避免此问题的最简单方法是在适当情况下使用非持久预订。 但是，在需要使用持久预订的情况下，如果订户使用 保留发布 选项进行预订，那么可以避免构建消息。 然后，订户可以通过使用 MQSUBRQ 调用来控制何时接收发布。

非持久预订

仅当预订应用程序与队列管理器的连接保持打开状态时，非持久预订才存在。 当预订应用程序有意地断开或由于连接中断而断开与队列管理器的连接时，将除去此预订。 关闭连接时，将从队列管理器中除去有关预订的信息，如果使用 DISPLAY SBSTATUS 命令显示预订，那么将不再显示这些信息。 不会将更多消息放入订户队列。

对于非持久预订的订户队列上的任何未使用的发布，将按如下所示进行确定。

- 如果预订应用程序正在使用 受管目标，那么将自动除去尚未使用的任何发布。
- 如果预订应用程序在预订时向其自己的订户队列提供句柄，那么不会自动除去未使用的消息。 如果需要，应用程序负责清除队列。 如果队列由多个订户或其他点到点应用程序共享，那么可能不适合完全清除该队列。

虽然非持久预订不需要预订名称，但队列管理器会使用预订名称 (如果提供)。 预订名称在队列管理器中必须唯一，以便可用于标识预订。

相关信息

克隆和共享的预订

如何定义 sharedSubscription 属性的示例

选择字符串

选择字符串 是应用于发布以确定其是否与预订匹配的表达式。 选择字符串可以包含通配符。

预订时，除了指定主题外，还可以指定选择字符串以根据其消息属性选择发布。

在修改消息以传递给每个订户之前，将根据发布程序放置的消息对选择字符串进行求值。 在选择字符串中使用可能在发布操作过程中修改的字段时，请小心。 例如， MQMD 字段 `UserIdentifier`， `MsgId` 和 `CorrelId`。

选择字符串不应引用队列管理器在发布操作中添加的任何消息属性字段 (请参阅 发布/预订消息属性)，但包含发布主题字符串的消息属性 `MQTopicString` 除外。

相关信息

选择字符串规则和限制

主题

主题是发布/订阅消息中发布的信息的论题。

将点到点系统中的消息发送到特定目标地址。 基于主题的发布/预订系统中的消息将根据描述消息内容的主题发送给订户。 在基于内容的系统中，根据消息本身的内容向订户发送消息。

IBM MQ 发布/预订系统是基于主题的发布/预订系统。 发布者创建一条消息，并使用最适合发布主题的主题字符串来发布该消息。 要接收发布内容，订户创建带有模式匹配主题字符串的预订，该字符串用于选择发布内容主题。 队列管理器将发布内容传递给其预订与发布内容主题匹配并有权接收发布内容的订户。 文章 第 173 页的『主题字符串』描述了用于标识出版物主题的主题字符串的语法。 订户还创建主题字符串以选择要接收的主题。 订户创建的主题字符串可以包含两个备用通配符方案中的任何一个，以便与发布内容中的主题字符串进行模式匹配。 第 174 页的『通配方案』中描述了模式匹配。

在基于主题的发布/预订中，发布者或管理员负责将主题分类为主题。 通常，主题以分层方式组织到主题树中，使用 '/' 字符在主题字符串中创建子主题。 请参阅 第 179 页的『主题树』 以获取主题树的示例。 主题是主题树中的节点。 主题可以是没有进一步子主题的叶节点，也可以是具有子主题的中间节点。

在将主题组织到分层主题树中的同时，可以将主题与管理主题对象相关联。 通过将主题与管理主题对象相关联，可以将属性分配给该主题，例如，该主题是否分布在集群中。 通过使用管理主题对象的 TOPICSTR 属性对主题进行命名来进行关联。 如果未显式将管理主题对象与主题关联，那么该主题将继承其在 具有 与管理主题对象关联的主题树中最接近的祖代的属性。 如果您根本未定义任何父主题，那么它将从 SYSTEM.BASE.TOPIC。 第 180 页的『管理主题对象』中描述了管理主题对象。

注: 即使您从 SYSTEM.BASE.TOPIC，为直接继承自 SYSTEM.BASE.TOPIC。 例如，在美国各州的主题空间中， USA/Alabama USA/Alaska 等， USA 是根主题。 根主题的主要用途是创建离散的非重叠主题空间，以避免发布与错误的预订匹配。 这也意味着您可以更改根主题的属性以影响整个主题空间。 例如，可以设置 **CLUSTER** 属性的名称。

以发布者或订户身份引用主题时，可以选择提供主题字符串或引用主题对象。 或者可以同时执行这两种操作，在这种情况下，您提供的主题字符串将定义主题对象的子主题。 队列管理器通过将主题字符串附加到主题对象中指定的主题字符串前缀，在两个主题字符串之间插入额外的 '/' (例如， 主题字符串/对象字符串) 来标识主题。 第 177 页的『组合主题字符串』 进一步对此进行了描述。 生成的主题字符串用于标识主题并将其与管理主题对象相关联。 管理主题对象不一定是与主主题对应的主题对象相同的主题对象。

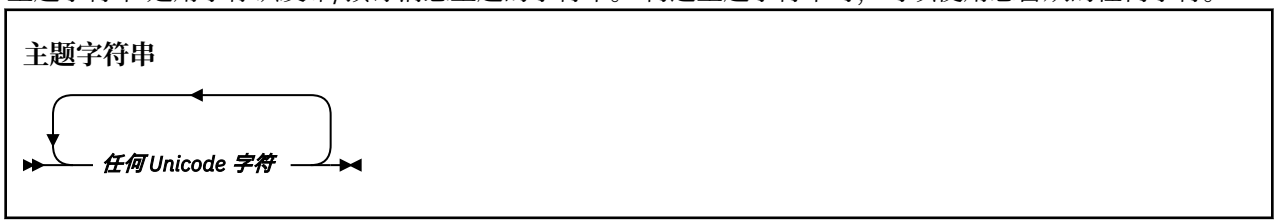
在基于内容的发布/预订中，您可以通过提供用于搜索每条消息内容的选择字符串来定义要接收的消息。 IBM MQ 提供了基于内容的发布/预订的中间形式，使用消息选择器来扫描消息属性而不是消息的完整内容，请参阅 选择器。 消息选择器的原型用途是预订主题，然后使用数字属性限定选择。 选择器使您能够指定您只对特定范围内的值感兴趣; 不能使用字符或基于主题的通配符来执行任何操作。 如果确实需要根据消息的完整内容进行过滤，那么需要使用 IBM Integration Bus。

主题字符串

您使用主题字符串将标签信息发布为主题。 然后，使用基于字符或主题的通配主题字符串来预订一组主题。

主题

主题字符串 是用于标识发布/预订消息主题的字符串。 构造主题字符串时，可以使用您喜欢的任何字符。



三个字符在 V 7 发布/预订中具有特殊含义。 在主题字符串中的任何位置都允许这些值，但请谨慎使用这些值。 在 第 174 页的『基于主题的通配方案』中说明了特殊字符的用法。

正斜杠 (/)

主题级别分隔符。 使用 '/' 字符将主题构造为主题树。

如果可以，请避免使用空的主题级别 '//'。 这些节点对应于主题层次结构中没有主题字符串的节点。 主题字符串中的前导或尾部 '/' 对应于前导或尾部空节点，也应避免。

散列符号 (#)

与 '/' 结合使用以在预订中构造多级通配符。 请注意在用于命名已发布主题的主题字符串中使用与 '/' 相邻的 '#' 。 第 174 页的『主题字符串示例』 显示了对 '#'的合理使用。

字符串 '.../#/...', '#/...' 和 '.../#' 在预订主题字符串中具有特殊含义。 这些字符串与主题层次结构中的一个或多个级别的所有主题匹配。 因此，如果您创建了具有其中一个序列的主题，那么无法预订该主题，同时也无法预订主题层次结构中多个级别的所有主题。

加号 (+)

与 '/' 结合使用以在预订中构造单层通配符。 请注意在用于命名已发布主题的主题字符串中使用与 '/' 相邻的 '+' 。

字符串 '.../+/...', '+/...' 和 '.../+' 在预订主题字符串中具有特殊含义。 这些字符串与主题层次结构中的一个级别的所有主题匹配。 因此，如果您创建了具有其中一个序列的主题，那么无法预订该主题，也无法在主题层次结构中的一个级别预订所有主题。

主题字符串示例

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

相关信息

TOPIC

通配方案

有两种用于预订多个主题的通配方案。 对方案的选择是预订选项。

MQSO_WILDCARD_TOPIC

选择要使用基于主题的通配符方案预订的主题。

如果未显式选择通配符模式，那么这是缺省值。

MQSO_WILDCARD_CHAR

选择要使用基于字符的通配符方案预订的主题。

通过在 DEFINE SUB 命令上指定 **wschema** 参数来设置任一方案。 有关更多信息，请参阅 DEFINE SUB。

注: 在 IBM WebSphere MQ 7.0 之前创建的预订始终使用基于字符的通配符方案。

示例

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

基于主题的通配方案

基于主题的通配符允许订户每次预订多个主题。

基于主题的通配符是 IBM MQ 发布/预订中主题系统的强大功能。 多点传送通配符和单一级别通配符可用于预订，但不能由消息的发布者在主题中使用。

基于主题的通配场景允许您选择按主题级别进行分组的发布内容。 对于主题层次结构中的每个级别，您可以选择该主题级别的订阅中的字符串是否必须与发布内容中的该字符串完全匹配。 例如，预订 IBM/+/Results 选择所有主题，

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

有两种类型的通配符。

多级别通配符

- 多级别通配符用于订阅中。 在发布中使用时，会将其视为文字。
- 多级通配符 '#' 用于匹配主题中的任意数量的级别。 例如，使用示例主题树，如果预订 'USA/Alaska/#'，那么将接收有关主题 'USA/Alaska' 和 'USA/Alaska/Juneau'的消息。
- 多级别通配符可以不表示任何级别，也可以表示多个级别。 因此， 'USA/#' 还可以与奇异 'USA'匹配，其中 '#' 表示零级别。 因为没有要分隔的级别，所以主题级别分隔符在此上下文中没有含义。
- 多级别通配符仅当单独指定或紧随主题级别分隔符指定时才有效。 因此， '#' 和 'USA/#' 是将 '#' 字符视为通配符的有效主题。 但是，虽然 'USA#' 也是有效的主题字符串，但 '#' 字符不会被视为通配符，并且没有任何特殊含义。 请参阅 第 176 页的『当基于主题的通配符无效时』，以了解更多信息。

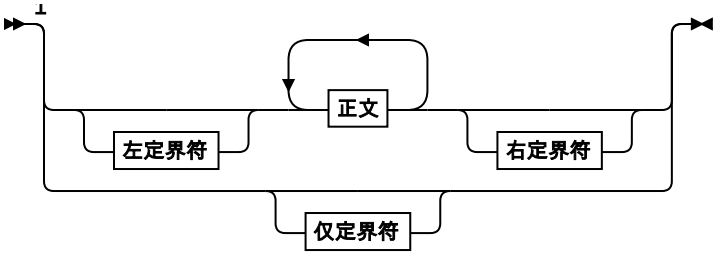
单一级别通配符

- 单一级别通配符用于订阅中。 在发布中使用时，会将其视为文字。
- 单级别通配符 '+' 与一个且仅与一个主题级别匹配。 例如， 'USA/+' 与 'USA/Alabama'匹配，但与 'USA/Alabama/Auburn'不匹配。 因为单级别通配符仅与单个级别匹配，所以 'USA/+' 与 'USA'不匹配。
- 可以在主题树中的任何级别使用单层通配符，并将其与多层通配符结合使用。 除了单独指定单一级别通配符时以外，必须紧随主题级别分隔符指定单一级别通配符。 因此， '+' 和 'USA/+' 是将 '+' 字符视为通配符的有效主题。 但是，虽然 'USA+' 也是有效的主题字符串，但 '+' 字符不会被视为通配符，并且没有任何特殊含义。 请参阅 第 176 页的『当基于主题的通配符无效时』，以了解更多信息。

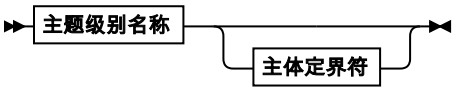
基于主题的通配符方案的语法没有转义字符。 是否将 '#' 和 '+' 视为通配符取决于它们的上下文。 请参阅 第 176 页的『当基于主题的通配符无效时』，以了解更多信息。

注: 以特殊方式处理主题字符串的开头和结尾。 使用 '$' 来表示字符串的结尾，那么 '$#/...' 是多级通配符， '$/#/..'。 是根处的空节点，后跟多级通配符。

基于主题的通配符字符串

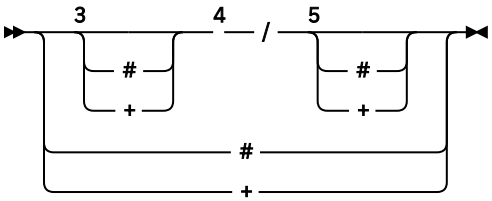


正文



主题级别名称

仅定界符

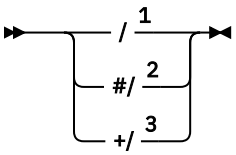


注:

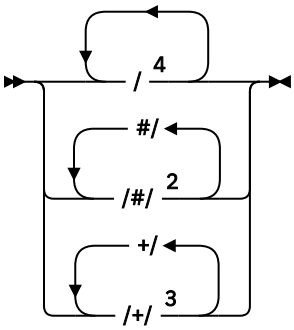
[1] 空或零长度主题字符串无效

[2] 建议您不要在级别名称字符串中使用任何 *, ?, %，以实现基于字符的通配符方案与基于主题的通配符方案之间的兼容性。
[3] 这些个案等同于 左定界符 模式。
[4] 没有通配符的 / 与单个空主题匹配。
[5] 这些个案等同于 右定界符 模式。
[6] 匹配每个主题。
[7] 匹配每个只有一个级别的主题。

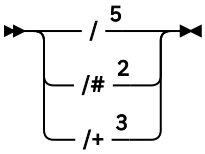
左定界符



主体定界符



右定界符



注:

[1] 主题字符串以空主题开头
[2] 与零个或多个级别匹配。 多个多级匹配字符串与一个多级匹配字符串具有相同的效果。
[3] 正好匹配一个级别。
[4] // 是空主题-没有主题字符串的主题对象。
[5] 主题字符串以空主题结尾

当基于主题的通配符无效时

当通配符 '+' 和 '#' 与主题级别中的其他字符 (包括它们本身) 混合时，它们没有特殊含义。

这意味着可以发布主题级别中包含 '+' 或 '#' 以及其他字符的主题。

例如以下两个主题:

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

在第一个示例中，字符 '+' 和 '#' 被视为通配符，因此在要发布到的主题字符串中无效，但在预订中有效。

在第二个示例中，不会将字符 '+' 和 '#' 视为通配符，因此可以发布和预订主题字符串。

示例

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

基于字符的通配符方案
基于字符的通配符方案允许您根据传统字符匹配来选择主题。

您可以使用字符串 '*'在主题层次结构中选择多个级别的所有主题。 在基于字符的通配符方案中使用 '*' 等同于使用基于主题的通配符字符串 '#'

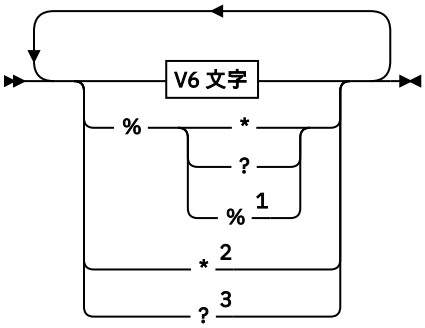
' *x*/*/*y* ' 等同于基于主题的方案中的 ' *x*/#/*y* '，并在级别 ' *x* 和 *y* '之间选择主题层次结构中的所有主题，其中 ' *x* ' 和 ' *y* ' 是不在通配符返回的级别集中的主题名称。

基于主题的方案中的 '/+/' 在基于字符的方案中没有完全相同的内容。 'IBM/*/Results' 还将选择 'IBM/Patents/Software/Results'。 仅当层次结构的每个级别的主题名称集都是唯一的时，才能始终使用生成相同匹配项的两个方案来构造查询。

以一般方式使用，基于字符的方案中的 '*' 和 '?' 在基于主题的方案中没有等效项。 基于主题的方案不会使用通配符执行部分匹配。 基于字符的通配符预订 'IBM/*ware/Results' 没有基于主题的等效项。

注: 使用字符通配符预订的匹配比使用基于主题的预订的匹配慢。

基于字符的通配符字符串



V6 文字

注:

[1] 表示对以下字符进行转义，以便将其视为文字。 '%' 必须后跟 '*'， '?' 或 '%'。 请参阅第 174 页的『主题字符串示例』。
[2] 表示与预订中的零个或多个字符匹配。
[3] 表示与预订中的一个字符完全匹配。

示例

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

组合主题字符串

创建预订或打开主题以使您可以向其发布消息时，可以通过组合两个单独的子主题字符串 (即 “子主题”) 来构成主题字符串。 一个子主题由应用程序或管理命令作为主题字符串提供，另一个子主题是与主题对象相关联的主题字符串。 您可以单独使用子主题作为主题字符串，也可以将其组合以形成新的主题名称。

例如，使用 MQSC 命令 **DEFINE SUB** 定义预订时，该命令可以将 **TOPICSTR** (主题字符串) 和/或 **TOPICOBJ** (主题对象) 作为属性。 如果仅提供了 **TOPICOBJ**，那么与该主题对象关联的主题字符串将用作主题字符

串。 如果仅提供了 **TOPICSTR**，那么将用作主题字符串。 如果同时提供了这两个字符串，那么将它们并置以形成格式为 **TOPICOBJ** / **TOPICSTR** 的单个主题字符串，其中 **TOPICOBJ** 配置的主题字符串始终是第一个，并且该字符串的两个部分始终以 “/” 字符分隔。

同样，在 MQI 程序中，完整主题名称由 MQOPEN 创建。 它由发布/预订 MQI 调用中使用的两个字段组成，按列出的顺序:

1. 主题对象的 **TOPICSTR** 属性，在 **ObjectName** 字段中指定。
2. 定义应用程序提供的子主题的 **ObjectString** 参数。

生成的主题字符串将在 **ResObjectString** 参数中返回。

如果每个字段的第一个字符不是空白或空字符，并且字段长度大于零，那么这些字段被视为存在。 如果仅存在其中一个字段，那么将其用作主题名称。 如果两个字段都没有值，那么调用将失败，原因码为 MQRC_UNKNOWN_OBJECT_NAME; 如果完整主题名称无效，那么调用将失败 MQRC_TOPIC_STRING_ERROR 。

如果两个字段都存在，那么将在生成的组合主题名称的两个元素之间插入 “/” 字符。

第 178 页的表 7 显示了主题字符串并置的示例:

表 7: 主题字符串并置示例

主题对象的 TOPICSTR	由应用程序或 DEFINE SUB 命令提供的主题字符串	完整主题名称	注释
足球/苏格兰人	''	足球/苏格兰人	单独使用主题对象的 TOPICSTR。
''	足球/苏格兰人	足球/苏格兰人	单独使用 ObjectString/ TOPICSTR。
美式足球	评分	足球/苏格兰人	在并置点添加 “/” 字符。
美式足球	/得分	足球//苏格兰人	在两个字符串之间生成 "空节点"。 这与 “Football/Scores”不同。
/足球	评分	/足球/苏格兰人	主题以 "空节点" 开头。 这与 “Football/Scores” 不同。

“/” 字符被视为特殊字符，为第 179 页的『主题树』中的完整主题名称提供结构。 不得将 “/” 字符用于任何其他原因，因为主题树的结构受到影响。 主题 “/Football” 与主题 “Football”不同。

注: 如果在创建预订时使用主题对象，那么主题对象主题字符串的值在定义时在预订中固定。 对主题对象的任何后续更改都不会影响预订所定义到的主题字符串。

主题字符串中的通配符

以下通配符是特殊字符:

- 加号 (+)
- 编号符号 (#)
- 星号 (*)
- 问号 (?)

仅当预订使用通配符时，通配符才具有特殊含义。 在其他位置使用时，这些字符不会被视为无效，但是您必须确保了解如何使用这些字符，并且在发布或定义主题对象时，您可能不希望在主题字符串中使用这些字符。

如果在主题级别中使用 # 或 + 与其他字符 (包括其本身) 混合的主题字符串进行发布，那么可以使用通配符方案来预订该主题字符串。

如果发布主题字符串时使用 # 或 + 作为两个 / 字符之间的唯一字符，那么应用程序无法使用通配符方案 MQSO_WILDCARD_TOPIC 显式预订主题字符串。 这种情况会导致应用程序获得比预期更多的发布。

不应在定义的主题对象的主题字符串中使用通配符。 如果执行此操作，那么当发布程序使用对象时，该字符将被视为字面值字符，当预订使用对象时，该字符将被视为通配符。 这会导致混乱。

示例代码片段

从示例程序 Example 2: Publisher to a variable topic 中抽取的此代码片段将主题对象与变量主题字符串组合在一起:

```
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);
```

主题树

您定义的每个主题都是主题树中的一个元素或节点。 主题树可以为空以开始，也可以包含先前使用 MQSC 或 PCF 命令定义的主题。 通过使用“创建主题”命令或通过指定主题，可以在发布或预订中首次定义新的主题。

虽然可以使用任何字符串来定义主题的主题字符串，但建议选择适合分层树结构的主题字符串。 经过深思熟虑的主题剌和主题树设计可以帮助您执行以下操作:

- 预订多个主题。
- 建立安全策略。

虽然您可以将主题树构造为平面的线性结构，但最好在具有一个或多个根主题的分层结构中构建主题树。 有关安全规划和主题的更多信息，请参阅 发布/预订安全性。

第 179 页的图 18 显示了具有一个根主题的主题树的示例。

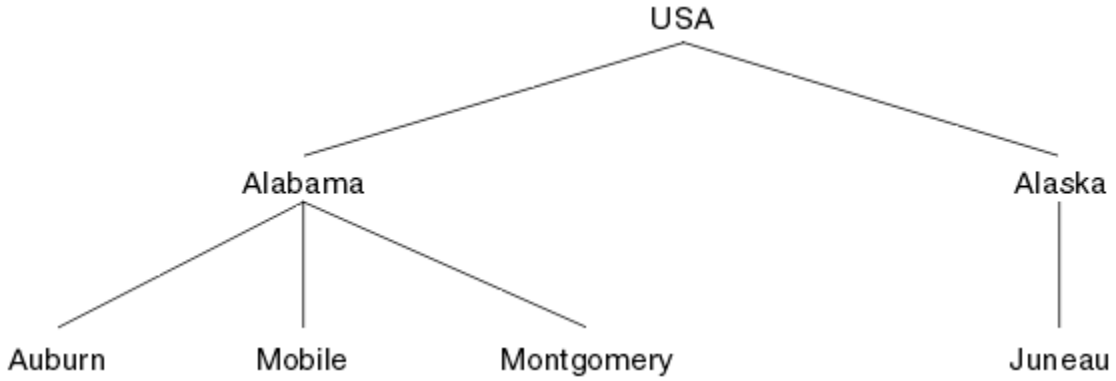


图 *18:* 主题树的示例

图中的每个字符串表示主题树中的一个节点。 通过聚集主题树中一个或多个级别的节点来创建完整的主题字符串。 级别由 "/" 字符分隔。 完全指定的主题字符串的格式为: "root/level2/level3"。

第 179 页的图 18 中显示的主题树中的有效主题包括:

"美国"
"美国/阿拉巴马州"
"美国/阿拉斯加"
"USA/Alabama/Auburn"
"USA/Alabama/Mobile"
"USA/Alabama/Montgomery"

"USA/Alaska/Juneau"

在设计主题字符串和主题树时，请记住队列管理器不会解释或尝试派生主题字符串本身的含义。 它只是使用主题字符串将所选消息发送到该主题的订户。

下列原则应用到主题树的构造和内容：

- 对主题树中级别的数目没有限制。
- 对主题树中级别的名称长度没有限制。
- 对“根”节点的数量无限制，即对主题树的数量无限制。

相关信息

减少主题树中不需要的主题数

管理主题对象

通过使用管理主题对象，可以将特定非缺省属性分配给主题。

第 180 页的图 19 显示了如何将划分为不同体育项目的不同主题的 Sport 高级别主题可视化为主题树：

图 *19:* 主题树的可视化

第 180 页的图 20 显示了如何进一步划分主题树，以分隔有关每项运动的不同类型的信息：

图 *20:* 扩展主题树

要创建说明的主题树，不需要定义任何管理主题对象。 此树中的每个节点都由在发布或预订操作中创建的主题字符串定义。 树中的每个主题都从其父代继承其属性。 属性继承自父主题对象，因为缺省情况下，所有属性都设置为 ASPARENT。 在此示例中，每个主题都具有与 Sport 主题相同的属性。 Sport 主题没有管理主题对象，并且从 SYSTEM.BASE.TOPIC。

请注意，在主题树的根节点 (即 SYSTEM.BASE.TOPIC，因为权限是继承的，但不能限制。 因此，通过授予此级别的权限，您将授予整个树的权限。 您应该在层次结构中的较低主题级别授予权限。

管理主题对象可用于定义主题树中特定节点的特定属性。 在以下示例中，定义了管理主题对象以将足球主题的持久预订属性 DURSUB 设置为值 NO：

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

现在可以将主题树可视化为：

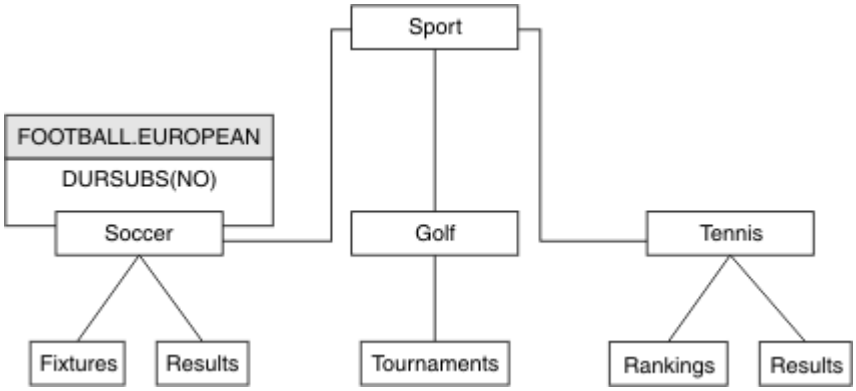


图 *21:* 与 *Sport/Soccer* 主题关联的管理主题对象的可视化

预订树中 Soccer 下的主题的任何应用程序仍可以使用它们在添加管理主题对象之前使用的主题字符串。 但是，现在可以使用对象名 FOOTBALL.EUROPEAN(而不是字符串 /Sport/Soccer) 来编写应用程序以进行预订。 例如，要预订 /Sport/Soccer/Results，应用程序可以将 MQSD.ObjectName 指定为 FOOTBALL.EUROPEAN，将 MQSD.ObjectString 指定为 Results。

通过此功能，您可以向应用程序开发者隐藏主题树的部分内容。 在主题树中的特定节点上定义管理主题对象，然后应用程序开发者可以将自己的主题定义为该节点的子代。 开发者必须了解父主题，但不能了解父树中的任何其他节点。

继承属性

如果主题树具有许多管理主题对象，那么缺省情况下，每个管理主题对象都将从其最接近的父管理主题继承其属性。 先前的示例已在 第 181 页的图 22 中扩展:

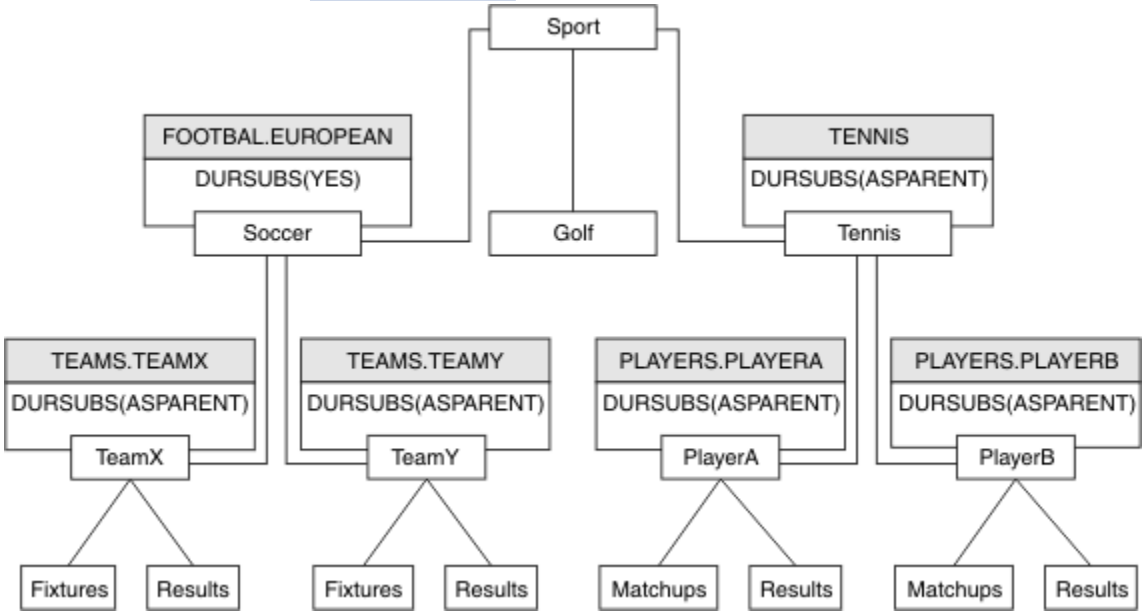


图 *22:* 具有多个管理主题对象的主题树

例如，使用继承为 /Sport/Soccer 的所有子主题提供预订为非持久预订的属性。 将 FOOTBALL.EUROPEAN 的 DURSUB 属性更改为 NO。

可使用以下命令设置此属性:

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

Sport/Soccer 的子主题的所有管理主题对象都将属性 DURSUB 设置为缺省值 ASPARENT。 将 FOOTBALL.EUROPEAN 的 DURSUB 属性值更改为 NO 后， Sport/Soccer 的子主题将继承 DURSUB 属性值

NO。 Sport/Tennis 的所有子主题从 SYSTEM.BASE.TOPIC 对象继承 DURSUB 的值。 SYSTEM.BASE.TOPIC 的值为 YES。

尝试对主题 Sport/Soccer/TeamX/Results 进行持久预订现在将失败; 但是，尝试对 Sport/Tennis/PlayerB/Results 进行持久预订将成功。

使用 通配符 属性控制通配符使用

使用 MQSC **Topic** 通配符 属性或等效 PCF 主题 WildcardOperation 属性来控制向使用通配符主题字符串名称的订户应用程序传递发布内容。 通配符 属性可以具有以下两个可能的值之一:

WILDCARD
与此主题有关的通配符预订的行为。

PASSTHRU
对没有此主题对象中主题字符串具体的通配主题进行的预订将接收对此主题以及比此主题更具体的主题字符串进行的发布。

BLOCK
对没有此主题对象中主题字符串具体的通配主题进行的预订不会接收对此主题或比此主题更具体的主题字符串进行的发布。

在定义预订时将使用此属性的值。 如果改变此属性，那么现有预订涵盖的主题集不会因为此修改而受到影响。 如果在创建或删除主题对象时拓扑发生更改，此场景也适用；将使用修改后的拓扑来创建与修改 WILDCARD 属性后创建的预订匹配的主题集。 如果要针对现有预订强制重新评估匹配的主题集，那么必须重新启动队列管理器。

在示例 第 185 页的『示例: 创建 Sport 发布/预订集群』 中，您可以遵循步骤来创建 第 182 页的图 23 中显示的主题树结构。

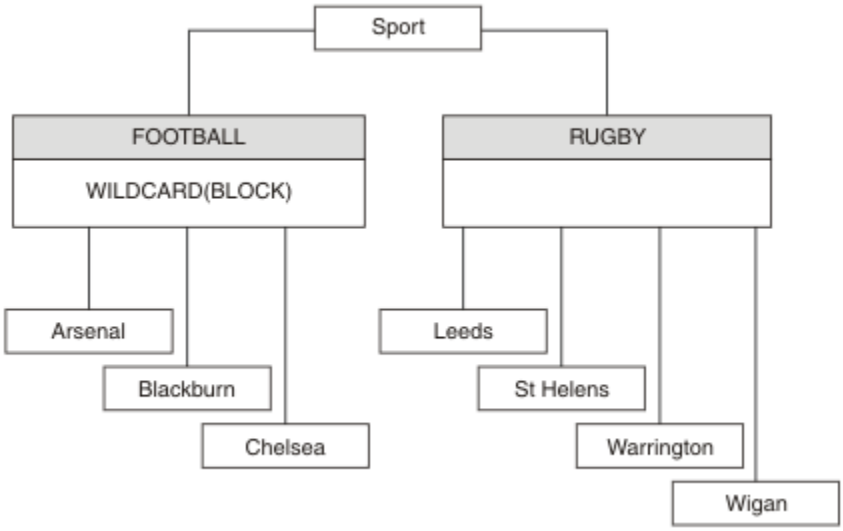


图 *23:* 使用 通配符 属性 *BLOCK* 的主题树

使用通配符主题字符串 # 的订户将接收到 Sport 主题和 Sport/Rugby 子树的所有发布。 订户不会接收到 Sport/Football 子树的发布，因为 Sport/Football 主题的 **通配符** 属性值为 BLOCK。

PASSTHRU 是缺省设置。 您可以将 **通配符** 属性值 PASSTHRU 设置为 Sport 树中的节点。 如果节点没有 **通配符** 属性值 BLOCK，那么设置 PASSTHRU 不会改变 Sports 树中节点的订户观察到的行为。

在此示例中，创建预订以查看通配符设置如何影响交付的发布; 请参阅 第 186 页的图 27。 在 第 187 页的图 30 中运行发布命令以创建一些发布。

```
pub QMA
```

图 *24:* 发布到 *QMA*

第 183 页的表 8 显示了结果。 请注意，如何设置 **通配符** 属性值 BLOCK，以防止具有通配符的预订将发布内容接收到通配符作用域内的主题。

表 *8: QMA* 上收到的出版物			
预订	**主题字符串**	**收到的出版物**	**注意**
SPORTS	Sports/#	Sports Sports/Rugby Sports/Rugby/Leeds	WILDCARD(BLOCK) 在 Sports/Football 上阻止了对 Football 子树的所有发布
SARSENAL	Sports/#/Arsenal	-	Sports/Football 上的 WILDCARD(BLOCK) 会阻止 Arsenal 上的通配符预订
SLEEDS	Sports/#/Leeds	Sports/Rugby/Leeds	Sports/Rugby 上的缺省 WILDCARD 不会阻止 Leeds 上的通配符预订。

注:

假设预订具有通配符，该通配符与具有 **通配符** 属性值 BLOCK 的主题对象相匹配。 如果预订还具有匹配通配符右侧的主题字符串，那么预订将从不接收发布内容。 未被阻止的发布集合是作为被阻止通配符的父代的主题的发布。 通配符将阻止发布到作为具有 BLOCK 属性值的主题的子代的主题。 因此，包含通配符右侧主题的预订主题字符串永远不会收到任何要匹配的发布内容。

将 WILDCARD 属性值设置为 BLOCK 并不意味着您无法使用包含通配符的主题字符串进行预订。 这样的订阅是正常的。 预订具有一个显式主题，该主题与具有 **通配符** 属性值 BLOCK 的主题对象匹配。 它将通配符用于作为具有 **通配符** 属性值 BLOCK 的主题的父代或子代的主题。 在 第 182 页的图 23 中的示例中，预订 (例如 Sports/Football/#) 可以接收发布。

通配符和集群主题

集群主题定义将传播到集群中的每个队列管理器。 在集群中的一个队列管理器上预订集群主题会导致队列管理器创建代理预订。 将在集群中的每个其他队列管理器上创建代理预订。 使用包含通配符的主题字符串 (与集群主题结合使用) 的预订可能难以预测行为。 此行为在以下示例中进行了说明。

在为示例 第 185 页的『示例: 创建 Sport 发布/预订集群』设置的集群中， QMB 具有与 QMA 相同的预订集，但 QMB 在发布者发布到 QMA 之后未收到任何发布，请参阅 第 182 页的图 24。 虽然 Sports/Football 和 Sports/Rugby 主题是集群主题，但 fullsubs.tst 中定义的预订不会引用集群主题。 不会将代理预订从 QMB 传播到 QMA。 如果没有代理预订，那么不会将 QMA 的任何发布转发到 QMB。

某些预订 (例如 Sports/#/Leeds) 可能似乎引用了集群主题，在本例中为 Sports/Rugby。 Sports/#/Leeds 预订实际解析为主题对象 SYSTEM.BASE.TOPIC。

用于解析预订 (例如， Sports/#/Leeds) 所引用的主题对象的规则如下所示。 将主题字符串截断为第一个通配符。 通过主题字符串向左扫描以查找具有关联管理主题对象的第一个主题。 主题对象可以指定集群名称，也可以定义本地主题对象。 在示例 Sports/#/Leeds 中，截断后的主题字符串是 Sports，它没有主题对象，因此 Sports/#/Leeds 继承自 SYSTEM.BASE.TOPIC(这是本地主题对象)。

要查看预订集群主题如何更改通配符传播的工作方式，请运行批处理脚本 upsubs.bat。 该脚本将清除预订队列，并在 fullsubs.tst 中添加集群主题预订。 再次运行 puba.bat 以创建一批出版物; 请参阅 第 182 页的图 24。

第 184 页的表 9 显示了将两个新预订添加到发布发布的同一队列管理器的结果。 结果与预期相同，新预订各接收一个发布，其他预订接收的发布数量保持不变。 意外结果发生在其他集群队列管理器上; 请参阅 第 184 页的表 10。

表 *9: QMA* 上收到的出版物

预订	主题字符串	收到的出版物	注意
SPORTS	Sports/#	Sports Sports/Rugby Sports/Rugby/Leeds	WILDCARD(BLOCK) 在 Sports/Football 上阻止了对 Football 子树的所有发布
SARSENAL	Sports/#/Arsenal	-	Sports/Football 上的 WILDCARD(BLOCK) 会阻止 Arsenal 上的通配符预订
SLEEDS	Sports/#/Leeds	Sports/Rugby/Leeds	Sports/Rugby 上的缺省 WILDCARD 不会阻止 Leeds 上的通配符预订。
FARSENAL	Sports/Football/Arsenal	Sports/Football/Arsenal	Arsenal 接收发布，因为预订没有通配符。
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds 将在任何情况下接收发布。

第 184 页的表 10 显示了在 QMB 上添加两个新预订以及在 QMA 上发布的结果。 请注意，在没有这两个新预订的情况下， QMB 未收到任何发布。 正如预期的那样，这两个新预订将接收发布内容，因为 Sports/FootBall 和 Sports/Rugby 都是集群主题。 QMB 将 Sports/Football/Arsenal 和 Sports/Rugby/Leeds 的代理预订转发到 QMA，然后将发布发送到 QMB。

意外的结果是，先前未收到任何发布的两个预订 Sports/# 和 Sports/#/Leeds 现在接收发布。 原因是针对其他预订转发到 QMB 的 Sports/Football/Arsenal 和 Sports/Rugby/Leeds 发布现在可用于连接到 QMB 的任何订户。 因此，本地主题 Sports/# 和 Sports/#/Leeds 的预订将接收 Sports/Rugby/Leeds 发布。 Sports/#/Arsenal 继续不接收发布内容，因为 Sports/Football 已将其 **通配符** 属性值设置为 BLOCK。

表 *10: QMB* 上收到的出版物

预订	主题字符串	收到的出版物	注意
SPORTS	*Sports/#*	*Sports/Rugby/Leeds*	*WILDCARD(BLOCK)* 在 *Sports/Football* 上阻止了对 *Football* 子树的所有发布
SARSENAL	Sports/#/Arsenal	-	Sports/Football 上的 WILDCARD(BLOCK) 会阻止 Arsenal 上的通配符预订
SLEEDS	*Sports/#/Leeds*	*Sports/Rugby/Leeds*	*Sports/Rugby* 上的缺省 *WILDCARD* 不会阻止 *Leeds* 上的通配符预订。
FARSENAL	Sports/Football/Arsenal	Sports/Football/Arsenal	Arsenal 接收发布，因为预订没有通配符。
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds 将在任何情况下接收发布。

在大多数应用程序中，不希望一个预订影响另一个预订的行为。 **通配符** 属性与值 BLOCK 的一个重要用途是使包含通配符的同一主题字符串的预订行为一致。 无论预订是在与发布程序相同的队列管理器上，还是在不同的队列管理器上，预订的结果都是相同的。

通配符和流

对于写入发布/预订 API 的新应用程序，结果是对 * 的预订不会收到任何发布。 要接收所有体育出版物，必须预订 Sports/*或 Sports/#以及类似的 Business 出版物。

将发布/预订代理迁移到 IBM MQ 版本 7 和更高版本时，现有排队的发布/预订应用程序的行为不会更改。 **Publish**， **Register Publisher** 或 **Subscriber** 命令中的 **StreamName** 属性将映射到流已迁移到的主题的名称。

通配符和预订点

对于写入发布/预订 API 的新应用程序，迁移的效果是，对 * 的预订不会收到任何发布。 要接收所有体育出版物，必须预订 Sports/*或 Sports/#以及类似的 Business 出版物。

将发布/预订代理迁移到 IBM MQ 版本 7 和更高版本时，现有排队的发布/预订应用程序的行为不会更改。 **Publish**， **Register Publisher** 或 **Subscriber** 命令中的 **SubPoint** 属性将映射到预订已迁移到的主题的名称。

示例: 创建 Sport 发布/预订集群

后续步骤将创建一个具有四个队列管理器的集群 CL1: 两个完整存储库 CL1A 和 CL1B 以及两个部分存储库 QMA 和 QMB。 完整存储库仅用于保存集群定义。 QMA 被指定为集群主题主机。 持久预订同时在 QMA 和 QMB 上定义。

注: 此示例针对 Windows 进行编码。 您必须重新编码 Create qmgrs.bat 和 create pub.bat 以在其他平台上配置和测试示例。

1. 创建脚本文件。
   a. 创建 topics.tst
   b. 创建 wildsubs.tst
   c. 创建 fullsubs.tst
   d. 创建 qmgrs.bat
   e. 创建 pub.bat
2. 运行 Create qmgrs.bat 以创建配置。

```
qmgrs
```

在 第 182 页的图 23 中创建主题。 图 5 中的脚本将创建集群主题 Sports/Football 和 Sports/Rugby。

注: REPLACE 选项不会替换主题的 TOPICSTR 属性。 TOPICSTR 是在示例中进行有效更改以测试不同主题树的属性。 要更改主题，请先删除该主题。

```
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')
```

图 *25:* 删除并创建主题*: topics.tst*

注: 删除主题，因为 REPLACE 不会替换主题字符串。

使用通配符创建预订。 通配符将主题与 第 182 页的图 23 中的主题对象相对应。 为每个预订创建一个队列。 运行或重新运行脚本时，将清除队列并删除预订。

注: REPLACE 选项不会替换预订的 TOPICOBJ 或 TOPICSTR 属性。 TOPICOBJ 或 TOPICSTR 是在用于测试不同预订的示例中进行有效更改的属性。 要对其进行更改，请先删除预订。

```
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)
```

图 *26:* 创建通配符预订*: wildsubs.tst*

创建引用集群主题对象的预订。

注:

将在 TOPICOBJ 引用的主题字符串与 TOPICSTR 定义的主题字符串之间自动插入定界符 /。

定义 DEFINE SUB(FARSENAL) TOPICSTR('Sports/Football/Arsenal') DEST(QFARSENAL) 将创建相同的预订。 TOPICOBJ 用作引用您已定义的主题字符串的快速方法。 创建预订时，不再引用主题对象。

```
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)
```

图 *27:* 删除并创建预订*: fullsubs.tst*

创建具有两个存储库的集群。 创建两个用于发布和预订的部分存储库。 重新运行脚本以删除所有内容，然后再次启动。 该脚本还会创建主题层次结构和初始通配符预订。

注:

在其他平台上，编写类似的脚本，或者输入所有命令。 通过使用脚本，可以快速删除所有内容，并使用相同的配置重新开始。

```
@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%)')
CLUSTER(CL1) REPLACE | runmqsc %1
goto:eof
```

图 *28:* 创建队列管理器*: qmgrs.bat*

通过将预订添加到集群主题来更新配置。

```
@echo off
for %%A in (QMA QMB) do runmqsc %%A < wildsubs.tst
for %%A in (QMA QMB) do runmqsc %%A < upsubs.tst
```

图 *29:* 更新预订*: upsubs.bat*

运行以队列管理器作为参数的 pub.bat，以发布包含发布主题字符串的消息。 Pub.bat 使用样本程序 **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
```

图 *30:* 发布*: pub.bat*

流和主题

已排队的发布/预订具有集成发布/预订模型中不存在的发布流的概念。 在已排队的发布/预订中，流提供了一种分隔不同主题的信息流的方法。 从 IBM WebSphere MQ 7.0 开始，流作为顶级主题实现，可通过管理方式映射到其他主题标识。

将为网络上的所有代理和队列管理器自动设置缺省流 SYSTEM.BROKER.DEFAULT.STREAM，并且无需其他配置即可使用缺省流。 将缺省流视为未命名的缺省主题空间。 从 版本 7.0 开始，发布到缺省流的主题立即

可供所有已连接的队列管理器使用，并且已启用排队的发布/预订。 指定的流类似于单独的，指定的主题空间。 必须在使用指定流的每个代理上定义该流。

如果发布者和订户位于不同的队列管理器上，那么在同一代理层次结构中连接代理之后，不需要进一步配置这些发布以及预订之间的流。 相同的互操作性也会逆向工作。

指定的流

解决方案设计人员在使用排队的发布/预订编程模型时，可能会决定将所有体育出版物放入名为 Sport 的指定流中。 在 版本 6.0 中，流通常会自动复制到使用模型队列 SYSTEM.BROKER.MODEL.STREAM 的其他代理上。 但是，要使流可供从 版本 7.0 开始运行且已启用排队的发布/预订的队列管理器使用，必须手动添加该流。

在流 Sport 上预订 Soccer/Results 的已排队发布/预订应用程序工作而不进行更改。 使用 MQSUB 预订主题 Sport 并提供主题字符串 Soccer/Results 的集成发布/预订应用程序也会接收相同的发布。

添加流主题中描述了添加流的任务。 由于两个原因，您可能需要手动添加流。

1. 继续开发在更高版本队列管理器上运行的已排队发布/预订应用程序，而不是将这些应用程序迁移到集成发布/预订 MQI 接口。
2. 流到主题的缺省映射会导致主题空间中的 "冲突"，并且流上的发布与来自其他位置的发布具有相同的主题字符串。

权限

缺省情况下，在主题树的根目录中有多个主题对象: SYSTEM.BASE.TOPIC, SYSTEM.BROKER.DEFAULT.STREAM 和 SYSTEM.BROKER.DEFAULT.SUBPOINT。 权限 (例如，用于发布或预订) 由 SYSTEM.BASE.TOPIC 上的权限确定; 将忽略 SYSTEM.BROKER.DEFAULT.STREAM 或 SYSTEM.BROKER.DEFAULT.SUBPOINT 上的任何权限。 如果删除 SYSTEM.BROKER.DEFAULT.STREAM 或 SYSTEM.BROKER.DEFAULT.SUBPOINT 中的任何一个并使用非空主题字符串重新创建，那么将以与普通主题对象相同的方式使用对这些对象定义的权限。

流与主题之间的映射

从 版本 7.0 开始，通过创建队列并为其提供与流相同的名称来模拟已排队的发布/预订流。 有时该队列被称为流队列，因为这就是它在排队的发布/预订应用程序中的显示方式。 通过将队列添加到名为 SYSTEM.QPUBSUB.QUEUE.NAMELIST 的特殊名称列表，可将该队列标识到发布/预订引擎。 通过向名称列表添加其他特殊队列，可以根据需要添加任意数量的流。 最后需要添加主题，名称与流相同，主题字符串与流名称相同，因此可以发布和预订主题。

但是，在特殊情况下，您可以为与流对应的主题提供您在定义主题时选择的任何主题字符串。 主题字符串的目的是在主题空间中为主题提供唯一的名称。 通常，流名称完全满足此目的。 有时，流名称与现有主题名称冲突。 要解决此问题，请为与流关联的主题选择另一主题字符串。 选择任何主题字符串，确保其唯一。

主题定义中定义的主题字符串以常规方式作为发布程序和订户使用 MQOPEN 或 MQSUB MQI 调用提供的主题字符串的前缀。 使用主题对象引用主题的应用程序不受前缀主题字符串选项的影响-因此，您可以选择任何在主题空间中保持出版物唯一的主题字符串。

将不同流重新映射到不同主题依赖于用于唯一主题字符串的前缀，以将一组主题与另一组主题完全分开。 您必须定义严格遵循的通用主题命名约定，才能使映射生效。

在 版本 7.0 中，如果主题字符串发生冲突，那么可以使用流来分隔主题空间。

从 版本 7.0 开始，您使用前缀机制将主题字符串重新映射到主题空间中的另一个位置。

注: 删除流时，请先删除该流上的所有预订。 如果任何预订源自代理层次结构中的其他代理，那么此操作最为重要。

示例

在 第 189 页的图 31 中，主题 'Sport' 具有主题字符串 'xyz'，导致源自流 'Sport' 的发布以版本 7 队列管理器主题空间中的字符串 'xyz' 作为前缀。 正在将 V 7 中的主题 'Sport' 前缀 'xyz' 发布或预订到主题字符串。 如果发布流向 版本 6 订户，那么将从发布中除去前缀 'xyz'，并将其放置在 'Sport' 流

中。 相反，当发布从 版本 6 流到 V 7 ，从 'Sport' 流到 'Sport' 主题时，会将前缀 'xyz' 添加到主题字符串。

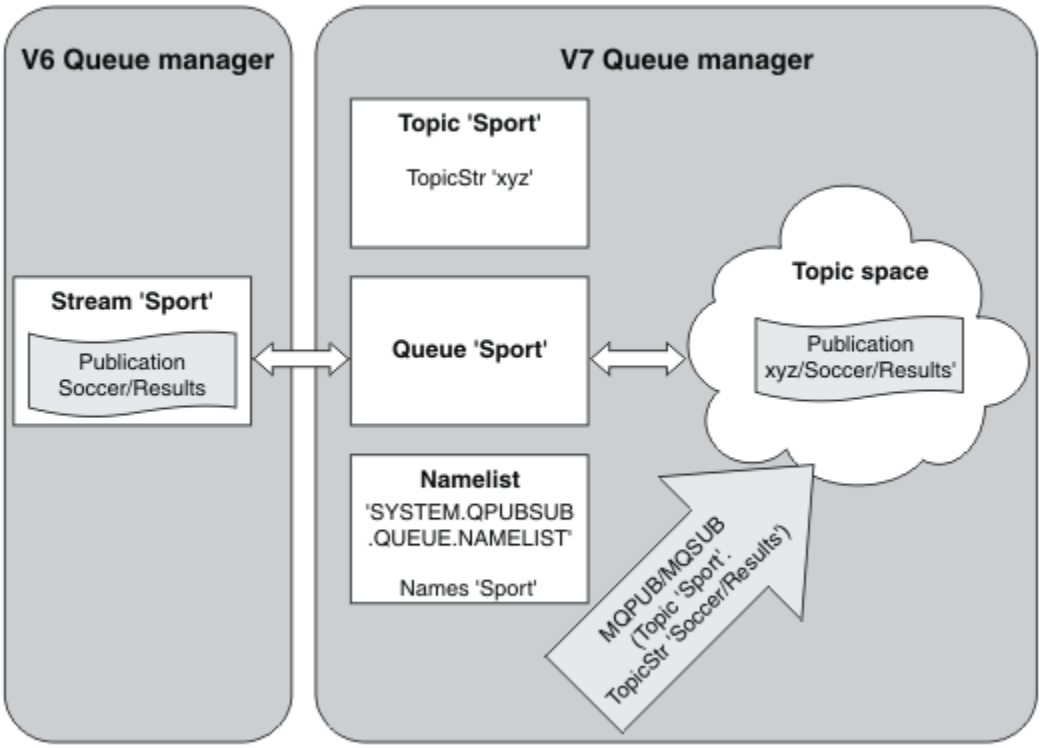


图 *31: 与 V 7* 主题共存的 版本 *6* 流

预订点和主题

指定的预订点由主题和主题对象进行仿真。

要手动添加预订点，请参阅 添加预订点。

IBM MQ 中的预订点

IBM MQ 将预订点映射到 IBM MQ 主题树中的不同主题空间。 没有预订点的命令消息中的主题将以未更改的方式映射到 IBM MQ 主题树的根，并从 SYSTEM.BASE.TOPIC 继承属性。

具有预订点的命令消息正在使用 SYSTEM.QPUBSUB.SUBPOINT.NAMELIST 中的主题对象列表进行处理。 命令消息中的预订点名称与列表中每个主题对象的主题字符串相匹配。 如果找到匹配项，那么会将预订点名称作为主题节点附加到主题字符串。 主题从 SYSTEM.QPUBSUB.SUBPOINT.NAMELIST 中找到的关联主题对象继承其属性。

使用预订点的效果是为每个预订点创建单独的主题空间。 主题空间植根于与预订点同名的主题中。 每个主题空间中的主题从与预订点同名的主题对象继承其属性。

将以正常方式从 SYSTEM.BASE.TOPIC 继承匹配主题对象中未设置的任何属性。

现有已排队的发布/预订应用程序使用 MQRFH2 消息头，通过在 Publish 或 Register subscriber 命令消息中设置 **SubPoint** 属性来继续工作。 预订点与命令消息中的主题字符串组合在一起，并与任何其他主题一样处理生成的主题。

IBM WebSphere MQ 7.0 或更高版本的应用程序不受预订点影响。 如果应用程序使用从其中一个匹配主题对象继承信息的主题，那么该应用程序将使用匹配的预订点与排队的应用程序进行互操作。

示例

现有 WebSphere Message Broker (现在称为 IBM Integration Bus) 迁移到 IBM MQ 的发布/预订应用程序创建了两个主题对象 GBP 和 USD 以及相应的主题字符串 'GBP' 和 'USD'。

已迁移到 IBM MQ 上运行的主题 NYSE/IBM/SPOT 的现有发布程序，这些发布程序使用预订点 USD 在主题 USD/NYSE/IBM/SPOT 上创建发布。 与 NYSE/IBM/SPOT 的现有订户类似，使用预订点 USD 创建对 USD/NYSE/IBM/SPOT 的预订。

在 IBM WebSphere MQ 7.0 或更高版本的发布/预订程序中，通过调用 MQSUB 来预订美元现货价格。 使用 USD 主题对象和主题字符串 'NYSE/IBM/SPOT'创建预订，如 "C" 代码片段中所示。

```
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. 在集群主题主机上设置 USD 和 GBP 主题对象的 CLUSTER 属性。
2. 删除集群中其他队列管理器上 USD 和 GBP 主题对象的所有副本。
3. 确保在集群中的每个队列管理器上的 SYSTEM.QPUBSUB.SUBPOINT.NAMELIST 中定义了 USD 和 GBP 。

单个队列管理器发布/预订配置的示例

第 190 页的图 32 说明了基本的单个队列管理器发布/预订配置。 此示例显示了新闻服务的配置，其中发布者提供了有关多个主题的信息:

- 发布程序 1 正在使用 "运动" 主题发布有关运动结果的信息
- 发布程序 2 正在使用 "股票" 主题发布有关股票价格的信息
- 发布者 3 正在使用 "电影" 主题发布有关电影评论的信息，并使用 "电视" 主题发布有关电视列表的信息

三个订户已注册了对不同主题的兴趣，因此队列管理器会向他们发送他们感兴趣的信息:

- 订户 1 接收体育结果和股价
- 订户 2 接收影片评论
- 订户 3 接收体育结果

没有一个用户登记了对电视列表的兴趣，所以这些都没有分发。

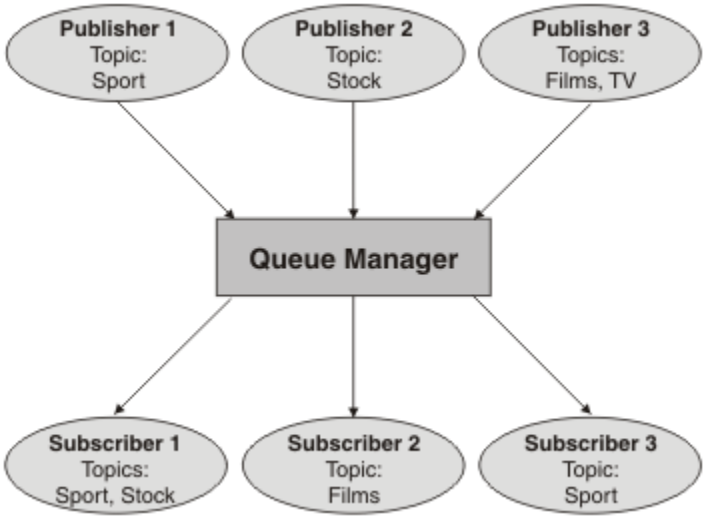


图 *32:* 单个队列管理器发布/预订示例

分布式发布/预订网络

每个队列管理器将发布到主题的消息与本地创建的预订 (已预订该主题) 相匹配。 您可以配置队列管理器网络，以便将连接到一个队列管理器的应用程序发布的消息传递到网络中其他队列管理器上创建的匹配预订。 这需要通过队列管理器之间的简单通道进行其他配置。

分布式发布/预订配置是一组连接在一起的队列管理器。 队列管理器可以全部位于同一物理系统上，也可以分布在多个物理系统上。 将队列管理器连接在一起时，订户可以预订一个队列管理器，并接收最初发布到另一个队列管理器的消息。 为了说明这一点，下图向 第 190 页的『单个队列管理器发布/预订配置的示例』中描述的配置添加了第二个队列管理器。

- 队列管理器 2 由发布者 4 使用 "天气" 主题发布天气预报信息，并使用 "交通" 主题发布有关主要道路交通状况的信息。
- 订户 4 还使用此队列管理器，并使用主题 "流量" 预订有关流量条件的信息。
- 订户 3 还预订有关天气状况的信息，即使它使用与发布程序不同的队列管理器也是如此。 这是可能的，因为队列管理器相互链接。

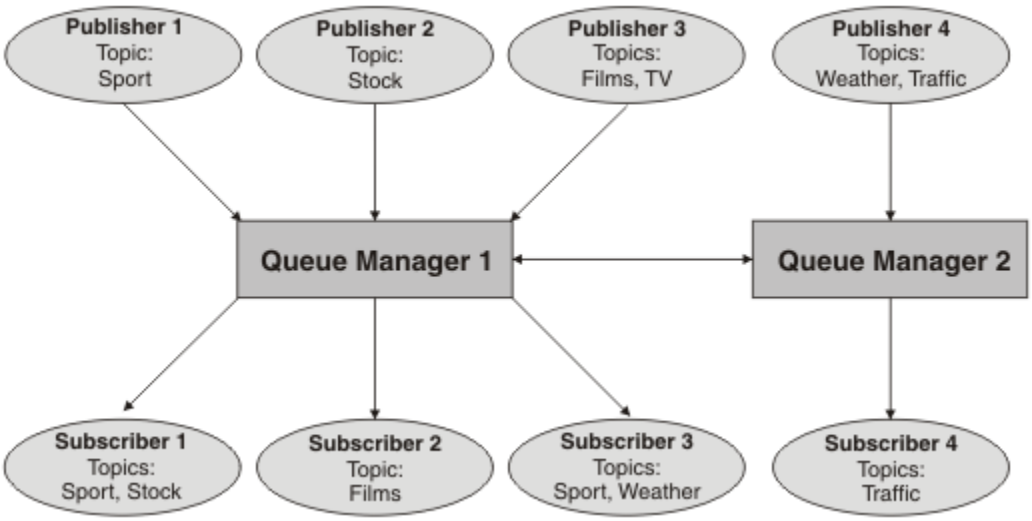


图 *33:* 具有两个队列管理器的发布/预订示例

您可以手动连接父层次结构和子层次结构中的队列管理器，也可以创建发布/预订集群，并让 IBM MQ 为您定义许多连接详细信息。 您还可以组合使用这两种拓扑，例如，通过在层次结构中将多个集群连接在一起。

发布/预订集群概述

发布/预订集群是向集群添加了一个或多个主题对象的标准集群。 当您在集群中的任何队列管理器上定义 管理主题对象 ，并通过指定集群名称使该主题对象成为集群对象时，该主题的发布者和订户可以连接到集群中的任何队列管理器，并且发布的消息将通过队列管理器之间的集群通道路由到订户。

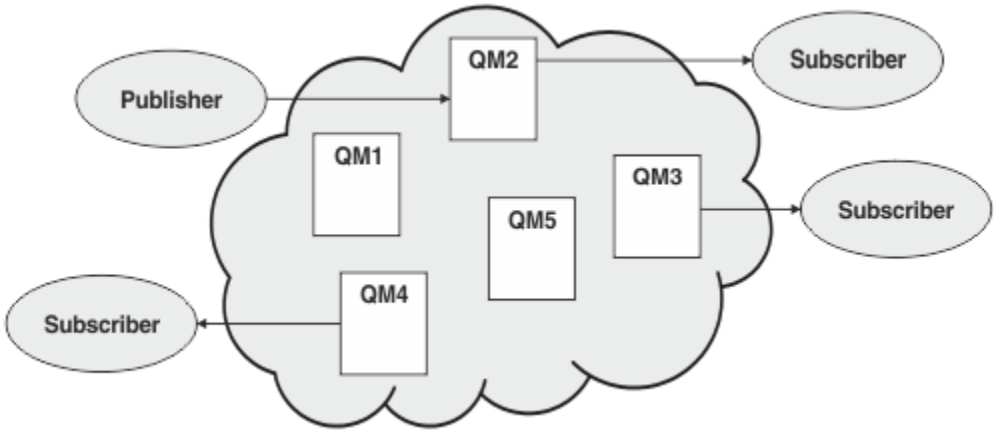


图 *34:* 发布/预订集群 *(publish/subscribe cluster)*

有两种方法可配置如何在集群中路由发布/预订消息:

- 直接路由 (direct routing)

- 主题主机路由 (topic host routing)

当您配置直接路由的集群主题时，在一个队列管理器上发布的消息将直接从该队列管理器发送到集群中任何其他队列管理器上的每个预订。 这可以为发布提供最直接的路径，但确实会导致集群中的所有队列管理器了解所有其他队列管理器，每个队列管理器都可能在它们之间建立集群通道。

使用主题主机路由时，在一个队列管理器上发布的消息将从该队列管理器发送到主管受管主题对象定义的队列管理器。 主题主机队列管理器会将消息路由至集群中其他任何队列管理器上的每个预订。 如果发布者或订户不在主题主机队列管理器上，那么这将导致发布的路径较长。 但是，好处是只有主题主机队列管理器才知道集群中的所有其他队列管理器，并且可能与它们一起建立了集群通道。

有关更多信息，请参阅第 193 页的『发布/预订集群』。

发布/预订层次结构概述

发布/预订层次结构是由通道连接到分层结构中的一组队列管理器。 每个队列管理器标识其 父 队列管理器，如 将队列管理器连接到发布/预订层次结构中所述。

主题的发布者和订户可以连接到层次结构中的任何队列管理器，并且消息使用分层队列管理器连接在它们之间流动。

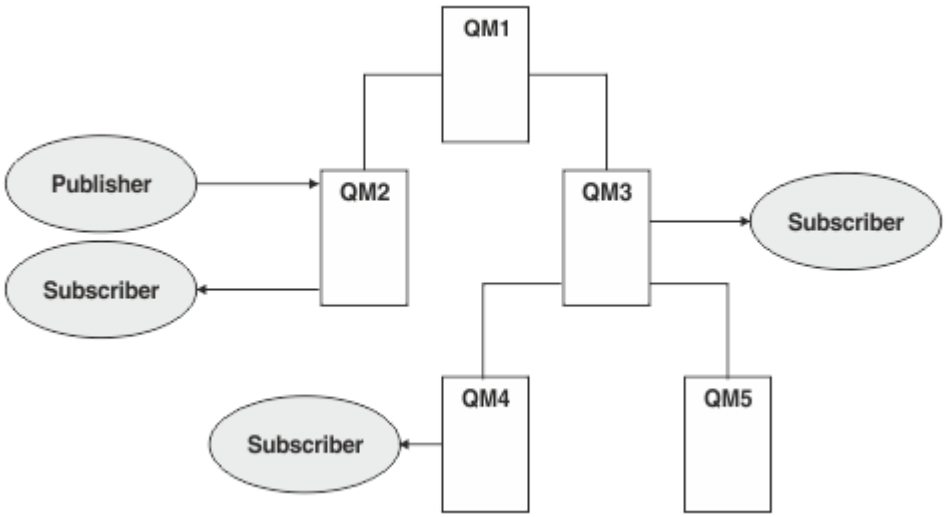


图 *35:* 发布/预订层次结构

在上图中，传递到 QM3 和 QM4 上的订户的出版物已从 QM2 路由到 QM1 ， 然后传递到 QM3 ， 最后传递到 QM4。

层次结构使您能够直接控制层次结构中每个队列管理器之间的关系。 这允许对从发布者到订户的消息路由进行细颗粒度控制，并且在具有受限连接的队列管理器网络之间进行路由时特别有用。 您应该仔细考虑每个队列管理器的可用性和功能，通过这些队列管理器将消息从发布者路由到订户。

有关更多信息，请参阅第 195 页的『发布/预订层次结构』。

队列管理器之间的发布分布

除了路由选项外，还有两种方法可以在队列管理器网络中分发发布内容:

- 仅将发布从一个队列管理器发送到当前主管该发布的预订的队列管理器。
- 将每个发布发送到所有队列管理器，并使其与预订相匹配。

前者导致仅在必要时发送发布消息，但需要在队列管理器之间共享一定级别的预订知识。 后者不需要共享预订知识，但会导致在队列管理器之间发送不必要的发布消息。

缺省情况下， IBM MQ 使用前一种方法，在此方法中，发布仅发送到具有预订的队列管理器。 预订知识以代理预订形式在队列管理器之间传播。 这取决于预订的分布和生存期，以及发布的频率，因为在分布式发布/预订拓扑中使用的效率最高。 请参阅 发布/预订网络中的预订性能。

相关概念

第 179 页的『主题树』
您定义的每个主题都是主题树中的一个元素或节点。 主题树可以为空以开始，也可以包含先前使用 MQSC 或 PCF 命令定义的主题。 通过使用“创建主题”命令或通过指定主题，可以在发布或预订中首次定义新的主题。

相关信息

设计发布/预订集群
发布/预订层次结构方案

发布/预订集群

发布/预订集群是互连队列管理器的标准集群，在该集群上，发布将自动从发布应用程序移至集群中任何队列管理器上存在的预订。 提供两个选项用于在发布/预订集群中路由发布：直接路由和主题主机路由。 您选择的路由取决于集群的大小和期望的活动模式。

用于发布/预订消息传递的集群与标准 IBM MQ 集群没有任何不同。 因此，发布/预订集群中的队列管理器可以存在于物理上独立的计算机上，必要时，集群通道会自动将每对队列管理器连接在一起。 有关更多信息，请参阅 集群。

要配置标准的队列管理器集群以进行发布/预订消息传递，可在集群中的一个队列管理器上定义一个或多个受管主题对象。 要使主题成为集群主题，请使用集群的名称配置 **CLUSTER** 属性。 执行此操作时， 主题树 中的发布者或订户在该点或以下位置使用的任何主题将在集群中的所有队列管理器之间共享，并且发布到主题树的集群分支的消息将自动路由到集群中其他队列管理器上的预订。

在发布者队列管理器和其他队列管理器之间仅发送每条消息的一个副本，而不考虑目标队列管理器上该消息的订户数。 到达具有一个或多个预订的队列管理器时，将在所有预订之间复制消息。

任何加入集群的队列管理器都会自动了解集群主题，并且该队列管理器上的发布者和订户会自动参与集群。

通过使用不属于集群主题对象的主题字符串，还可以在发布/预订集群中执行非集群发布/预订活动。

提供两个选项用于在发布/预订集群中路由发布：直接路由和主题主机路由。 要选择集群中使用的消息路由，可将受管主题对象上的 **CLROUTE** 属性设置为以下某个值：

- **DIRECT**
- **TOPICHOST**

缺省情况下，主题路由为 **DIRECT**。 这是 IBM MQ 8.0 之前的版本中的唯一选项。 当在队列管理器上配置直接路由集群主题时，集群中的所有队列管理器都可识别集群中的所有其他队列管理器。 在执行发布和预订操作时，每个队列管理器都可以直接连接到集群中的任何其他队列管理器。

从 IBM MQ 8.0 开始，您可以改为将主题路由配置为 **TOPICHOST**。 在您使用主题主机路由时，集群中的所有队列管理器都会知晓托管了路由主题定义的集群队列管理器（即，已定义主题对象的队列管理器）。 在执行发布和预订操作时，集群中的队列管理器只会连接到这些主题主机队列管理器，而不会彼此直接连接。 主题主机队列管理器负责将发布从执行发布的队列管理器路由至具有匹配预订的队列管理器。

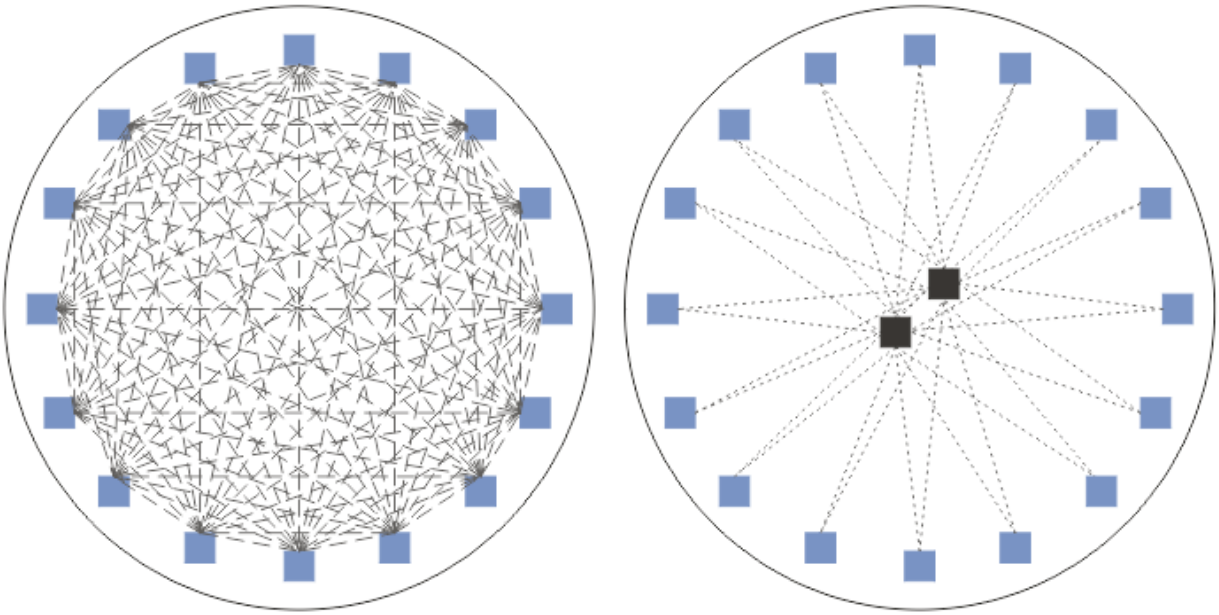


图 *36:* 直接路由和主题主机路由

直接路由概述

配置受管主题对象以进行直接路由时，只需要在集群中的其中一个队列管理器上定义主题对象，所有队列管理器才能了解该对象。 定义主题的队列管理器选项不会影响主题的发布/预订消息传递行为。

每条消息都直接从发布者队列管理器流向集群中其他队列管理器上的每个预订，而不是通过任何中间队列管理器传递。

缺省情况下，消息仅发送到集群中托管一个或多个预订的其他队列管理器。

- 这依赖于每个队列管理器直接向集群中的所有其他队列管理器通知当前具有一个或多个预订的所有主题。 这将导致集群中的所有队列管理器都知道正在预订的所有主题，以及托管预订的任何队列管理器都将建立到每个其他队列管理器的通道。 这与每个队列管理器是否具有发布程序无关。
- 可以通过更改为向集群中的所有队列管理器发送所有发布的模型 (无论它们是否具有预订) 来除去所有队列管理器上的每个单独预订主题的知识。 这将减少预订知识流量，但可能会增加发布流量以及每个队列管理器建立的通道数。 请参阅 发布/预订网络中的预订性能。

使用直接路由的集群主题的发布/预订消息流可以跨多个发布/预订集群，方法是将每个集群中的一个队列管理器添加到发布/预订层次结构中。 请参阅 组合多个集群的主题空间。

有关直接路由的更详细探索，请参阅 发布/预订集群中的直接路由。

主题主机路由概述

当为主题主机路由配置了受管主题对象时，来自集群中队列管理器的发布将通过配置了主题对象的队列管理器 ("主题主机") 进行路由，并从该队列管理器路由到存在预订的队列管理器。

- 这依赖于每个队列管理器向所有主题主机通知当前具有一个或多个预订的每个主题。 托管预订的任何队列管理器都会为与预订相关的主题建立到每个主题主机的通道。
- 出于发布/预订目的，不会使非主题托管队列管理器知道集群中的其他非主题托管队列管理器，并且不会为此目的在它们之间建立通道。
- 如果发布应用程序连接到托管主题的队列管理器，那么已发布的消息将直接路由到已创建匹配预订的队列管理器，而不需要额外的 "中继段"。 同样，如果在托管主题的唯一队列管理器上创建匹配的预订，那么发布到该主题的消息将直接路由到该队列管理器，而不需要额外的中继段。
- 将满足与发布程序相同的队列管理器上的预订，而不首先将发布路由到主题对象的主机。

对于集群队列，多个队列管理器可以配置相同的管理主题对象。 这将提供更高的消息路由可用性，并通过工作负载均衡进行水平缩放。 对于主题主机路由的主题对象，当多个队列管理器为主题树的同一分支配置同一指定主题时，托管预订的每个队列管理器将使每个主题主机了解预订的主题。

- 发布消息时，会将其发送到其中一个主题主机队列管理器，以转发到预订托管队列管理器。 主题主机队列管理器的选择遵循与集群点到点队列相同的缺省工作负载均衡规则。
- 如果发布队列管理器无法联系一个或多个主题主机队列管理器，那么会将消息路由到剩余的可用主题主管队列管理器。

主题树的路由分支中的主题的每个发布都将转发到其中一个主题主机，即使集群中的任何位置都没有该主题的预订也是如此。 缺省情况下，消息仅从此处发送到集群中托管一个或多个预订的其他队列管理器。

- 这依赖于向每个主题主机队列管理器通知集群中每个队列管理器上的所有预订主题字符串。
- 可以通过更改为将路由到主题主机的所有发布发送到集群中所有队列管理器的模型来除去每个单独预订的主题的知识，而不考虑它们是否具有预订。 这将减少预订知识流量，但可能会增加发布流量，并可能增加使用每个主题托管队列管理器建立的通道数。 请参阅 发布/预订网络中的预订性能。

使用主题主机路由的集群主题的发布/预订消息流 **不能** 通过使用发布/预订层次结构跨多个发布/预订集群。

有关主题主机路由的更详细探索，请参阅 发布/预订集群中的主题主机路由。

发布/预订层次结构

通过使用通道将队列管理器链接在一起，然后定义队列管理器对之间的子/父关系，构建发布/预订层次结构。 消息通过层次结构中的直接关系从发布者流向预订。 请注意，这可能意味着要到达多个 “中继段” 。

在任何一对队列管理器之间仅发送消息的一个副本，而不考虑目标队列管理器上消息的订户数。 到达具有一个或多个预订的队列管理器时，将在所有预订之间复制消息。

缺省情况下，仅将消息发送到层次结构中的其他队列管理器，这些队列管理器位于另一个队列管理器上的预订的路由上:

- 这依赖于每个队列管理器通知当前具有一个或多个预订的所有主题的每个直接关系 (在此队列管理器上或在其另一个关系上)。 这将导致层次结构中的所有队列管理器都知道要预订的所有主题。
- 可以将此行为更改为始终向层次结构中的所有队列管理器发送发布，而不考虑是否存在任何预订。 这样就不需要在层次结构中传播预订信息，但可以增加发布流量。

创建集群时，需要注意不要创建导致消息在网络中永久循环的循环。 不能在层次结构中创建此类循环。

每个队列管理器都必须具有唯一的队列管理器名称。

发布/预订消息流可以跨多个发布/预订集群。 为此，请将每个集群中的一个队列管理器添加到发布/预订层次结构中。

有关更详细的探索，请参阅 在发布/预订层次结构中路由。

发布/预订网络中的代理预订

代理预订是一个队列管理器为另一个队列管理器上发布的主题所作的预订。 代理预订会针对预订所预订的每一个主题字符串在队列管理器间流动。 您无需明确创建代理预订，队列管理器会替您执行此操作。

您可以将队列管理器一起连接到发布/预订集群或发布/预订层次结构中。 代理预订在已连接的队列管理器之间流动。 代理预订导致连接到一个队列管理器的发布程序创建的主题的发布由连接到其他队列管理器的该主题的订户接收。 请参阅第 190 页的『分布式发布/预订网络』。

在具有针对各个主题字符串的数千个预订的发布/预订拓扑中，或者在这些预订的存在可能快速变化的情况下，必须考虑代理预订传播的开销。 除了本主题的其余部分中描述的自动聚集外，您还可以进行手动配置更改，以进一步限制已连接队列管理器之间的代理预订和发布的流，并减少等待代理预订传播到所有已连接队列管理器的等待时间。 请参阅 发布/预订网络中的预订性能。

代理预订不包含本地预订所使用的任何选择器，可以简化包含通配符的预订主题字符串。 这可能会导致发布与实际预订不匹配的代理预订相匹配，从而在队列管理器之间产生额外的发布流。 托管预订的队列管理器会过滤掉此类差异，以便不会将其他发布返回给预订。

代理预订聚集

代理预订是使用重复消除系统聚集的。 对于特定的已解析主题字符串，将在第一个本地预订或接收到的代理预订上发送代理预订。 对同一主题字符串的后续预订将使用此现有代理预订。

在取消最后一个本地预订或接收到的代理预订之后，将取消代理预订。

发布聚集

当队列管理器上存在对同一主题字符串的多个预订时，仅从发布/预订拓扑中的其他队列管理器发送与该主题字符串匹配的每个发布的单个副本。 当消息到达时，本地队列管理器会将消息副本传递到每个匹配的预订。

当代理预订包含通配符时，可能有多个代理预订与单个发布的主题字符串匹配。 如果在与单个连接的队列管理器创建的两个或多个代理预订匹配的队列管理器上发布消息，那么仅会将该发布的一个副本转发到远程队列管理器以满足多个代理预订。

相关信息

分布式发布/预订网络中进行回路检测

代理预订中的通配符

预订可以使用主题字符串中的通配符来匹配发布中的多个主题字符串。

预订可以使用两个通配符模式: 基于主题的 和 基于字符的。 请参阅第 174 页的『通配方案』。

在 IBM WebSphere MQ 7.0 和更高版本中，通配符预订的所有代理预订都将转换为使用基于主题的通配符。 如果找到基于字符的通配符，那么会将其替换为 # 字符，返回到最近的 /。 例如， /aaa/bbb/c*d 将转换为 /aaa/bbb/#。 此转换导致远程队列管理器发送的发布数略多于显式预订的发布数。 当本地队列管理器将发布内容传递给其本地订户时，会将其他发布内容过滤掉。

使用 通配符 属性控制通配符使用

使用 MQSC **Topic** 通配符 属性或等效 PCF 主题 `WildcardOperation` 属性来控制向使用通配符主题字符串名称的订户应用程序传递发布内容。 通配符 属性可以具有以下两个可能的值之一:

WILDCARD
与此主题有关的通配符预订的行为。

PASSTHRU
对没有此主题对象中主题字符串具体的通配主题进行的预订将接收对此主题以及比此主题更具体的主题字符串进行的发布。

BLOCK
对没有此主题对象中主题字符串具体的通配主题进行的预订不会接收对此主题或比此主题更具体的主题字符串进行的发布。

在定义预订时将使用此属性的值。 如果改变此属性，那么现有预订涵盖的主题集不会因为此修改而受到影响。 如果在创建或删除主题对象时拓扑发生更改，此场景也适用；将使用修改后的拓扑来创建与修改 WILDCARD 属性后创建的预订匹配的主题集。 如果要针对现有预订强制重新评估匹配的主题集，那么必须重新启动队列管理器。

在示例 第 185 页的『示例: 创建 Sport 发布/预订集群』中，您可以遵循步骤来创建 第 182 页的图 23 中显示的主题树结构。

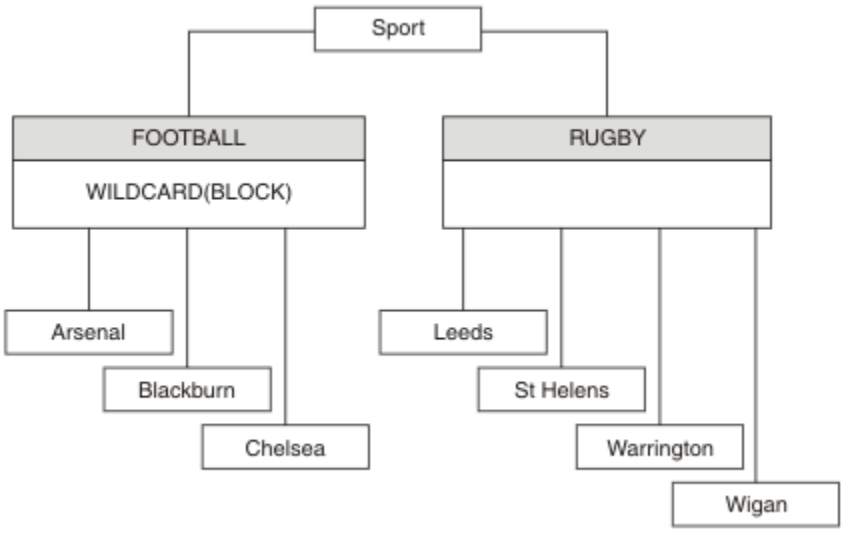


图 *37:* 使用 通配符 属性 *BLOCK* 的主题树

使用通配符主题字符串 *#* 的订户将接收到 Sport 主题和 Sport/Rugby 子树的所有发布。 订户不会接收到 Sport/Football 子树的发布，因为 Sport/Football 主题的 **通配符** 属性值为 BLOCK。

PASSTHRU 是缺省设置。 您可以将 **通配符** 属性值 PASSTHRU 设置为 Sport 树中的节点。 如果节点没有 **通配符** 属性值 BLOCK，那么设置 PASSTHRU 不会改变 Sports 树中节点的订户观察到的行为。

在此示例中，创建预订以查看通配符设置如何影响交付的发布; 请参阅 第 186 页的图 27。 在 第 187 页的图 30 中运行发布命令以创建一些发布。

```
pub QMA
```

图 *38:* 发布到 *QMA*

第 183 页的表 8 显示了结果。 请注意，如何设置 **通配符** 属性值 BLOCK，以防止具有通配符的预订将发布内容接收到通配符作用域内的主题。

表 *11: QMA* 上收到的出版物

预订	**主题字符串**	**收到的出版物**	**注意**
SPORTS	Sports/#	Sports Sports/Rugby Sports/Rugby/Leeds	WILDCARD(BLOCK) 在 Sports/Football 上阻止了对 Football 子树的所有发布
SARSENAL	Sports/#/Arsenal	-	Sports/Football 上的 WILDCARD(BLOCK) 会阻止 Arsenal 上的通配符预订
SLEEDS	Sports/#/Leeds	Sports/Rugby/Leeds	Sports/Rugby 上的缺省 WILDCARD 不会阻止 Leeds 上的通配符预订。

注:

假设预订具有通配符，该通配符与具有 **通配符** 属性值 BLOCK 的主题对象相匹配。 如果预订还具有匹配通配符右侧的主题字符串，那么预订将从不接收发布内容。 未被阻止的发布集合是作为被阻止通配符的父代的主题的发布。 通配符将阻止发布到作为具有 BLOCK 属性值的主题的子代的主题。 因此，包含通配符右侧主题的预订主题字符串永远不会收到任何要匹配的发布内容。

将 WILDCARD 属性值设置为 BLOCK 并不意味着您无法使用包含通配符的主题字符串进行预订。 这样的订阅是正常的。 预订具有一个显式主题，该主题与具有 **通配符** 属性值 BLOCK 的主题对象匹配。 它将通配符用于作为具有 **通配符** 属性值 BLOCK 的主题的父代或子代的主题。 在 第 182 页的图 23 中的示例中，预订 (例如 Sports/Football/#) 可以接收发布。

通配符和集群主题

集群主题定义将传播到集群中的每个队列管理器。 在集群中的一个队列管理器上预订集群主题会导致队列管理器创建代理预订。 将在集群中的每个其他队列管理器上创建代理预订。 使用包含通配符的主题字符串 (与集群主题结合使用) 的预订可能难以预测行为。 此行为在以下示例中进行了说明。

在为示例 第 185 页的『示例: 创建 Sport 发布/预订集群』设置的集群中， QMB 具有与 QMA 相同的预订集，但 QMB 在发布者发布到 QMA 之后未收到任何发布，请参阅 第 182 页的图 24。 虽然 Sports/Football 和 Sports/Rugby 主题是集群主题，但 fullsubs.tst 中定义的预订不会引用集群主题。 不会将代理预订从 QMB 传播到 QMA。 如果没有代理预订，那么不会将 QMA 的任何发布转发到 QMB。

某些预订 (例如 Sports/#/Leeds) 可能似乎引用了集群主题，在本例中为 Sports/Rugby。 Sports/#/Leeds 预订实际解析为主题对象 SYSTEM.BASE.TOPIC。

用于解析预订 (例如， Sports/#/Leeds) 所引用的主题对象的规则如下所示。 将主题字符串截断为第一个通配符。 通过主题字符串向左扫描以查找具有关联管理主题对象的第一个主题。 主题对象可以指定集群名称，也可以定义本地主题对象。 在示例 Sports/#/Leeds 中，截断后的主题字符串是 Sports，它没有主题对象，因此 Sports/#/Leeds 继承自 SYSTEM.BASE.TOPIC(这是本地主题对象)。

要查看预订集群主题如何更改通配符传播的工作方式，请运行批处理脚本 upsubs.bat。 该脚本将清除预订队列，并在 fullsubs.tst 中添加集群主题预订。 再次运行 puba.bat 以创建一批出版物; 请参阅 第 182 页的图 24。

第 184 页的表 9 显示了将两个新预订添加到发布发布的同一队列管理器的结果。 结果与预期相同，新预订各接收一个发布，其他预订接收的发布数量保持不变。 意外结果发生在其他集群队列管理器上; 请参阅 第 184 页的表 10。

表 *12: QMA* 上收到的出版物

预订	主题字符串	收到的出版物	注意
SPORTS	Sports/#	Sports Sports/Rugby Sports/Rugby/Leeds	WILDCARD(BLOCK) 在 Sports/Football 上阻止了对 Football 子树的所有发布
SARSENAL	Sports/#/Arsenal	-	Sports/Football 上的 WILDCARD(BLOCK) 会阻止 Arsenal 上的通配符预订
SLEEDS	Sports/#/Leeds	Sports/Rugby/Leeds	Sports/Rugby 上的缺省 WILDCARD 不会阻止 Leeds 上的通配符预订。
FARSENAL	Sports/Football/Arsenal	Sports/Football/Arsenal	Arsenal 接收发布，因为预订没有通配符。
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds 将在任何情况下接收发布。

第 184 页的表 10 显示了在 QMB 上添加两个新预订以及在 QMA 上发布的结果。 请注意，在没有这两个新预订的情况下， QMB 未收到任何发布。 正如预期的那样，这两个新预订将接收发布内容，因为 Sports/FootBall 和 Sports/Rugby 都是集群主题。 QMB 将 Sports/Football/Arsenal 和 Sports/Rugby/Leeds 的代理预订转发到 QMA，然后将发布发送到 QMB。

意外的结果是，先前未收到任何发布的两个预订 Sports/# 和 Sports/#/Leeds 现在接收发布。 原因是针对其他预订转发到 QMB 的 Sports/Football/Arsenal 和 Sports/Rugby/Leeds 发布现在可用于连接到 QMB 的任何订户。 因此，本地主题 Sports/# 和 Sports/#/Leeds 的预订将接收 Sports/Rugby/

Leeds 发布。 Sports/#/Arsenal 继续不接收发布内容，因为 Sports/Football 已将其 **通配符** 属性值设置为 BLOCK。

表 *13: QMB* 上收到的出版物			
预订	**主题字符串**	**收到的出版物**	**注意**
SPORTS	*Sports/#*	*Sports/Rugby/Leeds*	*WILDCARD(BLOCK)* 在 *Sports/Football* 上阻止了对 *Football* 子树的所有发布
SARSENAL	Sports/#/Arsenal	-	Sports/Football 上的 WILDCARD(BLOCK) 会阻止 Arsenal 上的通配符预订
SLEEDS	*Sports/#/Leeds*	*Sports/Rugby/Leeds*	*Sports/Rugby* 上的缺省 *WILDCARD* 不会阻止 *Leeds* 上的通配符预订。
FARSENAL	Sports/Football/Arsenal	Sports/Football/Arsenal	Arsenal 接收发布，因为预订没有通配符。
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds 将在任何情况下接收发布。

在大多数应用程序中，不希望一个预订影响另一个预订的行为。 **通配符** 属性与值 BLOCK 的一个重要用途是使包含通配符的同一主题字符串的预订行为一致。 无论预订是在与发布程序相同的队列管理器上，还是在不同的队列管理器上，预订的结果都是相同的。

通配符和流

对于写入发布/预订 API 的新应用程序，结果是对 * 的预订不会收到任何发布。 要接收所有体育出版物，必须预订 Sports/*或 Sports/#以及类似的 Business 出版物。

将发布/预订代理迁移到 IBM MQ 版本 7 和更高版本时，现有排队的发布/预订应用程序的行为不会更改。 **Publish**， **Register Publisher** 或 **Subscriber** 命令中的 **StreamName** 属性将映射到流已迁移到的主题的名称。

通配符和预订点

对于写入发布/预订 API 的新应用程序，迁移的效果是，对 * 的预订不会收到任何发布。 要接收所有体育出版物，必须预订 Sports/*或 Sports/#以及类似的 Business 出版物。

将发布/预订代理迁移到 IBM MQ 版本 7 和更高版本时，现有排队的发布/预订应用程序的行为不会更改。 **Publish**， **Register Publisher** 或 **Subscriber** 命令中的 **SubPoint** 属性将映射到预订已迁移到的主题的名称。

示例: 创建 Sport 发布/预订集群

后续步骤将创建一个具有四个队列管理器的集群 CL1: 两个完整存储库 CL1A 和 CL1B 以及两个部分存储库 QMA 和 QMB。 完整存储库仅用于保存集群定义。 QMA 被指定为集群主题主机。 持久预订同时在 QMA 和 QMB 上定义。

注: 此示例针对 Windows 进行编码。 您必须重新编码 Create qmgrs.bat 和 create pub.bat 以在其他平台上配置和测试示例。

1. 创建脚本文件。
   a. 创建 topics.tst
   b. 创建 wildsubs.tst
   c. 创建 fullsubs.tst

d. 创建 qmgrs.bat

e. 创建 pub.bat

2. 运行 Create qmgrs.bat 以创建配置。

```
qmgrs
```

在 第 182 页的图 23 中创建主题。 图 5 中的脚本将创建集群主题 Sports/Football 和 Sports/Rugby。

注: REPLACE 选项不会替换主题的 TOPICSTR 属性。 TOPICSTR 是在示例中进行有效更改以测试不同主题树的属性。 要更改主题，请先删除该主题。

```
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')
```

图 *39:* 删除并创建主题*: topics.tst*

注: 删除主题，因为 REPLACE 不会替换主题字符串。

使用通配符创建预订。 通配符将主题与 第 182 页的图 23 中的主题对象相对应。 为每个预订创建一个队列。 运行或重新运行脚本时，将清除队列并删除预订。

注: REPLACE 选项不会替换预订的 TOPICOBJ 或 TOPICSTR 属性。 TOPICOBJ 或 TOPICSTR 是在用于测试不同预订的示例中进行有效更改的属性。 要对其进行更改，请先删除预订。

```
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)
```

图 *40:* 创建通配符预订*: wildsubs.tst*

创建引用集群主题对象的预订。

注:

将在 TOPICOBJ 引用的主题字符串与 TOPICSTR 定义的主题字符串之间自动插入定界符 /。

定义 DEFINE SUB(FARSENAL) TOPICSTR('Sports/Football/Arsenal') DEST(QFARSENAL) 将创建相同的预订。 TOPICOBJ 用作引用您已定义的主题字符串的快速方法。 创建预订时，不再引用主题对象。

```
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)
```

图 *41:* 删除并创建预订*: fullsubs.tst*

创建具有两个存储库的集群。 创建两个用于发布和预订的部分存储库。 重新运行脚本以删除所有内容，然后再次启动。 该脚本还会创建主题层次结构和初始通配符预订。

注:

在其他平台上，编写类似的脚本，或者输入所有命令。 通过使用脚本，可以快速删除所有内容，并使用相同的配置重新开始。

```
@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%)')
CLUSTER(CL1) REPLACE | runmqsc %1
goto:eof
```

图 *42:* 创建队列管理器*: qmgrs.bat*

通过将预订添加到集群主题来更新配置。

```
@echo off
for %%A in (QMA QMB) do runmqsc %%A < wildsubs.tst
for %%A in (QMA QMB) do runmqsc %%A < upsubs.tst
```

图 *43:* 更新预订*: upsubs.bat*

运行以队列管理器作为参数的 pub.bat，以发布包含发布主题字符串的消息。 Pub.bat 使用样本程序 **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
```

图 *44: 发布: pub.bat*

相关信息
通配符预订和保留发布

发布作用域

配置发布/预订集群或层次结构时，发布的作用域将进一步控制队列管理器是否将发布转发到远程队列管理器。 使用 **PUBSCOPE** 主题属性来管理发布的作用域。

如果未将发布转发至远程队列管理器，那么只有本地订户才会接收该发布。

使用发布/预订集群时，发布范围主要由主题树中某些点的集群主题对象定义控制。 必须设置发布作用域以允许发布流至集群中的其他队列管理器。 仅当需要对特定队列管理器上的特定主题进行细粒度控制时，才应限制集群主题的发布范围。

使用发布/预订层次结构时，发布的作用域主要由此属性与 预订作用域 属性组合控制。

PUBSCOPE 属性用于确定对特定主题进行的发布的作用域。 可以将属性设置为下列其中一个值:

QMGR
该出版物仅提供给本地订户。 这些出版物称为 本地出版物。 本地发布不会转发至远程队列管理器，因此连接至远程队列管理器的订户不会接收本地发布。

ALL
发布将传递到本地订户以及连接到发布/预订集群或层次结构中的远程队列管理器的订户。 这些出版物称为 全局出版物。

ASPARENT
使用主题树中父主题的 **PUBSCOPE** 设置。

发布者还可以使用 MQPMO_SCOPE_QMGR put 消息选项来指定发布是本地发布还是全局发布。 如果使用此选项，那么它将覆盖已使用 **PUBSCOPE** 主题属性设置的任何行为。

相关概念
第 180 页的『管理主题对象』
通过使用管理主题对象，可以将特定非缺省属性分配给主题。

相关信息
配置分布式发布/预订网络

预订作用域

预订的作用域控制一个队列管理器上的预订是接收在发布/预订集群或层次结构中的另一个队列管理器上发布的发布，还是仅接收来自本地发布程序的发布。

将预订作用域限制为队列管理器将阻止代理预订转发到发布/预订拓扑中的其他队列管理器。 这将减少队列管理器之间的发布/预订消息传递流量。

使用发布/预订集群时，预订的作用域主要由主题树中某些点的集群主题对象的定义控制。 必须设置预订作用域以允许代理预订流向集群中的其他队列管理器。 仅当需要对特定队列管理器上的特定主题进行细粒度控制时，才应限制集群主题的预订范围。

使用发布/预订层次结构时，预订的作用域主要由此属性与 发布作用域 属性组合控制。

SUBSCOPE 主题属性用于确定对特定主题进行的预订的作用域。 可以将属性设置为下列其中一个值:

QMGR
预订仅接收本地发布，并且不会将代理预订传播到远程队列管理器。

ALL
代理预订传播到发布/预订集群或层次结构中的远程队列管理器，并且订户接收本地和远程发布。

ASPARENT
使用主题树中父主题的 **SUBSCOPE** 设置。

当主题的预订作用域设置为 ALL(直接设置或通过 ASPARENT 解析) 时，针对该主题的各个预订可以通过在创建预订时指定 MQSO_SCOPE_QMGR 来将其作用域限制为 QMGR 。 对作用域为 QMGR 的主题的预订无法将作用域扩大到 ALL。

相关概念
第 180 页的『管理主题对象』
通过使用管理主题对象，可以将特定非缺省属性分配给主题。

相关信息
配置分布式发布/预订网络

主题空间

主题空间是您可以预订和发布的主题集。 分布式发布/预订拓扑中的队列管理器具有一个主题空间，该主题空间可能包含已在该拓扑中的已连接队列管理器上预订和发布的主题。

注: 有关队列管理器中的主题 (例如管理主题对象，主题字符串和主题树) 的概述，请参阅第 172 页的『主题』。 除非另有指定，否则当前文章中对 主题 的进一步引用将引用 主题字符串 。

主题最初是通过以下任一方式创建的:

- 以管理方式定义主题对象或持久预订时。
- 当应用程序动态地创建新主题的发布或预订时。

主题通过代理预订和通过创建管理集群主题对象传播到其他队列管理器。 代理预订导致发布从发布程序所连接的队列管理器转发到订户的队列管理器。

代理预订将在通过队列管理器层次结构中的父子关系连接在一起的所有队列管理器之间传播。 结果是，您可以在一个队列管理器上预订层次结构中任何其他队列管理器上定义的主题。 只要队列管理器之间存在已连接的路径，那么队列管理器的连接方式就无关紧要。

还会为发布/预订集群中的集群主题的预订传播代理预订。 集群主题是附加到具有 **CLUSTER** 属性或从其父代继承该属性的主题对象的主题。 不是集群主题的主题称为本地主题，不会复制到集群。 不会将任何代理预订从预订传播到集群到本地主题。

总之，在两种情况下会为订户创建代理预订。

1. 队列管理器是层次结构的成员，代理预订将转发给队列管理器的父代和子代。
2. 队列管理器是集群的成员，预订主题字符串解析为与集群主题对象关联的主题。 当主题是 直接路由 集群主题时，会将代理预订转发到集群的所有成员。 当主题是 主题主机路由 集群主题时，仅会将代理预订转发到已定义集群主题对象的集群中的队列管理器。 有关更多信息，请参阅第 193 页的『发布/预订集群』。

如果队列管理器是集群和层次结构的成员，那么代理预订由两种机制传播，而不会将重复的发布传递给订户。

以下列表中描述了三个发布/预订拓扑的主题空间:

- 第 204 页的『案例 1。 发布/预订集群』.
- 第 204 页的『案例 2。 版本 7 或更高版本中的发布/预订层次结构』.
- 第 204 页的『案例 3。 版本 6 中的发布/预订层次结构和流』.

在单独的主题中，以下配置任务描述了如何组合主题空间。

- 在发布/预订集群中创建单个主题空间。
- 将 V 7 或更高版本的队列管理器添加到现有 版本 6 主题空间。
- 组合多个集群的主题空间。
- 组合和隔离多个集群中的主题空间。

- 发布和预订多个集群中的主题空间。

案例 1。 发布/预订集群

在此示例中，假定队列管理器 未 连接到发布/预订层次结构。

如果队列管理器是发布/预订集群的成员，那么其主题空间由本地主题和集群主题组成。 本地主题与没有 **CLUSTER** 属性的主题对象相关联。 如果队列管理器具有本地主题对象定义，那么其主题空间与集群中另一个也具有其自己的本地定义主题对象的队列管理器不同。

在发布/预订集群中，无法预订另一个队列管理器上定义的主题，除非您预订的主题解析为集群主题对象。

在多个队列管理器上需要集群主题对象的相同指定定义时 (例如，使用 主题主机路由时) ，所有定义在必要时匹配很重要。 有关更多信息，请参阅 在发布/预订集群中创建单个主题空间。

主题对象的本地定义 (无论该定义是针对集群主题还是本地主题) 优先于集群中其他位置定义的相同主题对象。 使用本地定义的主题，即使在其他位置定义的对象是最新的。

集群主题对象与集群中所有位置的相同主题字符串相关联很重要。 不能修改与主题对象关联的主题字符串。 要将同一主题对象与另一主题字符串相关联，必须删除该主题对象，并使用新的主题字符串重新创建该主题对象。 如果主题是集群的，那么效果是删除存储在集群的其他成员上的主题对象的副本，然后在集群中的所有位置创建新主题对象的副本。 主题对象的副本都引用同一个主题字符串。

可能会意外地在集群中的不同队列管理器上创建两个具有不同主题字符串的相同命名主题对象的定义。 这可能会导致混淆行为，因为具有不同主题字符串的同一主题对象的多个定义可能会产生不同的结果，具体取决于引用主题的方式和位置。 请参阅 同名的多个集群主题定义 ，以获取有关此重要点的更多信息。

案例 2。 版本 7 或更高版本中的发布/预订层次结构

在此示例中，假定队列管理器 不是 发布/预订集群的成员。

在 V 7 或更高版本中，如果队列管理器是发布/预订层次结构的成员，那么其主题空间由本地定义的所有主题以及在已连接的队列管理器上定义的所有主题组成。 层次结构中所有队列管理器的主题空间都相同。 没有将主题划分为局部主题和全局主题。

将 **PUBSCOPE** 和 **SUBSCOPE** 选项中的任一选项设置为 QMGR，以防止主题上的发布从发布者流向连接到层次结构中不同队列管理器的订户。

假设您在队列管理器 QMA 上使用主题字符串 USA/Alabama 定义主题对象 Alabama 。 结果如下:

1. QMA 上的主题空间现在包含主题对象 Alabama 和主题字符串 USA/Alabama。
2. 应用程序或管理员可以使用主题对象名称 Alabama 在 QMA 上创建预订。
3. 应用程序可以在层次结构中的任何队列管理器上创建对任何主题 (包括 USA/Alabama) 的预订。 如果未在本地定义 QMA ，那么主题 USA/Alabama 将解析为主题对象 SYSTEM.BASE.TOPIC。

案例 3。 版本 6 中的发布/预订层次结构和流

在 V 7 之前，主题空间分为不同的流，其中包括所有队列管理器上存在的缺省流。 发布不能在不同流之间流动。 如果使用指定的流，那么不同队列管理器上的主题空间可能不同。 主题分为缺省流中的主题和不同命名流中的主题。

注: 每个指定的流构成一个单独的主题空间。 要构成已连接的拓扑，每个指定的流都必须存在于已连接的队列管理器上。 假设流 X 是在 QMA 和 QMC 上定义的，而不是在 QMB 上定义的。 如果 QMA 是 QMB 的父代，而 QMB 是 QMC 的父代，那么流 X 中的任何主题都不能在 QMA 和 QMC 之间流动。

将 **PUBSCOPE** 和 **SUBSCOPE** 选项都设置为 QMGR 或 ALL 要求主题的发布者和订户仅将发布内容交换为本地使用，或仅将发布内容交换为全局使用。

从 V 7 开始，流不可使用发布/预订 API。 如果在 V 7 队列管理器上使用排队的发布/预订，那么会将流映射到可模拟流效果的不同主题对象。 通过创建作为流中所有主题的根主题的主题对象来模拟流。 队列管理器在流与每个树的相应根主题之间映射发布和预订。

相关信息

配置分布式发布/预订网络

发布作用域
预订作用域

IBM MQ 多点广播

IBM MQ 多点广播提供具有低等待时间和高扇出的可靠多点广播消息传递。

多点广播是一种高效的发布/预订消息传递形式，因为它可以扩展为大量订户，而不会对性能产生不利影响。IBM MQ 使用应答、否定应答和序号来支持可靠的多点广播消息传递，以实现具有高扇出的低等待时间消息传递。

IBM MQ 多点广播的公平传递支持几乎同时的传递，从而确保不偏向任何接收方。 IBM MQ 多点广播使用网络来传递消息，因此无需发布/预订引擎对数据进行扇出。 将主题映射到组地址后，不需要队列管理器，因为发布程序和订户可以在对等方式下运行。 这就允许减少队列管理器服务器上的负载，而队列管理器服务器将不再是潜在的故障点。

初始多点广播概念

IBM MQ 通过使用 "通信信息" (COMMINFO) 对象，可以轻松地将多点广播集成到现有系统和应用程序中。两个 TOPIC 对象字段支持快速配置现有 TOPIC 对象以支持或忽略多点广播流量。

多点广播所需的对象

以下信息是 IBM MQ 多点广播所需的两个对象的简要概述:

***COMMINFO* 对象**
COMMINFO 对象包含与多点广播传输关联的属性。 有关 COMMINFO 对象参数的更多信息，请参阅 DEFINE COMMINFO。

必须设置的唯一 COMMINFO 字段是 COMMINFO 对象的名称。 然后，此名称用于向主题标识 COMMINFO 对象。 必须检查 COMMINFO 对象的 **GRPADDR** 字段，以确保该值是有效的多点广播组地址。

***TOPIC* 对象**
主题是发布/预订消息中发布的信息的主题，通过创建 TOPIC 对象来定义主题。 有关 TOPIC 对象参数的更多信息，请参阅 DEFINE TOPIC。

通过更改以下 TOPIC 对象参数的值，可以将现有主题用于多点广播: **COMMINFO** 和 **MCAST**。

- **COMMINFO** 此参数指定多点广播通信信息对象的名称。
- **MCAST** 此参数指定在主题树中的此位置是否允许多点广播。 缺省情况下， **MCAST** 设置为 ASPARENT 表示从父代继承主题的多点广播属性。 将 **MCAST** 设置为 ENABLED 允许此节点上的多点广播流量。

多点广播网络和主题

以下信息概述了对具有不同类型的预订和主题定义的预订执行的操作。 这些示例都假定 TOPIC 对象 **COMMINFO** 参数设置为有效 COMMINFO 对象的名称:

主题集已启用多点广播
如果主题字符串 **MCAST** 参数设置为 ENABLED，那么允许来自支持多点广播的客户机的预订并进行多点广播预订，除非:

- 它是来自支持多点广播的客户机的持久预订。
- 它是来自支持多点广播的客户机的非受管预订。
- 它是来自不支持多点广播的客户机的预订。

在这些情况下，将进行非多点广播预订，并将 预订降级为正常发布/预订。

已禁用设置为多点广播的主题
如果主题字符串 **MCAST** 参数设置为 DISABLED，那么将始终进行非多点广播预订，并且 预订降级为正常发布/预订。

主题设置为仅多点广播
如果主题字符串 **MCAST** 参数设置为 ONLY，那么将允许来自支持多点广播的客户机的预订，并且将进行多点广播预订，除非:

- 这是持久预订: 已拒绝持久预订，原因码为 2436 (0984) (RC2436): MQRC_DURABILITY_NOT_ALLOWED
- 它是非受管预订: 非受管预订被拒绝，原因码为 2046 (07FE) (RC2046): MQRC_OPTIONS_ERROR
- 它是来自不支持多点广播的客户机的预订: 这些预订被拒绝，原因码为 2560 (0A00) (RC2560): MQRC_MULTICAST_ONLY
- 这是来自本地绑定应用程序的预订: 将拒绝这些预订，原因码为 2560 (0A00) (RC2560): MQRC_MULTICAST_ONLY

IBM MQ Telemetry

IBM MQ Telemetry 包含作为队列管理器一部分的遥测 (MQXR) 服务，可自行编写或免费下载的遥测客户机以及命令行和资源管理器管理界面。 遥测指收集远程设备数据并管理各种远程设备。 通过 IBM MQ Telemetry，您可以将数据收集和设备控制与 Web 应用程序集成。

MQTT 支持先前随 WebSphere Message Broker 一起提供 (现在称为 IBM Integration Bus) 或 IBM WebSphere MQ 7.0.1，其中 IBM MQ Telemetry 是单独的功能部件。

IBM MQ Telemetry 是 IBM WebSphere MQ 7.1 和更高版本的组件。 这些版本的升级基本上是卸载 IBM MQ Telemetry 版本 7.0.1 并安装更高版本的 IBM MQ。

但是，从 IBM MQ 8.0 开始，不再作为产品的一部分提供 Client Software Development Kit。 可以继续从 Eclipse Paho 和 MQTT.org 免费获取类似的样本应用程序。 请参阅 IBM MQ Telemetry Transport 样本程序。

由于 IBM MQ Telemetry 是 IBM WebSphere MQ 7.1 和更高版本的组件，因此 IBM MQ Telemetry 可以随主产品一起安装，也可以在安装主产品之后安装。 有关迁移信息，请参阅 从 版本 7.0.1 到版本 8.0 迁移 IBM MQ Telemetry 。

IBM MQ Telemetry 中包含以下组件:

遥测通道
使用遥测通道来管理 MQTT 客户机与 IBM MQ 的连接。 遥测通道使用新的 IBM MQ 对象 (例如 SYSTEM.MQTT.TRANSMIT.QUEUE) 与 IBM MQ 进行交互。

遥测 (MQXR) 服务
MQTT 客户机使用 SYSTEM.MQXR.SERVICE 遥测服务来连接到遥测通道。

MQ Explorer 支持 IBM MQ Telemetry
可以使用 MQ Explorer 来管理 IBM MQ Telemetry 。

Documentation
IBM MQ Telemetry 文档包含在 版本 7.1 的标准 IBM MQ 产品文档中。 Java 和 C 客户机的 SDK 文档在产品文档中以 Javadoc 和 HTML 形式提供。

遥测概念

您从周围的环境收集信息以决定要执行的操作。 作为消费者，你先检查你在店里有什么，然后再决定买什么食物。 您想知道如果现在离开，在预订连接之前，一段旅程将需要多长时间。 你检查一下你的症状，然后再决定是否去看医生。 在决定是否等待之前，请检查公交车何时到达。 这些决策的信息直接来自计量表和设备，来自纸上的文字或屏幕，以及来自您的信息。 在任何时候，您都需要收集信息，汇集信息，对其进行分析，并对其采取行动。

如果信息来源广泛分散或无法获取，收集最准确的信息就会变得困难和昂贵。 如果您要进行的更改很多，或者很难进行更改，那么这些更改不会进行，或者在效果较差时进行。

如果通过将具有数字技术的设备连接到互联网来大大降低从广泛分散的设备收集信息和控制这些设备的成本，将会怎样? 可以利用互联网和企业的资源对信息进行分析。 您有更多机会做出明智的决策并采取行动。

技术趋势以及环境和经济压力正在推动这些变化发生:

1. 由于标准化和与低成本数字处理器的连接，连接和控制传感器和执行器的成本正在降低。

2. 互联网和互联网技术越来越多地用于连接设备。 在一些国家，在与互联网应用程序的连接数量上，移动电话超过了个人计算机。 其他设备肯定在关注。
3. 互联网和互联网技术使应用程序更容易获取数据。 轻松访问数据正在推动使用数据分析，将来自传感器的数据转化为更多解决方案中有用的信息。
4. 智能使用资源往往是减少碳排放和成本的更快和更便宜的方法。 替代办法: 寻找新的资源，或开发新的技术来利用现有资源，可能是长期的解决办法。 从短期来看，开发新技术或寻找新资源，通常比改进现有解决方案风险更高，速度更慢，成本更高。

示例

一个示例显示了这些趋势如何创造与环境智能交互的新机会。

The International Convention for the Safety of Life at Sea (SOLAS) requires Automatic Identification System (AIS) to be deployed on many ships. 300 吨以上的商船和客船都需要。 AIS 主要是沿海航运的避撞系统。 它被海洋当局用来监测和控制沿海水域。

世界各地的爱好者正在部署低成本的 AIS 跟踪站，并将沿海航运信息放到互联网上。 其他爱好者正在编写将来自 AIS 的信息与来自互联网的其他信息相结合的应用程序。 结果将放在 Web 站点上，并使用 Twitter 和 SMS 进行发布。

在一个应用程序中，来自南安普敦附近 AIS 站的信息与船舶所有权和地理信息相结合。 该应用程序将有关轮渡到达和离开的实时信息提供给 Twitter。 使用南安普敦和怀特岛之间的渡轮的普通通勤者使用推特或短信订阅新闻订阅。 如果饲料显示他们的轮渡运行较晚，通勤者可能会延迟他们的出发，并在其停靠时间晚于其预定到达时间时捕获轮渡。

有关更多示例，请参阅第 209 页的『遥测用例』。

相关信息

安装 IBM MQ Telemetry
管理 IBM MQ Telemetry
从 版本 7.0.1 到版本 7.5 迁移 IBM MQ Telemetry
为 IBM MQ Telemetry 开发应用程序
IBM MQ Telemetry 参考
IBM MQ Telemetry 故障诊断

IBM MQ Telemetry 简介

人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

什么是 IBM MQ Telemetry?

- 它是 IBM MQ 的一个功能部件，用于将 IBM MQ 提供的通用消息传递主干扩展至各种远程传感器，执行器和遥测设备。 IBM MQ Telemetry 扩展 IBM MQ ，使其能够将智能企业应用程序，服务和决策制定者与受检测设备网络互连。
- IBM MQ Telemetry 的核心部件为:

  IBM MQ Telemetry (MQXR) 服务。
  此服务在 IBM MQ 服务器中运行，并使用 IBM MQ Telemetry Transport (MQTT) 协议与遥测设备进行通信。

  您编写的 MQTT 应用程序。
  这些应用程序控制遥测设备与 IBM MQ 队列管理器之间传递的信息以及为响应该信息而执行的任何操作。 要帮助创建这些应用程序，请使用 MQTT 客户机库。

  第 249 页的『设备的 MQTT 守护程序』。
  守护程序是高级遥测客户机，充当网络集中器以将更多 MQTT 客户机连接到单个队列管理器。 守护程序还可以为在短网络中断期间缺乏缓冲消息的方法的小型设备提供存储和转发设施。

它具有什么功能?

- MQTT 是一种开放式消息传递传输，允许为各种设备创建 MQTT 实现。
- MQTT 客户机可以在资源有限的小型占用量设备上运行。
- MQTT 在带宽较低，发送数据的成本较高或可能很脆弱的网络上高效工作。
- 保证消息传递并使其与应用程序分离。
- 应用程序员不需要具备通信编程知识。
- 可以与其他消息传递应用程序交换消息。 这些应用程序可以是另一个遥测应用程序，也可以是 MQI，JMS 或企业消息传递应用程序。

如何使用?

- 下载免费的 IBM Messaging Telemetry Client SupportPac，其中包含 MQTT 客户机库和设备的 MQTT 守护程序。
- 使用 MQ Explorer 及其关联工具来管理 IBM MQ 的遥测功能。
- 使用客户机库可帮助您创建连接到队列管理器并使用发布/预订消息传递的 MQTT 应用程序。
- 将应用程序和客户机库分发到要运行应用程序的设备。

工作原理

- MQTT 是发布预订协议。 MQTT 客户机应用程序可以将消息发布到 MQTT 服务器，或者预订由连接到 MQTT 服务器的应用程序发送的消息。
- MQTT 客户机应用程序使用实现 MQTT 消息传输的客户机库。
- 基本 MQTT 客户机应用程序类似于标准 MQ 客户机，但可以在更广泛的各种平台和网络上运行。
- MQ Telemetry (MQXR) 服务将 IBM MQ 队列管理器转换为 MQTT 服务器。
- 当 IBM MQ 队列管理器充当 MQTT 服务器时，连接到队列管理器的其他应用程序可以从 MQTT 客户机预订和接收消息。
- 队列管理器充当路由器将消息从发布应用程序分发到预订应用程序。
- 可以在不同类型的客户机应用程序之间分发消息。 例如，在 Telemetry 客户机与 JMS 客户机之间。

注: IBM MQ Telemetry 替换在 WebSphere Message Broker V 7 中撤销的 SCADA 节点 (现在称为 IBM Integration Bus) 并在 Windows， Linux 和 AIX 上运行。

相关概念

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

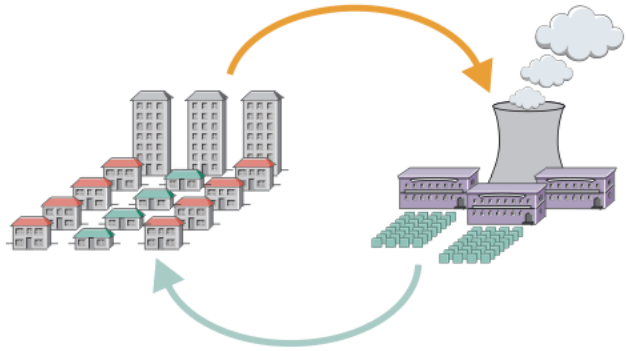
遥测用例

遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

IBM MQ Telemetry 使用 MQTT protocol 连接小型设备，并使用 IBM MQ 将这些设备连接到其他应用程序。 IBM MQ Telemetry 弥合了设备与互联网之间的差距，使其更容易构建 "智能解决方案"。 智能解决方案为监控和控制设备的应用程序解锁了互联网上和企业应用程序中可用的丰富信息。

下图演示了 IBM MQ Telemetry 的一些典型用法:

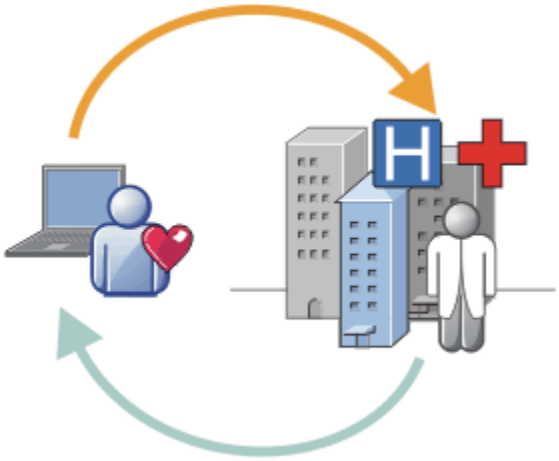
遥测: 智能电力



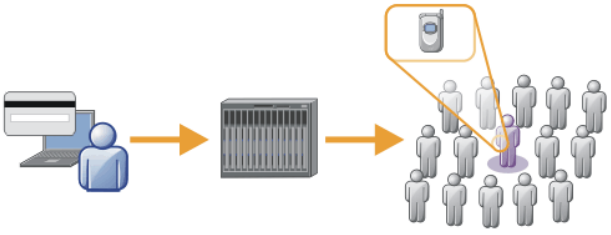
- 包含发送到服务提供者的能源使用数据的 MQTT 消息。
- IBM MQ Telemetry 根据对能源使用情况数据的分析发送 CONTROL COMMANDS。
- 有关更多信息，请参阅以下用例: 第 213 页的『遥测用例: 家庭能源监测与控制』

遥测: 智慧健康服务

- IBM MQ Telemetry 将健康数据发送到您的医院和医生。
- 可以根据运行状况数据分析来发送 MQTT 消息警报或反馈。
- 有关更多信息，请参阅以下用例: 第 212 页的『遥测用例: 家庭患者监控』



遥测: Crowd 中的一个



- 将简单的卡交易发送到银行的服务器。
- IBM MQ Telemetry 识别千人中的一人，提醒客户已使用他们的卡。
- IBM MQ Telemetry 可以使用最简单的信息输入，并找到该个人。

子主题中描述的用例来自实际示例。 它们说明了一些使用遥测的方法，以及遥测技术必须解决的一些常见问题。

相关概念

IBM MQ Telemetry 简介

人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备

MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息

执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成

MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

遥测用例: 家庭患者监控

在 IBM 与心脏患者护理系统上的医疗保健提供者之间的协作中，植入的心内膜除颤器与医院进行通信。 使用 RF 遥测将有关患者和植入设备的数据传输到患者家中的 MQTT 设备。

通常，传输会在夜间进行到位于床头的发射器。 发射器通过电话系统将数据安全地传输到医院，在医院中分析数据。

该系统减少了患者必须看医生的次数。 它检测患者或设备何时需要关注，在发生紧急情况时，它会向待命医生发出警报。

IBM 与医疗保健提供者之间的协作具有许多遥测用例所共有的特征:

Invisibility

该设备无需用户干预，只需提供电源，电话线，并在一天中的部分时间与该设备接近。 其操作可靠，使用简单。

为了消除患者设置设备的需求，设备供应商对设备进行预配置。 患者必须将其插入。 患者消除了配置，简化了设备的操作，减少了设备配置错误的机会。

MQTT 客户机作为设备的一部分嵌入。 设备开发者将 MQTT 客户机实现嵌入到设备中，开发者或供应商将 MQTT 客户机配置为预配置的一部分。

MQTT 客户机作为 Java SE jar 文件提供，开发者在其 Java 应用程序中包含此文件。 对于非 Java 环境 (例如，此环境) ，设备开发者可以使用已发布的 MQTT 格式和协议以其他语言实现客户机。 或者，开发者可以使用其中一个作为 Windows， Linux 和 ARM 平台的共享库交付的 C 客户机。

不均衡的连接

除颤器与医院之间的通信具有不均匀的网络特性。 用两个不同的网络来解决从病人那里收集数据，把数据送到医院的不同问题。 在专利与 MQTT 设备之间，使用短程低功耗 RF 网络。 发送器使用 VPN TCP/IP 连接通过低带宽电话线连接到医院。

通常无法找到将每个设备直接连接到 Internet Protocol 网络的方法。 使用两个网络 (通过中心连接) 是常见的解决方案。 MQTT 设备是一个简单的集线器，用于存储来自患者的信息，并将其转发到医院。

安全性

医生必须能够信任患者数据的真实性，患者希望其数据的隐私得到尊重。

在某些情况下，它足以使用 VPN 或 SSL 对连接进行加密。 在其他情况下，即使已存储数据，也应确保数据安全。

有时遥测设备不安全。 例如，它可能位于共享住宅中。 必须对设备的用户进行认证，以确保数据来自正确的患者。 可以使用 SSL 向服务器认证设备本身，并向设备认证服务器。

设备与队列管理器之间的遥测通道支持 JAAS 用于用户认证， SSL 用于通信加密和设备认证。 对发布的访问由 IBM MQ 中的对象权限管理器控制。

用于认证用户的标识可以映射到其他标识，例如公共患者标识。 公共标识可简化对 IBM MQ 中发布主题的授权配置。

连通性

MQTT 设备与医院之间的连接使用拨号，并使用低至 300 波特的带宽。

为了在 300 波特处有效运行，除了 TCP/IP 头外， MQTT protocol 仅向消息添加几个额外的字节。

MQTT protocol 提供单次传输 触发并忘记 消息传递，这使等待时间保持较低。 它还可以使用多个传输来保证 至少一次 和 正好一次 传递 (如果保证传递比响应时间更重要)。 为了保证传递，消息将存储在设备上，直到成功传递为止。 如果设备以无线方式连接，那么保证交付特别有用。

可扩展性

遥测设备通常大量部署，从数万到数百万。

将许多设备连接到系统会对解决方案产生很大的需求。 存在业务需求，例如设备及其软件的成本，以及管理许可证，设备和用户的管理需求。 技术需求包括网络和服务器上的负载。

与维护打开的连接相比，打开连接使用的服务器资源更多。 但是，在使用电话线之类的用例中，连接费用意味着连接处于打开状态的时间不会超过所需时间。 数据传输在很大程度上具有批处理性质。 可以安排通宵达旦的连接，避免在睡前出现突然的连接高峰。

在客户机上，客户机的可伸缩性由所需的最低客户机配置提供帮助。 MQTT 客户机嵌入在设备中。 不需要在向患者部署设备时内置配置或 MQTT 客户机许可证接受步骤。

在服务器上， IBM MQ Telemetry 的初始目标是每个队列管理器打开 50,000 个连接。

这些连接使用 MQ Explorer 进行管理。 MQ Explorer 会将要显示的连接过滤为可管理的数字。 通过适当选择的将标识分配给客户机的方案，您可以根据地理位置或按患者名称的字母顺序来过滤连接。

遥测用例: 家庭能源监测与控制

与传统计量表相比，智能计量表收集更多有关能耗的详细信息。

智能电表通常与本地遥测网络耦合在一起，以监控家庭中的各个设备。 还可以远程连接某些智能电表，以在远处进行监控。

远程连接可由个人，电源实用程序或中央控制点设置。 远程控制点可以读取电源使用情况并提供使用情况数据。 它可以提供数据来影响使用情况，例如持续定价和天气信息。 可限制负荷以提高整体发电效率。

智能计量表已开始广泛部署。 例如，英国政府正在就到 2020 年将智能电表部署到每个英国家庭进行协商。

家庭计量用例有以下几个共同特点:

Invisibility

除非用户希望通过使用计量表来参与节能，否则计量表不得要求用户干预。 它不得降低个别电器的能源供应的可靠性。

MQTT 客户机可以嵌入到随计量表一起部署的软件中，并且不需要单独的安装或配置。

不均衡的连接

设备与智能计量表之间的通信要求的连接标准与计量表与远程连接点之间的连接标准不同。

从智能计量表到设备的连接必须高度可用，并且符合家庭区域网络的网络标准。

远程网络可能使用各种物理连接。 其中一些，如细胞，其传播成本很高，可以是间歇性的。 MQTT v3 规范针对远程连接以及本地适配器与智能计量表之间的连接。

电源插座和应用程序与计量表之间的连接使用家庭区域网络，例如 Zigbee。 MQTT 用于传感器网络 (MQTT-S) ，旨在使用 Zigbee 和其他低带宽网络协议。 IBM MQ Telemetry 不直接支持 MQTT-S。 它需要网关将 MQTT-S 连接到 MQTT v3。

与家庭病人监测一样，家庭能源监测和控制的解决方案需要多个网络，使用智能仪表作为枢纽进行连接。

安全性

存在许多与智能计量表相关联的安全问题。 这些问题包括交易的不可抵赖性，启动的任何控制操作的授权以及功耗数据的隐私。

为确保隐私，可以使用 SSL 对 MQTT 在计量表和远程控制点之间传输的数据进行加密。 为确保控制操作的授权，可以使用 SSL 对计量表与远程控制点之间的 MQTT 连接进行相互认证。

连通性

远程网络的物理性质可能有很大差异。 它可能使用现有宽带连接，或者使用具有高呼叫成本和间歇性可用性的移动网络。 对于高成本，间歇性的连接， MQTT 是一种高效且可靠的协议; 请参阅 第 212 页的『遥测用例: 家庭患者监控』。

可扩展性

最终电力公司或中央控制点，计划部署数千万台智能电表。 最初，每个部署的计量表数在数万到数十万。 此数目与每个队列管理器 50,000 个开放式客户机连接的初始 MQTT 目标相当。

家庭能源监测和控制的架构的一个关键方面是使用智能仪表作为网络集中器。 每个设备适配器都是一个单独的传感器。 通过使用 MQTT 将它们连接到本地集线器，集线器可以将数据流集中到具有中央控制点的单个 TCP/IP 会话上，还可以在短时间内存储消息以克服会话中断。

在家庭能源用例中，远程连接必须保持打开状态，原因有二。 首先，由于打开连接相对于发送请求需要很长时间。 在较短的时间间隔内打开多个连接以发送 “装入限制” 请求的时间太长。 其次，要接收来自电力公司的负载限制请求，必须首先由客户打开连接。 使用 MQTT 时，连接始终由客户机启动，要接收来自电力公司的负载限制请求，连接必须保持打开状态。

如果打开连接的速率是临界值，或者服务器启动时间临界请求，那么解决方案通常是维护许多打开的连接。

遥测用例: 射频识别 (RFID)

RFID 是使用嵌入式 RFID 标签来无线识别和跟踪物体。 RFID 标签可以被读取到几米的范围内，并且脱离 RFID 阅读器的视线。 无源标签由 RFID 阅读器激活。 在没有外部激活的情况下传输活动标记。 活动标记必须具有电源。 无源标签可以包含电源以增加其范围。

RFID 在很多应用中都有使用，用例的类型也差别很大。 RFID 用例，以及家庭患者监测和家庭能源监测与控制用例，都有一些相似之处和不同之处。

Invisibility

在许多用例中， RFID 阅读器是大量部署的，必须在没有用户干预的情况下工作。 阅读器包含用于与中央控制点通信的嵌入式 MQTT 客户机。

例如，在配送仓库中，阅读器使用运动传感器来检测托盘。 它激活托盘上项目的 RFID 标签，并将数据和请求发送到中央应用程序。 数据用于更新库存位置。 这些请求可控制下一个托盘发生的情况，例如将其移动到特定托架。 航空公司和机场行李系统都在以这种方式使用 RFID。

在某些 RFID 用例中，阅读器具有标准计算环境，例如 Java ME。 在这些情况下， MQTT 客户机可能在制造后部署在不同的配置步骤中。

不均衡的连接

RFID 阅读器可能与包含 MQTT 客户机的本地控制设备分离，或者每个阅读器都可能嵌入 MQTT 客户机。 通常，地理或通信因素指示拓扑的选择。

安全性

隐私和真实性是 RFID 标签附件中的安全问题。 RFID 标签具有不可侵扰性，可被秘密监控，欺骗或篡改。

RFID 安全问题的解决方案增加了部署新的 RFID 解决方案的机会。 尽管安全漏洞在 RFID 标签和本地阅读器中，但使用中央信息处理建议了应对不同威胁的方法。 例如，可以通过将库存级别与交付和分派动态关联来检测标记篡改。

连通性

RFID 应用通常涉及批量存储和从 RFID 阅读器收集的信息以及即时查询。 在配送仓库用例中， RFID 阅读器一直连接。 读取标记时，会将其与有关读者的信息一起发布。 仓储应用程序将响应发布回阅读器。

在仓储应用程序中，网络通常是可靠的，即时请求可能会使用 触发和忘记 消息以实现低延迟性能。 批处理存储和转发数据可能使用 精确一次 消息传递，以最大限度降低与松散数据相关联的管理成本。

可扩展性

如果 RFID 应用程序需要立即响应 (按秒或秒的顺序) ，那么 RFID 阅读器必须保持连接。

遥测用例: 环境传感

环境传感利用遥测收集有关河流水位和质量，大气污染物以及其他环境数据的信息。

传感器经常位于偏远的地方，无法进行有线通信。 无线带宽昂贵，可靠性低。 通常，在一个小的地理区域内的一些环境传感器被连接到一个安全位置的本地监控设备。 本地连接可以是有线连接或无线连接。

Invisibility

与中央监控设备相比，这些传感器设备的可访问性可能较低，供电能力较低，并且部署的数量更多。 传感器有时是 "哑" 的，本地监控设备包括用于转换和存储传感器数据的适配器。 监视设备可能包含支持 Java SE 或 ME 的通用计算机。 配置 MQTT 客户机时，不太可能主要需要隐蔽性。

不均衡的连接

传感器的功能以及远程连接的成本和带宽通常会导致本地监控中心连接到中央服务器。

安全性

除非在军事或防御用例中使用该解决方案，否则安全不是主要要求。

连通性

许多用途不需要持续监控或即时提供数据。 需要立即转发异常数据 (例如，洪水级别警报)。 在本地监视器上聚集传感器数据以降低连接和通信成本，然后使用预定连接进行传输。 一旦在监视器上检测到异常数据，就会将其转发。

可扩展性

传感器集中在本地集线器周围，传感器数据聚集到根据调度传输的包中。 这两个因素都降低了中央服务器上通过使用直接连接的传感器施加的负载。

遥测用例: 移动应用程序

移动应用程序是在无线设备上运行的应用程序。 这些设备是通用应用程序平台或定制设备。

一般平台包括手机和个人数据助手等手持设备，笔记本电脑等便携式设备。 定制设备使用针对特定应用程序定制的特殊用途硬件。 用于记录 "signed-for" 包裹递送的设备是定制移动设备的示例。 定制移动设备上的应用程序通常是在通用软件平台上构建的。

Invisibility

定制移动应用程序的部署是受管的，并且可以包含 MQTT 客户机应用程序的配置。 配置 MQTT 客户机时，不太可能主要需要隐蔽性。

不均衡的连接

与先前用例的本地中心拓扑不同，移动式客户机远程连接。 客户机应用程序层直接连接到中央集线器上的应用程序。

安全性

在物理安全性很小的情况下，必须对移动设备和移动用户进行认证。 SSL 用于确认设备的身份， JAAS 用于认证用户。

连通性

如果移动应用程序依赖于无线覆盖，那么它必须能够脱机操作，并且能够高效地处理中断的连接。 在此环境中，目标是保持连接，但应用程序必须能够存储和转发消息。 通常，消息是订单或交货确认，并且具有重要的业务价值。 需要对它们进行可靠的存储和转发。

可扩展性

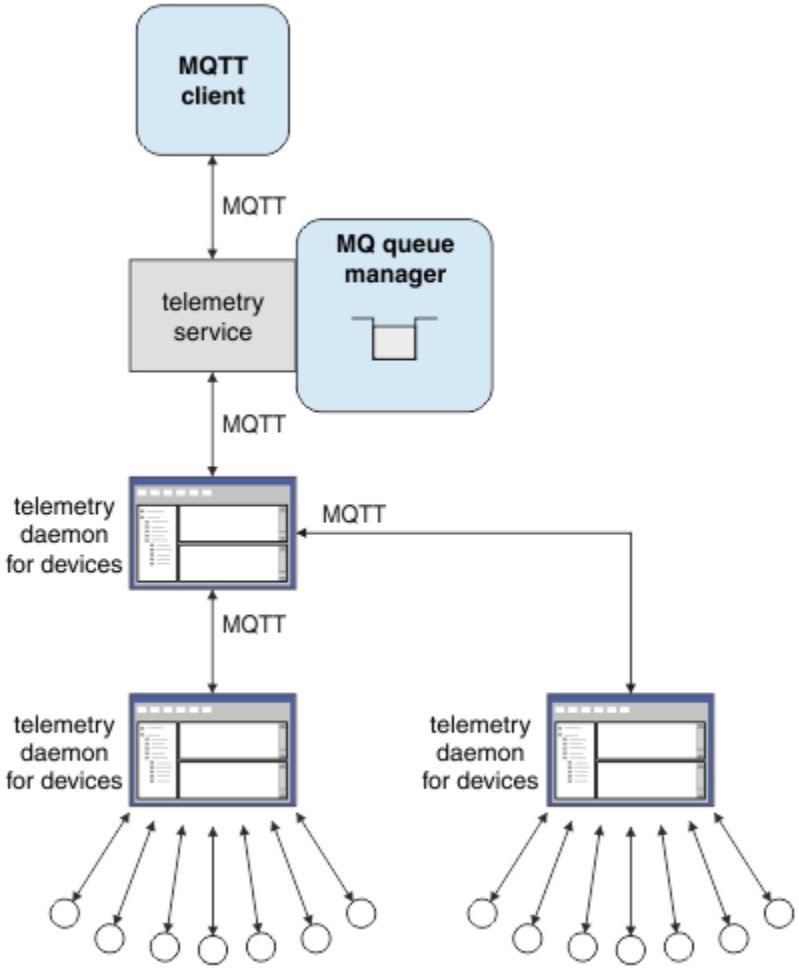
可扩展性不是一个主要问题。 在定制移动应用程序用例中，应用程序客户机的数量可能不会超过数千或数万个。

将遥测设备连接至队列管理器

遥测设备使用 MQTT v3 客户机连接到队列管理器。 MQTT v3 客户机使用 TCP/IP 连接到称为遥测 (MQXR) 服务的 TCP/IP 侦听器。

作为将遥测设备直接连接到 MQTT 服务器的替代方法，您可以将这些设备连接到设备的 MQTT 守护程序。 守护程序是高级 MQTT V3 客户机应用程序，也可以充当 MQTT 边缘服务器。 遥测设备和其他 MQTT 客户机 (包括设备的其他 MQTT 守护程序) 可以连接到设备的 MQTT 守护程序。 设备的守护程序对设备连接进行池化，并建立到遥测 (MQXR) 服务的单个连接。 您可以在层次结构中连接守护程序，从而将可以间接连接到 IBM MQ 的设备数增加许多数量级。

注: 设备的守护程序不再可用。 有关备用解决方案，请参阅 eclipse.org “Mosquitto” 项目 (https://eclipse.org/mosquitto)。



将遥测设备或设备守护程序连接到队列管理器时， MQTT 客户机将使用 `MqttClient.connect` 方法启动 TCP/IP 连接。 与 IBM MQ 客户机一样， MQTT 客户机必须连接到队列管理器以发送和接收消息。 使用随 IBM MQ Telemetry 一起安装的 TCP/IP 侦听器 (称为遥测 (MQXR) 服务) 在服务器上建立连接。 每个队列管理器最多运行一个遥测 (MQXR) 服务。

遥测 (MQXR) 服务使用每个客户机在 `MqttClient.connect` 方法中设置的远程套接字地址来分配与遥测通道的连接。 套接字地址是 TCP/IP 主机名和端口号的组合。 使用同一远程套接字地址的多个客户机通过遥测 (MQXR) 服务连接到同一遥测通道。

如果服务器上有多个队列管理器，请在队列管理器之间拆分遥测通道。 在队列管理器之间分配远程套接字地址。 使用唯一的远程套接字地址定义每个遥测通道。 两个遥测通道不得使用相同的套接字地址。

如果为多个队列管理器上的遥测通道配置了相同的远程套接字地址，那么要连接的第一个遥测通道将获胜。 在同一地址上连接的后续通道失败。

如果服务器上有多个网络适配器，请在遥测通道之间拆分远程套接字地址。 只要仅在一个遥测通道上配置任何特定套接字地址，那么套接字地址的分配完全是任意的。

配置 IBM MQ 以使用 IBM MQ Explorer 的 IBM MQ Telemetry 补充信息中提供的向导来连接 MQTT 客户机。 或者，遵循 在 Linux 和 AIX 上配置队列管理器以进行遥测 和 在 Windows 上配置队列管理器以进行遥测手动配置 中的指示信息。

相关概念

IBM MQ Telemetry 简介
人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

相关信息

MQXR 属性

遥测连接协议

IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

相关概念

IBM MQ Telemetry 简介
人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息

IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

遥测 (MQXR) 服务

遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

IBM MQ Telemetry (MQXR) 服务称为 SYSTEM.MQXR.SERVICE。

MQ Explorer 的 IBM MQ Telemetry 函数中提供的 **Telemetry sample configuration** 向导将创建遥测服务和样本遥测通道; 请参阅 使用 MQ Explorer 验证 IBM MQ Telemetry 的安装。 从命令行创建样本配置; 请参阅 使用命令行验证 IBM MQ Telemetry 的安装。

遥测 (MQXR) 服务会随队列管理器自动启动和停止。 使用 MQ Explorer 中的 services 文件夹来控制服务。 要查看服务，必须单击该图标以停止资源管理器从显示中过滤掉 SYSTEM 对象。

installMQXRService_unix.mqsc 显示了如何在 AIX 和 Linux 上手动创建服务的示例。
installMQXRService_win.mqsc 显示如何在 Windows 中手动创建服务。

相关概念

IBM MQ Telemetry 简介
人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性

保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

遥测通道

创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

使用 MQ Explorer 的 IBM MQ Telemetry 函数中提供的 **New Telemetry Channel** 向导创建遥测通道。 使用向导配置通道以接受来自特定 TCP/IP 端口上的 MQTT 客户机的连接。 从 版本 7.1 开始，您可以使用命令行程序 **runmqsc** 来配置 IBM MQ Telemetry 。

在不同端口上创建多个遥测通道，通过将客户机拆分为多个组，使大量客户机连接更易于管理。 每个遥测通道都有不同的名称。

您可以配置具有不同安全性属性的遥测通道，以创建不同类型的连接。 创建多个通道以接受不同 TCP/IP 地址上的客户机连接。 使用 SSL 对消息进行加密并对遥测通道和客户机进行认证; 请参阅 MQTT 客户机和遥测通道的 SSL 配置。 指定用户标识以简化对 IBM MQ 对象的授权访问。 指定 JAAS 配置以使用 JAAS 认证 MQTT 用户; 请参阅 MQTT 客户机标识，授权和认证。

相关概念

IBM MQ Telemetry 简介
人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client

MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

MQTT protocol

IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT protocol 已发布; 请参阅 IBM MQ Telemetry Transport 格式和协议。 协议版本 3 使用发布/预订，并支持三种服务质量: 触发和忘记， 至少一次和 正好一次。

协议头和字节数组消息有效内容的较小大小使消息保持较小。 这些头包含一个 2 字节的固定头，以及最多 12 个字节的其他变量头。 该协议使用 12 字节变量头来预订和连接，并且仅 2 字节变量头用于大多数发布。

通过三种服务质量，您可以在低延迟和可靠性之间进行权宜之法; 请参阅 MQTT 客户机提供的服务的程度。 "火与忘" 不使用持久设备存储，仅使用一个传输来发送或接收发布。 至少一次，并且 正好一次 需要设备上的持久存储器以保持协议状态并保存消息，直到确认为止。

相关概念

IBM MQ Telemetry 简介
人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时

当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

MQTT Client

MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

与 IBM MQ 客户机应用程序不同， MQTT 客户机应用程序不是 IBM MQ 应用程序。 它们不指定要连接到的队列管理器。 它们不限于使用特定的 IBM MQ 编程接口。 相反， MQTT 客户机实现 MQTT 3 协议。 您可以编写自己的客户机库，以便在您选择的编程语言和平台上与 MQTT protocol 进行交互。 请参阅 IBM MQ Telemetry Transport 格式和协议。

要简化 MQTT 客户机应用程序的编写，请使用封装多个平台的 MQTT protocol 的 C， Java 和 JavaScript 客户机库。 如果将这些库合并到 MQTT 应用程序中，那么功能齐全的 MQTT 客户机可以短于 15 行代码。 MQTT 客户机库可从 Eclipse Paho 和 MQTT.org 免费获取。 请参阅 IBM MQ Telemetry Transport 样本程序。

MQTT 客户机应用程序始终负责启动与遥测通道的连接。 连接后， MQTT 客户机应用程序或 IBM MQ 应用程序可以启动消息交换。

MQTT 客户机应用程序和 IBM MQ 应用程序发布并预订同一组主题。 IBM MQ 应用程序还可以直接向 MQTT 客户机应用程序发送消息，而无需客户机应用程序首先创建预订。 请参阅 配置分布式排队以将消息发送到 MQTT 客户机。

MQTT 客户机应用程序使用遥测通道连接到 IBM MQ 。 遥测通道充当 MQTT 和 IBM MQ 所使用的不同类型消息之间的网桥。 它代表 MQTT 客户机应用程序在队列管理器中创建发布和预订。 遥测通道将与 MQTT 客户机应用程序预订匹配的发布从队列管理器发送到 MQTT 客户机应用程序。

相关概念

IBM MQ Telemetry 简介

人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性

在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

向 MQTT 客户机发送消息

IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

MQTT 客户机预订从 IBM MQ 接收的发布内容

执行任务 第 229 页的『从 IBM MQ Explorer 将消息发布到 MQTT 客户机实用程序』 以将发布从 IBM MQ 发送到 MQTT 客户机。

MQTT v3 客户机接收消息的标准方法是它创建对主题或一组主题的预订。 在示例代码片段 第 227 页的图 45 中， MQTT 客户机使用主题字符串 “MQTT Examples”进行预订。 IBM MQ C 应用程序 第 227 页的图 46 使用主题字符串 “MQTT Examples”发布到主题。 在代码片段 第 227 页的图 47 中， MQTT 客户机接收回调方法 `messageArrived` 中的发布内容。

有关如何配置 IBM MQ 以发送发布以响应来自 MQTT 客户机的预订的更多信息，请参阅 发布消息以响应 MQTT 客户机预订。

IBM MQ 应用程序将消息直接发送到 MQTT 客户机

执行任务 第 233 页的『使用 IBM MQ Explorer 向 MQTT 客户机发送消息』 以将消息直接从 IBM MQ 发送到 MQTT 客户机。

以这种方式发送到 MQTT 客户机的消息称为非请求消息。 MQTT v3 客户机作为主题名称集的发布接收自发消息。 遥测 (MQXR) 服务将主题名称设置为远程队列名称。

MQTT v3 客户机无法将自发消息发送到另一个 MQTT v3 客户机，也无法发送到 IBM MQ 队列。

有关如何配置 IBM MQ 以将消息直接发送到 MQTT 客户机的更多信息，请参阅 直接将消息发送到客户机。

MQTT 客户机发布消息

MQTT v3 客户机可以发布另一个 MQTT v3 客户机接收到的消息，但它无法发送未经请求的消息。 代码片段 第 228 页的图 48 显示了以 Java 编写的 MQTT v3 客户机如何发布消息。

用于将消息发送到一个特定 MQTT v3 客户机的典型模式是每个客户机创建对其自己的 `ClientIdentifier` 的预订。 执行任务 第 234 页的『将消息发布到特定 MQTT v3 客户机』 以使用 `ClientIdentifier` 作为主题字符串将消息从一个 MQTT 客户机发布到另一个 MQTT 客户机。

示例代码片段

第 227 页的图 45 中的代码片段显示了在 Java 中编写的 MQTT 客户机如何创建预订。 它还需要回调方法 `messageArrived` 来接收预订的发布。

```
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);
```

图 *45: MQTT v3* 客户机订户

第 227 页的图 46 中的代码片段显示了以 C 编写的 IBM MQ 应用程序如何发送发布内容。 将从以下任务中抽取代码片段: Create a publisher to a variable topic

```
/* 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);
```

图 *46: IBM MQ* 发布者

当发布到达时， MQTT 客户机将调用 MQTT application client `MqttCallback` 类的 `messageArrived` 方法。

```
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
}
```

图 *47: messageArrived* 方法

第 228 页的图 48 显示了 MQTT v3 将消息发布到 第 227 页的图 45 中创建的预订。

```
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);
```

图 *48: MQTT v3* 客户机发布程序

相关概念

IBM MQ Telemetry 简介
人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

从 IBM MQ Explorer 将消息发布到 MQTT 客户机实用程序

执行本任务中的步骤以使用 IBM MQ Explorer 发布消息，并使用 MQTT 客户机实用程序预订该消息。 其他任务显示如何配置队列管理器别名，而不是将缺省传输队列设置为 SYSTEM.MQTT.TRANSMIT.QUEUE。

开始之前

该任务假定您熟悉 IBM MQ 和 IBM MQ Explorer ，并且已安装 IBM MQ 和 IBM MQ Telemetry 功能部件。

为此任务创建队列管理器资源的用户必须具有足够的权限才能执行此操作。 出于演示目的，假定 IBM MQ Explorer 用户标识是 mqm 组的成员。

关于此任务

在该任务中，您将在 IBM MQ 中创建主题，并使用 MQTT 客户机实用程序预订该主题。 使用 IBM MQ Explorer 发布到主题时， MQTT 客户机将接收到该发布。

过程

请执行下列其中一项任务:

- 您已安装 IBM MQ Telemetry，但尚未将其启动。 执行任务: 第 230 页的『在尚未定义遥测 (MQXR) 服务的情况下启动任务』。
- 您之前已运行 IBM MQ 遥测，但希望使用新的队列管理器来执行演示。 执行任务: 第 230 页的『在尚未定义遥测 (MQXR) 服务的情况下启动任务』。
- 您希望使用未定义遥测资源的现有队列管理器来执行该任务。 您不希望运行 " **定义样本配置** " 向导。

  a. 执行下列其中一项任务以设置遥测:

  - 在 Linux 和 AIX 上配置队列管理器以进行遥测
  - 在 Windows 上配置队列管理器以进行遥测

b. 执行任务: 第 231 页的『使用正在运行的遥测 (MQXR) 服务启动任务』

- 如果要使用已定义遥测资源的现有队列管理器来执行任务，请执行以下任务: 第 231 页的『使用正在运行的遥测 (MQXR) 服务启动任务』。

下一步做什么

执行 第 233 页的『使用 IBM MQ Explorer 向 MQTT 客户机发送消息』 以将消息直接发送到客户机实用程序。

相关任务

使用 IBM MQ Explorer 向 MQTT 客户机发送消息
通过使用 IBM MQ Explorer 将消息放入 IBM MQ 队列，将消息发送到 MQTT 客户机实用程序。 此任务说明如何配置远程队列定义以将消息直接发送到 MQTT 客户机。

将消息发布到特定 MQTT v3 客户机
将消息从一个 MQTT v3 客户机发布到另一个客户机，将 `ClientIdentifier` 用作主题名称，将 IBM MQ 用作发布/预订代理。

在尚未定义遥测 *(MQXR)* 服务的情况下启动任务

创建队列管理器并运行 **定义样本配置** 以定义队列管理器的样本遥测资源。 使用 IBM MQ Explorer 发布消息，并使用 MQTT 客户机实用程序对其进行预订。

关于此任务

使用 **定义样本配置**设置样本遥测资源时，向导会设置访客用户标识许可权。 请仔细考虑是否希望以这种方式对访客用户标识进行授权。 `guest` on Windows 和 `nobody` on Linux 被授予发布和预订主题树的根以及将消息放入 `SYSTEM.MQTT.TRANSMIT.QUEUE` 的许可权。

向导还将缺省传输队列设置为 `SYSTEM.MQTT.TRANSMIT.QUEUE`，这可能会干扰在现有队列管理器上运行的应用程序。 可以但费力地配置遥测，而不使用缺省传输队列; 请执行以下任务: 第 232 页的『使用队列管理器别名』。 在此任务中，创建队列管理器以避免干扰任何现有缺省传输队列的可能性。

过程

1. 使用 IBM MQ Explorer 创建并启动新的队列管理器。
   a) 右键单击 `Queue Managers` 文件夹> **新建** > **队列管理器 ...**。 输入队列管理器名称> **完成**。

   组成队列管理器名称; 例如， `MQTTQMGR`。
2. 创建并启动遥测 (MQXR) 服务，并创建样本遥测通道。
   a) 打开 `Queue Managers\`*`QmgrName`*`\Telemetry` 文件夹。
   b) 单击 **定义样本配置 ...** > **完成**

   保持选中 **启动 MQTT 客户机实用程序** 复选框。
3. 使用 MQTT 客户机实用程序为 `MQTT Example` 创建预订。
   a) 单击**连接**。

   客户机历史记录 记录 `Connected` 事件。
   b) 在 **预订 \ 主题** 字段> **预订**中输入 `MQTT Example` 。

   客户机历史记录 记录 `Subscribed` 事件。
4. 在 IBM MQ 中创建 `MQTTExampleTopic` 。
   a) 右键单击 IBM MQ 资源管理器> **新建** > **主题**中的 `Queue Managers\`*`QmgrName`*`\Topics` 文件夹。
   b) 输入 `MQTTExampleTopic` 作为 **名称** > **下一步**。
   c) 输入 `MQTT Example` 作为 **主题字符串** > **完成**。
   d) 单击 **确定** 以关闭确认窗口。
5. 使用 IBM MQ Explorer 将 `Hello World!` 发布到主题 `MQTT Example` 。
   a) 单击 IBM MQ Explorer 中的 `Queue Managers\`*`QmgrName`*`\Topics` 文件夹。

b) 右键单击 MQTTExampleTopic > **测试发布 ...**

c) 在 **消息数据** 字段> **发布消息** > 切换到 " MQTT 客户机实用程序" 窗口中输入 Hello World! 。

客户机历史记录 记录 Received 事件。

使用正在运行的遥测 *(MQXR)* 服务启动任务

创建遥测通道和主题。 授权用户使用主题和遥测传输队列。 使用 IBM MQ Explorer 发布消息，并使用 MQTT 客户机实用程序对其进行预订。

开始之前

在此版本的任务中，定义并运行了队列管理器 *QmgrName*。 定义并运行遥测 (MQXR) 服务。 遥测 (MQXR) 服务可能是手动创建的，也可能是通过运行 " **定义样本配置** " 向导创建的。

关于此任务

在此任务中，您将配置现有队列管理器以将发布发送到 MQTT 客户机实用程序。

任务的步骤 第 231 页的『1』 将缺省传输队列设置为 SYSTEM.MQTT.TRANSMIT.QUEUE，这可能会干扰在现有队列管理器上运行的应用程序。 可以但费力地配置遥测，而不使用缺省传输队列; 请执行以下任务: 第 232 页的『使用队列管理器别名』。

过程

1. 将 SYSTEM.MQTT.TRANSMIT.QUEUE 设置为缺省传输队列。
   a) 右键单击 Queue Managers*QmgrName* folder > **属性 ...**
   b) 在导航器中单击 **通信** 。
   c) 单击 **选择 ...** > 选择 SYSTEM.MQTT.TRANSMIT.QUEUE > **确定** > **确定**。
2. 创建遥测通道 MQTTExampleChannel 以将 MQTT 客户机实用程序连接到 IBM MQ，并启动 MQTT 客户机实用程序。
   a) 右键单击 IBM MQ 资源管理器> **新建** > **遥测通道 ...**中的 Queue Managers*QmgrName*\Telemetry\Channels 文件夹。
   b) 在 **通道名称** 字段中输入 MQTTExampleChannel > **Next** > **Next**。
   c) 将客户机授权面板上的 **固定用户标识** 更改为要发布和预订 MQTTExample > **下一步**的用户标识。
   d) 保持选中 **启动客户机实用程序** > **完成**。
3. 使用 MQTT 客户机实用程序为 MQTT Example 创建预订。
   a) 单击**连接**。

      客户机历史记录 记录 Connected 事件。
   b) 在 **预订 \ 主题** 字段> **预订**中输入 MQTT Example 。

   客户机历史记录 记录 Subscribed 事件。
4. 在 IBM MQ 中创建 MQTTExampleTopic 。
   a) 右键单击 IBM MQ 资源管理器> **新建** > **主题**中的 Queue Managers*QmgrName*\Topics 文件夹。
   b) 输入 MQTTExampleTopic 作为 **名称** > **下一步**。
   c) 输入 MQTT Example 作为 **主题字符串** > **完成**。
   d) 单击 **确定** 以关闭确认窗口。
5. 如果您希望用户 (不在 mqm 组中) 发布和预订 MQTTExample 主题，请执行以下操作:
   a) 授权用户发布和预订主题 MQTTExampleTopic:

```
setmqaut -m qMgrName -t topic -n MQTTExampleTopic -p User ID -all +pub +sub
```

   b) 授权用户将消息放入 SYSTEM.MQTT.TRANSMIT.QUEUE:

```
setmqaut -m qMgrName -t q -n SYSTEM.MQTT.TRANSMIT.QUEUE -p User ID -all +put
```

6. 使用 IBM MQ Explorer 将 Hello World! 发布到主题 MQTT Example。
   a) 单击 IBM MQ Explorer 中的 Queue Managers*QmgrName*\Topics 文件夹。
   b) 右键单击 MQTTExampleTopic > **测试发布 ...**
   c) 在 **消息数据** 字段> **发布消息** > 切换到 " MQTT 客户机实用程序" 窗口中输入 Hello World!。

   客户机历史记录 记录 Received 事件。

使用队列管理器别名

使用 IBM MQ Explorer 将消息发布到 MQTT 客户机实用程序，而不将缺省传输队列设置为 SYSTEM.MQTT.TRANSMIT.QUEUE。

该任务是先前任务的延续，并使用队列管理器别名来避免将缺省传输队列设置为 SYSTEM.MQTT.TRANSMIT.QUEUE。

开始之前

完成任务 第 230 页的『在尚未定义遥测 (MQXR) 服务的情况下启动任务』 或任务 第 231 页的『使用正在运行的遥测 (MQXR) 服务启动任务』。

关于此任务

当 MQTT 客户机创建预订时， IBM MQ 将使用 ClientIdentifier 作为远程队列管理器名称来发送其响应。 在此任务中，它使用 ClientIdentifier MyClient。

如果没有名为 MyClient 的传输队列或队列管理器别名，那么响应将放在缺省传输队列上。 通过将缺省传输队列设置为 SYSTEM.MQTT.TRANSMIT.QUEUE， MQTT 客户机将获取响应。

您可以避免使用队列管理器别名将缺省传输队列设置为 SYSTEM.MQTT.TRANSMIT.QUEUE 。 必须为每个 ClientIdentifier 设置队列管理器别名。 通常，客户机过多，以致无法实际使用队列管理器别名。 通常， ClientIdentifier 不可预测，因此无法以此方式配置遥测。

但是，在某些情况下，您可能必须将缺省传输队列配置为 SYSTEM.MQTT.TRANSMIT.QUEUE 以外的传输队列。 过程 中的步骤配置队列管理器别名，而不是将缺省传输队列设置为 SYSTEM.MQTT.TRANSMIT.QUEUE。

过程

1. 除去 SYSTEM.MQTT.TRANSMIT.QUEUE 作为缺省传输队列。
   a) 右键单击 Queue Managers*QmgrName* folder > **属性 ...**
   b) 在导航器中单击 **通信** 。
   c) 从 **缺省传输队列** 字段中除去 SYSTEM.MQTT.TRANSMIT.QUEUE > **确定**。
2. 检查您是否无法再使用 MQTT 客户机实用程序创建预订:
   a) 单击**连接**。

      客户机历史记录 记录 Connected 事件。
   b) 在 **预订 \ 主题** 字段> **预订**中输入 MQTT Example 。

   客户机历史记录 记录 Subscribe failed 和 Connection lost 事件。
3. 为 ClientIdentifier MyClient 创建队列管理器别名。
   a) 右键单击 Queue Managers*QmgrName*\Queues 文件夹> **新建** > **远程队列定义**。
   b) 将定义命名为 MyClient > **Next**。
   c) 在 **远程队列管理器** 字段中输入 MyClient 。
   d) 在 **传输队列** 字段> **完成**中输入 SYSTEM.MQTT.TRANSMIT.QUEUE 。

4. 再次连接 MQTT 客户机实用程序。
   a) 检查 **客户机标识** 是否设置为 MyClient。
   b) **连接**

   客户机历史记录 记录 Connected 事件。
5. 使用 MQTT 客户机实用程序为 MQTT Example 创建预订。
   a) 单击**连接**。

      客户机历史记录 记录 Connected 事件。
   b) 在 **预订 \ 主题** 字段> **预订**中输入 MQTT Example 。

   客户机历史记录 记录 Subscribed 事件。
6. 使用 IBM MQ Explorer 将 Hello World! 发布到主题 MQTT Example 。
   a) 单击 IBM MQ Explorer 中的 Queue Managers*QmgrName*\Topics 文件夹。
   b) 右键单击 MQTTExampleTopic > **测试发布 ...**
   c) 在 **消息数据** 字段> **发布消息** > 切换到 " MQTT 客户机实用程序" 窗口中输入 Hello World! 。

   客户机历史记录 记录 Received 事件。

使用 IBM MQ Explorer 向 MQTT 客户机发送消息

通过使用 IBM MQ Explorer 将消息放入 IBM MQ 队列，将消息发送到 MQTT 客户机实用程序。 此任务说明如何配置远程队列定义以将消息直接发送到 MQTT 客户机。

开始之前

执行任务 第 229 页的『从 IBM MQ Explorer 将消息发布到 MQTT 客户机实用程序』。 使 MQTT 客户机实用程序保持连接。

关于此任务

此任务演示如何使用队列将消息发送到 MQTT 客户机，而不是发布到主题。 您不会在客户机中创建预订。 该任务的步骤 第 233 页的『2』 演示已删除先前的预订。

过程

1. 通过断开连接并重新连接 MQTT 客户机实用程序来废弃任何现有预订。

   将废弃预订，因为除非您更改缺省值，否则 MQTT 客户机实用程序将与清除会话连接; 请参阅 清除会话。

   为了更容易执行该任务，请输入您自己的 ClientIdentifier，而不是使用由 MQTT 客户机实用程序创建的生成的 ClientIdentifier 。
   a) 单击 **断开连接** 以断开 MQTT 客户机实用程序与遥测通道的连接。

      客户机历史记录 记录 Disconnected 事件
   b) 将 **客户机标识** 更改为 MyClient。
   c) 单击**连接**。

   客户机历史记录 记录 Connected 事件
2. 检查 MQTT 客户机实用程序是否不再接收 MQTTExampleTopic 的发布。
   a) 单击 IBM MQ Explorer 中的 Queue Managers*QmgrName*\Topics 文件夹。
   b) 右键单击 MQTTExampleTopic > **测试发布 ...**
   c) 在 **消息数据** 字段> **发布消息** > 切换到 " MQTT 客户机实用程序" 窗口中输入 Hello World! 。

   客户机历史记录中未记录任何事件。
3. 为客户机创建远程队列定义。

将 ClientIdentifier MyClient 设置为远程队列定义中的远程队列管理器名称。 使用您喜欢的任何名称作为远程队列名称。 远程队列名称将作为主题名称传递到 MQTT 客户机。

a) 右键单击 Queue Managers*QmgrName*\Queues 文件夹> **新建** > **远程队列定义**。

b) 将定义命名为 MyClientRemoteQueue > **Next**。

c) 在 **远程队列** 字段中输入 MQTTExampleQueue 。

d) 在 **远程队列管理器** 字段中输入 MyClient 。

e) 在 **传输队列** 字段> **完成**中输入 SYSTEM.MQTT.TRANSMIT.QUEUE 。

4. 将测试消息放在 MyClientRemoteQueue 上。

a) 右键单击 **MyClientRemoteQueue** > **放置测试消息 ...**

b) 在 "消息数据" 字段中输入 Hello queue! > **放入消息** > **关闭**

客户机历史记录 记录 Received 事件。

5. 除去 SYSTEM.MQTT.TRANSMIT.QUEUE 作为缺省传输队列。

a) 右键单击 Queue Managers*QmgrName* folder > **属性 ...**

b) 在导航器中单击 **通信** 。

c) 从 **缺省传输队列** 字段中除去 SYSTEM.MQTT.TRANSMIT.QUEUE > **确定**。

6. 重做步骤 第 234 页的『4』。

MyClientRemoteQueue 是明确指定传输队列的远程队列定义。 您不需要定义缺省传输队列以将消息发送到 MyClient。

下一步做什么

如果缺省传输队列不再设置为 SYSTEM.MQTT.TRANSMIT.QUEUE，那么除非为 ClientIdentifier MyClient 定义了队列管理器别名，否则 MQTT 客户机实用程序无法创建新的预订。 将缺省传输队列复原到 SYSTEM.MQTT.TRANSMIT.QUEUE。

相关任务

从 IBM MQ Explorer 将消息发布到 MQTT 客户机实用程序
执行本任务中的步骤以使用 IBM MQ Explorer 发布消息，并使用 MQTT 客户机实用程序预订该消息。 其他任务显示如何配置队列管理器别名，而不是将缺省传输队列设置为 SYSTEM.MQTT.TRANSMIT.QUEUE。

将消息发布到特定 MQTT v3 客户机
将消息从一个 MQTT v3 客户机发布到另一个客户机，将 ClientIdentifier 用作主题名称，将 IBM MQ 用作发布/预订代理。

将消息发布到特定 MQTT v3 客户机

将消息从一个 MQTT v3 客户机发布到另一个客户机，将 ClientIdentifier 用作主题名称，将 IBM MQ 用作发布/预订代理。

开始之前

执行任务 第 229 页的『从 IBM MQ Explorer 将消息发布到 MQTT 客户机实用程序』。 使 MQTT 客户机实用程序保持连接。

关于此任务

该任务演示了两件事:

1. 在一个 MQTT 客户机中预订主题，并从另一个 MQTT 客户机接收发布内容。
2. 通过使用 ClientIdentifier 作为主题字符串来设置 "点到点" 预订。

附加任务 第 236 页的『将设备的 MQTT 守护程序用作发布/预订代理程序』使用设备的 MQTT 守护程序作为发布/预订代理，而不是 IBM MQ。

注: 设备的守护程序不再可用。 有关备用解决方案，请参阅 eclipse.org “Mosquitto” 项目 (https://eclipse.org/mosquitto)。

过程

1. 通过断开连接并重新连接 MQTT 客户机实用程序来废弃任何现有预订。

   将废弃预订，因为除非您更改缺省值，否则 MQTT 客户机实用程序将与清除会话连接; 请参阅 清除会话。

   为了更容易执行该任务，请输入您自己的 `ClientIdentifier`，而不是使用由 MQTT 客户机实用程序创建的生成的 `ClientIdentifier` 。

   a) 单击 **断开连接** 以断开 MQTT 客户机实用程序与遥测通道的连接。

   客户机历史记录 记录 `Disconnected` 事件

   b) 将 **客户机标识** 更改为 `MyClient`。

   c) 单击**连接**。

   客户机历史记录 记录 `Connected` 事件

2. 创建主题 `MyClient` 的预订

   `MyClient` 是此客户机的 `ClientIdentifier` 。

   a) 在 **预订 \ 主题** 字段> **预订**中输入 `MyClient` 。

   客户机历史记录 记录 `Subscribed` 事件。

3. 启动另一个 MQTT 客户机实用程序。

   a) 打开 `Queue Managers\`*`QmgrName`*`\Telemetry\channels` 文件夹。

   b) 右键单击 **PlainText** 通道> **运行 MQTT 客户机实用程序 ...**

   c) 单击**连接**。

   客户机历史记录 记录 `Connected` 事件

4. 将 `Hello MyClient!` 发布到主题 `MyClient`。

   a) 从使用 `ClientIdentifier MyClient` 运行的 MQTT 客户机实用程序复制预订主题 `MyClient`。

   b) 将 `MyClient` 粘贴到每个 MQTT 客户机实用程序实例的 **发布 \ 主题** 字段中。

   c) 在 **Publication \ message** 字段中输入 `Hello MyClient!` 。

   d) 在这两个实例中单击 **发布** 。

结果

具有 `ClientIdentifier MyClient` 的 MQTT 客户机实用程序中的 **客户机历史记录** 记录了两个 **已接收** 事件和一个 **已发布** 事件。 另一个 MQTT 客户机实用程序实例记录一个 **已发布** 事件。

如果只看到一个 **已接收** 事件，请检查以下可能的原因:

1. 队列管理器的缺省传输队列是否设置为 `SYSTEM.MQTT.TRANSMIT.QUEUE` ?
2. 是否在执行其他练习时创建了引用 `MyClient` 的队列管理器别名或远程队列定义? 如果存在配置问题，请删除引用 `MyClient` 的任何资源，例如队列管理器别名或传输队列。 断开客户机实用程序的连接，停止并重新启动遥测 (MQXR) 服务。

下一步做什么

执行下一个任务 第 236 页的『将设备的 MQTT 守护程序用作发布/预订代理程序』。 MQTT 客户机实用程序连接到设备的 MQTT 守护程序，而不是连接到遥测通道。

相关任务

从 IBM MQ Explorer 将消息发布到 MQTT 客户机实用程序
执行本任务中的步骤以使用 IBM MQ Explorer 发布消息，并使用 MQTT 客户机实用程序预订该消息。 其他任务显示如何配置队列管理器别名，而不是将缺省传输队列设置为 `SYSTEM.MQTT.TRANSMIT.QUEUE`。

使用 IBM MQ Explorer 向 MQTT 客户机发送消息
通过使用 IBM MQ Explorer 将消息放入 IBM MQ 队列，将消息发送到 MQTT 客户机实用程序。 此任务说明如何配置远程队列定义以将消息直接发送到 MQTT 客户机。

将设备的 *MQTT* 守护程序用作发布/预订代理程序

将设备的 MQTT 守护程序用作发布/预订代理程序，而不是 IBM MQ。 通过使用其 `ClientIdentifier` 作为主题字符串进行预订，发布一条消息，其中包含要发送到另一个实例的 MQTT 客户机实用程序的一个实例。

开始之前

注: 设备的守护程序不再可用。 有关备用解决方案，请参阅 eclipse.org “Mosquitto” 项目 (https://eclipse.org/mosquitto)。

如果尚未安装 第 249 页的『设备的 MQTT 守护程序』，请进行安装。

请勿运行验证; 它使用端口 1883 ， 该端口已由 `PlainText` 遥测通道使用。

关于此任务

在该任务中，使用非缺省 TCP/IP 端口将 MQTT 客户机连接到设备的 MQTT 守护程序。 一个客户机使用其 `ClientIdentifier` 作为主题字符串进行预订，另一个客户机以与先前任务相同的方式发布到 `ClientIdentifier`，请参阅 过程。

注: 记录了用于在 Windows 上运行守护程序的任务。 要在 Linux 上运行守护程序，请修改配置文件的路径和许可权。

过程

1. 在包含设备的 MQTT 守护程序的目录中打开命令窗口。
2. 在另一个 TCP/IP 端口上运行守护程序。
   a) 在守护程序所在的目录中创建名为 `<configfile name>.cfg` 的文件。
   b) 向文件添加一行以配置守护程序的其他缺省端口。

   ```
   port 1884
   ```

   c) 保存文件。
3. 启动该守护程序。

   ```
   amqtdd
   ```

   守护程序将其控制台日志写入命令窗口:

   ```
   20100712 123133.857 CWNAN9999I MQTT daemon for devices
   20100712 123133.857 CWNAN9997I Licensed Materials - Property of IBM
   20100712 123133.857 CWNAN9996I Copyright IBM Corp. 2007, 2023. All Rights Reserved
   20100712 123133.857 CWNAN9995I US Government Users Restricted Rights ...
   20100712 123133.857 CWNAN0049I Configuration file name is .\<configfile name>.cfg
   20100712 123133.873 CWNAN0054I Features included: bridge
   20100712 123134.060 CWNAN0014I MQTT protocol starting, listening on port 1884
   ```

4. 启动 MQTT 客户机实用程序的实例。

   仅从遥测通道启动 MQTT 客户机实用程序，然后可以连接到守护程序。 Alternatively you can install the <configfile name> SupportPac, IA92. SupportPac 可从 IA92: WBI 代理- IBM MQ Telemetry 传输的 Java 实现获取。

   a) 打开 `Queue Managers\`*`QmgrName`*`\Telemetry\channels` 文件夹。
   b) 右键单击 **PlainText** 通道> **运行 MQTT 客户机实用程序 ...**
   c) 将 **端口** 更改为 1884。
   d) 将 **客户机标识** 更改为 `MyClient`。
   e) 单击**连接**。

客户机历史记录 记录 Connected 事件

5. 创建主题 MyClient 的预订

   MyClient 是此客户机的 ClientIdentifier 。

   a) 在 **预订 \ 主题** 字段> **预订**中输入 MyClient 。

   客户机历史记录 记录 Subscribed 事件。

6. 启动另一个 MQTT 客户机实用程序。

   a) 打开 Queue Managers*QmgrName*\Telemetry\channels 文件夹。

   b) 右键单击 **PlainText** 通道> **运行 MQTT 客户机实用程序 ...**

   c) 将 **端口** 更改为 1884。

   d) 单击**连接**。

   客户机历史记录 记录 Connected 事件

7. 将 Hello MyClient! 发布到主题 MyClient。

   a) 从使用 ClientIdentifier MyClient 运行的 MQTT 客户机实用程序复制预订主题 MyClient。

   b) 将 MyClient 粘贴到每个 MQTT 客户机实用程序实例的 **发布 \ 主题** 字段中。

   c) 在 **Publication \ message** 字段中输入 Hello MyClient! 。

   d) 在这两个实例中单击 **发布** 。

结果

具有 ClientIdentifier MyClient 的 MQTT 客户机实用程序中的 **客户机历史记录** 记录了两个 **已接收** 事件和一个 **已发布** 事件。 另一个 MQTT 客户机实用程序实例记录一个 **已发布** 事件。

您还可以在命令窗口中监视与设备的 MQTT 守护程序的连接和断开连接事件。

从 MQTT 客户机向 IBM MQ 应用程序发送消息

IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

执行任务 第 238 页的『从 MQTT 客户机将消息发布到 IBM MQ』，以了解如何将发布从 MQTT 客户机发送到 IBM MQ 中定义的预订。

如果主题是集群主题，或者使用发布/预订层次结构进行分发，那么预订可以位于与 MQTT 客户机所连接的队列管理器不同的队列管理器上。

相关概念

IBM MQ Telemetry 简介
人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol

IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

从 MQTT 客户机将消息发布到 IBM MQ

使用 IBM MQ Explorer 创建主题预订，并使用 IBM MQTT 客户机实用程序发布到主题。

开始之前

执行任务 第 229 页的『从 IBM MQ Explorer 将消息发布到 MQTT 客户机实用程序』。 使 MQTT 客户机实用程序保持连接。

关于此任务

该任务演示使用 MQTT 客户机发布消息，并使用使用 IBM MQ Explorer 创建的非受管持久预订接收发布。

过程

1. 创建对主题字符串 MQTT Example 的持久预订。

   执行以下步骤以使用 IBM MQ Explorer 创建队列和预订。

   a) 右键单击 IBM MQ 资源管理器> **新建** > **本地队列 ...**中的 Queue Managers*QmgrName*\Queues 文件夹。

   b) 输入 MQTTExampleQueue 作为队列名称> **完成**。

   c) 右键单击 IBM MQ 资源管理器> **新建** > **预订 ...**中的 Queue Managers*QmgrName*\Subscriptions 文件夹。

   d) 输入 MQTTExampleSubscription 作为队列名称> **Next**。

   e) 单击 **选择 ...** > MQTTExampleTopic > **确定**。

   您已在 第 229 页的『从 IBM MQ Explorer 将消息发布到 MQTT 客户机实用程序』的步骤 第 230 页的『4』 中创建主题 MQTTExampleTopic 。

   f) 输入 MQTTExampleQueue 作为目标名称> **完成**。

2. 作为可选步骤，在没有 mqm 权限的情况下设置队列以供其他用户使用。

   如果要为权限低于 mqm 的用户设置配置，那么必须将 put 和 get 权限授予 MQTTExampleQueue。 在 第 229 页的『从 IBM MQ Explorer 将消息发布到 MQTT 客户机实用程序』中配置了对主题和传输队列的访问权。

   a) 授权用户放入并进入队列 MQTTExampleQueue:

   ```
   setmqaut -m qMgrName -t queue -n MQTTExampleQueue -p User ID -all +put +get
   ```

3. 使用 MQTT 客户机实用程序将 Hello IBM MQ! 发布到主题 MQTT Example 。

   如果尚未使 MQTT 客户机实用程序保持连接，请右键单击 **PlainText** 通道> **运行 MQTT 客户机实用程序 ...** > **连接**。

   a) 在 **发布 \ 主题** 字段中输入 MQTT Example 。

   b) 在 **发布 \ 消息** 字段> **发布**中输入 Hello IBM MQ! 。

4. 打开 Queue Managers*QmgrName*\Queues 文件夹并查找 MQTTExampleQueue。

   当前队列深度 字段为 1

5. 右键单击 MQTTExampleQueue > **浏览消息 ...** 并检查该出版物。

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息

执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

开始之前

注: 设备的守护程序不再可用。 有关备用解决方案，请参阅 eclipse.org “Mosquitto” 项目 (https://eclipse.org/mosquitto)。

执行任务 第 234 页的『将消息发布到特定 MQTT v3 客户机』 和 第 236 页的『将设备的 MQTT 守护程序用作发布/预订代理程序』 以熟悉使用 MQTT 客户机实用程序。 完成任务后，保留 MQTT 客户机实用程序的一个实例与设备的 MQTT 守护程序连接。 保留另一个实例与遥测通道连接。

该任务假定您已定义到遥测服务的通道，该服务正在侦听地址 127.0.0.1 上的端口 1883 。 同样，缺省守护程序侦听器配置为侦听地址 127.0.0.1 上的端口 1884 。 配置文件中的一行 (与守护程序存储在同一目录中) 配置缺省守护程序侦听器端口。

```
port 1884
```

关于此任务

在此任务中，您更新正在运行的守护程序以创建到 IBM MQ 遥测 (MQXR) 服务的连接网桥，然后与守护程序交换消息。

过程

1. 确保正在运行两个 MQTT 客户机实用程序实例。 一个连接到端口 1884 上的守护程序，另一个连接到端口 1883 上运行的遥测通道。
2. 使用文件中的以下命令在守护程序所在的目录中创建该文件。

```
connection daemon1
address 127.0.0.1:1883
topic # in  import/ export/
topic # out export/ import/
try_private false
```

   - 网桥称为 daemon1，它连接到为在套接字地址 127.0.0.1:1883 上运行的遥测 (MQXR) 服务配置的通道。 try_private 命令是可选的; true 是缺省值。 如果没有此线路，那么网桥首先尝试使用设备的 MQTT 守护程序所理解的专用协议进行连接。 在命令中包含 try_private false 可避免此步骤，并加快完成成功连接的时间。
   - 行 topic # in import/ export/指示 daemon1 预订与队列管理器中创建的主题字符串 export/# 匹配的所有主题。 它将匹配的发布从队列管理器传输到守护程序，将主题字符串的开头从 export/ 更改为 import/。 行 topic # out export/ import/在本地守护程序上创建预订。 网桥预订与守护程序中创建的主题字符串 export/# 匹配的所有主题。 它将发布从守护程序传输到队列管理器，将主题字符串的开头从 export/ 更改为 import/。

   第 240 页的图 49 显示了生成的控制台日志。

```
CWNAN0124I Starting bridge connection daemon1
CWNAN0133I Bridge connection daemon1 to 127.0.0.1:1883 now established
```

   图 *49:* 来自启动连接网桥的控制台日志

3. 在 MQTT 客户机实用程序的每个实例中，在 **预订/主题:** 输入字段> **预订**中输入 import/# 。
4. 在 MQTT 客户机实用程序的每个实例中，在 **Publication/Topic:** 输入字段中输入 export/# 。
   a) 在连接到端口 1883(遥测通道) 的 MQTT 客户机实用程序中，在 **发布/消息:** 输入字段> **发布**中输入 From the queue manager 。
   b) 在连接到端口 1884 的 MQTT 客户机实用程序 MQTT 守护程序中，在 **发布/消息:** 输入字段> **发布**中输入 From the daemon 。

   每个 MQTT 客户机实用程序中的客户机历史记录都会显示已从一个代理程序传输到另一个代理程序的发布。

相关概念

IBM MQ Telemetry 简介

人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例

遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性

在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

MQTT 发布/预订应用程序

使用基于主题的发布/预订来编写 MQTT 应用程序。

当 MQTT 客户机连接时，发布在客户机与服务器之间的任一方向上流动。 在客户机上发布信息时，将从客户机发送这些发布。 当将消息发布到与客户机创建的预订匹配的主题时，将在客户机上接收到发布内容。

IBM MQ 发布/预订代理程序管理 MQTT 客户机创建的主题和预订。 MQTT 客户机创建的主题与 IBM MQ 应用程序创建的主题共享相同的主题空间。

将与 MQTT 客户机预订中的主题字符串匹配的发布放在 SYSTEM.MQTT.TRANSMIT.QUEUE 上，并将远程队列管理器名称设置为客户机的 `ClientIdentifier`。 遥测 (MQXR) 服务将发布转发到创建预订的客户机。 它使用已设置为远程队列管理器名称的 `ClientIdentifier` 来标识客户机。

通常，必须将 SYSTEM.MQTT.TRANSMIT.QUEUE 定义为缺省传输队列。 可以将 MQTT 配置为不使用缺省传输队列，但这很繁重; 请参阅 配置分布式排队以将消息发送到 MQTT 客户机。

MQTT 客户机可以创建持久会话; 请参阅 第 251 页的『MQTT 无状态和有状态会话』。 在持久会话中创建的预订是持久的。 针对具有持久会话的客户机的发布将存储在 SYSTEM.MQTT.TRANSMIT.QUEUE 中，并在客户机重新连接时转发到该客户机。

MQTT 客户机还可以发布和预订保留的发布; 请参阅 保留的发布和 MQTT 客户机。 保留发布主题的订户接收到该主题的最新发布。 订户在创建预订时或在重新连接到其先前会话时接收保留的发布内容。

相关概念

IBM MQ Telemetry 简介
人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息

IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

遥测应用程序

使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

使用 JMS， MQI 或其他 IBM MQ 编程接口来编程 IBM MQ 中的遥测应用程序。

遥测 (MQXR) 服务在 MQTT v3 消息和 IBM MQ 消息之间进行转换。 它代表 MQTT 客户机创建预订和发布，并将发布转发到 MQTT 客户机。 发布是 MQTT v3 消息的有效内容。 有效内容包含消息头和 `jms-bytes` 格

式的字节数组。 遥测服务器在 MQTT v3 消息与 IBM MQ 消息之间映射头; 请参阅 第 245 页的『IBM MQ Telemetry 与队列管理器的集成』。

使用 Publication， MQInput 和 JMSInput 节点在 IBM Integration Bus 和 MQTT 客户机之间发送和接收发布。

通过使用消息流，您可以使用 HTTP 将遥测与 Web 站点以及其他使用 IBM MQ 和 WebSphere Adapters 的应用程序集成。

相关概念

IBM MQ Telemetry 简介
人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合

MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

IBM MQ Telemetry 与队列管理器的集成

MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

不支持与 IBM MQ 提供的主题，发布，预订和消息相关联的许多属性。 第 245 页的『MQTT 客户机到 IBM MQ 发布/预订代理程序』 和 第 246 页的『IBM MQ 到 MQTT 客户机』 描述如何设置发布属性。 这些设置取决于发布是要进入还是从 IBM MQ 发布/预订代理程序进行。

在 IBM MQ 中，发布/预订主题与管理主题对象相关联。 MQTT 客户机创建的主题没有任何不同。 当 MQTT 客户机为发布创建主题字符串时， IBM MQ 发布/预订代理程序会将其与管理主题对象相关联。 代理程序将出版物中的主题字符串映射到最近的管理主题对象父代。 此映射与 IBM MQ 应用程序的映射相同。 如果没有用户创建的主题，那么发布主题将映射到 SYSTEM.BASE.TOPIC。 应用于发布的属性派生自主题对象。

当 IBM MQ 应用程序或管理员创建预订时，将命名该预订。 使用 MQ Explorer 或使用 **runmqsc** 或 PCF 命令列出预订。 已命名全部 MQTT 客户机预订。 将为其提供以下格式的名称: *ClientIdentifier*:*Topic name*

MQTT 客户机到 IBM MQ 发布/预订代理程序

MQTT 客户机已将发布发送到 IBM MQ。 遥测 (MQXR) 服务会将发布内容转换为 IBM MQ 消息。 IBM MQ 消息包含三个部分:

1. MQMD
2. RFH2
3. 消息

MQMD 属性设置为其缺省值，但 第 246 页的表 14 中注明的属性除外。

表 *14: MQMD*		
MQMD 字段	**类型**	**值**
Format	MQCHAR8	MQFMT_RF_HEADER_2
UserIdentifier	MQCHAR12	设置为下列其中一项: MqttClient.ClientIdentifier MqttConnectOptions.UserName IBM MQ 管理员为遥测通道设置的用户标识。
Priority	MQLONG	MQPRI_PRIORITY_AS_Q_DEF（IBM MQ 的缺省值，与缺省值为 4 的 JMS 不同。）
Persistence	MQLONG	QoS=0→MQPER_NOT_PERSISTENT QoS=1→MQPER_PERSISTENT QoS=2→MQPER_PERSISTENT

RFH2 头不包含用于定义 JMS 消息类型的 <msd> 文件夹。 遥测 (MQXR) 服务会将 IBM MQ 消息创建为缺省 JMS 消息。 缺省 JMS 消息类型为 jms-bytes 消息。 应用程序可以将其他头信息作为消息属性进行访问; 请参阅 消息属性。

RFH2 值设置为如 第 246 页的表 15 中所示。 "格式" 属性在 RFH2 固定头中设置，其他值在 RFH2 文件夹中设置。

表 *15: RFH2*		
RFH2 属性	**类型/文件夹**	**头**
FORMAT	MQCHAR8	MQFMT_NONE
ClientIdentifier	mqtt/clientId	复制长度为 1...23 字节的 MqttClient.ClientIdentifier 。
QoS	mqtt/qos	从入局 MQTT 消息复制 QoS 。
消息标识	mqtt/msgid	如果 QoS 为 1 或 2，请从入局 MQTT 消息复制 **消息标识** 。
MQIsRetained	mqps/Ret	如果原始 MQTT 发布是使用 RETAIN 属性集发送的，并且消息是作为保留发布接收的，请进行设置。
MQTopicString	mqps/Top	将 MQTT 消息发布到的主题。

MQTT 发布中的有效内容将映射到 IBM MQ 消息的内容:

表 *16:* 消息内容		
消息内容	**类型**	**内容**
缓冲区	MQBYTE *n*	从入局 MQTT 消息复制字节。 长度可以为零。

IBM MQ 到 MQTT 客户机

客户机已预订发布主题。 IBM MQ 应用程序已发布到主题，从而导致发布由 IBM MQ 发布/预订代理发送到 MQTT 订户。 或者， IBM MQ 应用程序已直接向 MQTT 客户机发送非请求消息。 第 247 页的表 17 描述了如何在发送到 MQTT 客户机的消息中设置固定消息头。 将废弃 IBM MQ 消息头或任何其他头中的任何其他

数据。 IBM MQ 消息中的消息数据将作为 MQTT 消息中的消息有效内容发送，而不进行任何更改。 MQTT 消息由遥测 (MQXR) 服务发送到 MQTT 客户机。

表 *17: MQTT* 固定头属性

MQTT 字段	**类型**	**值**
DUP	BOOLEAN	如果设置为 QoS = 1 或 2，并且在先前的传输中已将消息发送到此客户机，并且在一段时间后未确认该消息。
QoS	int	从 IBM MQ 中的发布/预订代理设置出局发布中 QoS 的值的方式取决于传入发布。 这取决于是从 MQTT 客户机发送传入发布，还是从 IBM MQ 应用程序发送传入发布。 **MQTT** 入局发布以及订户所请求的 QoS 中 QoS 的较小值。 **IBM MQ** 从传入发布派生的 QoS 的较小值: MQPER_NOT_PERSISTENT→QoS=0 MQPER_PERSISTENT→QoS=2 以及订户请求的 QoS 。 如果在没有预订的情况下将消息发送到客户机，那么缺省情况下会将 QoS 设置为 2。 客户机可以通过预订具有不同 QoS 的 DEFAULT.QoS 来更改此值。
RETAIN	BOOLEAN	如果传入发布具有保留属性集，请进行设置。

第 247 页的表 18 描述了如何在发送到 MQTT 客户机的 MQTT 消息中设置变量消息头。

表 *18: MQTT* 变量头属性

MQTT 字段	**类型**	**值**
Topic name	**字符串**	发布消息所使用的主题字符串。
Message ID	**字符串**	MQMD.MsgId 属性 (当它放置在 SYSTEM.MQTT.TRANSMIT.QUEUE 中时)。
Payload	byte[]	将字节从传入发布直接复制到发布/预订代理。 长度可以为零。

相关概念

IBM MQ Telemetry 简介

人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例

遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议

IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务

遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道

创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

设备的 MQTT 守护程序

设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

注: 设备的守护程序不再可用。 有关备用解决方案，请参阅 eclipse.org “Mosquitto” 项目 (https://eclipse.org/mosquitto)。

设备的守护程序具有以下基本用途:

- **在发布/预订网络中将本地 MQTT 客户机连接在一起**。

  您可以将设备的传感器和执行器作为单独的 MQTT 客户机连接到守护程序。 传感器将发布其测量读数，传动结构将预订读数，从而基于这些读数值修改行为。 将在本地对读数做出反应。

- **过滤要将哪些预订和消息发布到队列管理器和设备**。

  在先前的示例中， IBM Integration Bus 消息流可能会预订守护程序将读数发布到的主题。 该流将更新 Web 页面并显示设备的状态。

  守护程序还可能转发传动结构创建的针对队列管理器的预订。 IBM Integration Bus 流将消息发布到 MQTT 客户机为所预订的执行器提供服务的主题。 MQTT 客户机修改设备设置。

  消息流可以使用 IBM Integration Bus HTTPInput 节点从 Web 页面开始。

- **将多个 MQTT 客户机集中到与遥测服务器的一个连接中**。

  守护程序通过单个 TCP/IP 连接转发发布和预订，而不是每个设备单独连接到遥测服务器。 守护程序减少遥测 (MQXR) 服务所管理的 TCP/IP 连接数。

  个别 MQTT 客户机连接到守护程序。 各个客户机对于队列管理器不可见。 守护程序将代表所有相连的客户机，与队列管理器建立一个连接。

- **存储消息并在设备与队列管理器之间转发**

  守护程序负责防止遥测设备与队列管理器的短暂连接失败。

  设备可能仅支持“发出消息之后无需等待应答”消息传递。 如果与队列管理器的连接仅间歇可用，或者不可靠，那么设备将无法以可预测或可靠的方式传输信息。

  一种解决方案是：使用始终可用的本地连接，将设备连接到守护程序。 守护程序可在内存中缓存往来于队列管理器之间的消息。 它可以通过不可靠的连接，使用可靠的服务质量，与队列管理器相互传递消息。

注: 守护程序没有用于 “进行中” 消息的持久存储器。 消息会在内存中缓存。

设备的 MQTT 守护程序作为免费下载 IBM Messaging Telemetry Client SupportPac 的一部分提供。

相关概念

IBM MQ Telemetry 简介

人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例

遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议

IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务

遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道

创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol

IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

MQTT 无状态和有状态会话

MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

当 MQTT 客户机连接到遥测通道时，它将启动新会话或恢复旧会话。 新会话没有未确认的未完成消息，没有预订，也没有等待传递的发布。 当客户机连接时，它指定是从干净会话开始，还是恢复现有会话; 请参阅 干净会话。

如果客户机恢复现有会话，那么它将继续执行，就像连接未中断一样。 等待传递的发布将发送到客户机，并且未落实的任何消息传输都将完成。 当持久会话中的客户机与遥测 (MQXR) 服务断开连接时，客户机创建的任何预订都将保留。 预订的发布将在客户机重新连接时发送到客户机。 如果在不恢复旧会话的情况下重新连接，那么遥测 (MQXR) 服务将废弃这些发布。

会话状态信息由队列管理器保存在 SYSTEM.MQTT.PERSISTENT.STATE 队列中。

IBM MQ 管理员可以断开连接并清除会话。

相关概念

IBM MQ Telemetry 简介
人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

未连接 MQTT 客户机时

当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

如果您希望在客户机意外断开连接时收到通知，那么可以注册最后一个遗嘱和遗嘱发布; 请参阅 最后一个遗嘱和遗嘱发布。 它由遥测 (MQXR) 服务发送，如果它检测到与客户机的连接已断开，而客户机未发出请求。

客户机可以随时发布保留的出版物; 请参阅 保留的出版物和 MQTT 客户机。 对主题的新预订可以请求发送与主题关联的任何保留发布。 如果创建最后一个遗嘱和遗嘱作为保留发布，那么可以使用它来监视客户机的状态。

例如，客户机在连接时发布保留发布，以公布其可用性。 同时，它创建了一个保留的最后遗嘱和遗嘱出版物，宣布其不可用。 此外，就在它进行计划中的断开连接之前，它将其不可用性作为保留出版物发布。 要了解客户机是否可用，您将预订保留发布的主题。 您将始终收到三个出版物中的一个。

如果客户机要在断开连接时接收发布的消息，请将该客户机重新连接到其先前会话; 请参阅 第 251 页的『MQTT 无状态和有状态会话』。 其预订处于活动状态，直到删除这些预订或客户机创建干净的会话为止。

相关概念

IBM MQ Telemetry 简介
人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务
在设备的 MQTT 守护程序与 IBM MQ 之间传输消息

执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成

MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合

MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

MQTT 客户机和 IBM MQ 应用程序以两种方式松散耦合:

1. 发布者和订阅者通过出版物和预订与主题的关联进行松散耦合。 发布者和订户通常不知道发布或订阅的其他来源的地址或身份。
2. MQTT 客户机在不同的线程上发布，预订，接收发布和处理传递应答。

MQTT 客户机应用程序不会等到发布已交付。 应用程序将消息传递到 MQTT 客户机，然后应用程序在其自己的线程上继续操作。 传递令牌用于将应用程序与发布的传递同步; 请参阅 传递令牌。

将消息传递到 MQTT 客户机后，应用程序可以选择等待 delivery-token。 客户机可以提供在将发布传递到 IBM MQ 时调用的回调方法，而不是等待。 它还可以忽略 delivery-token。

根据与消息关联的服务质量，会将 delivery-token 立即返回到回调方法，或者可能在一段相当长的时间后返回。 在客户机断开连接并重新连接后，甚至可能返回 delivery-token。 如果服务质量为 *fire and* 算了，那么将立即返回 delivery-token。 在其他两种情况下，仅当客户机接收到已向订户发送发布的确认时，才会返回传递令牌。

由于客户机预订而发送到 MQTT 客户机的发布将传递到 `messageArrived` 回调方法。 `messageArrived` 在与主应用程序不同的线程上运行。

将消息直接发送到 MQTT 客户机

您可以通过两种方法之一向特定 MQTT 客户机发送消息。

1. IBM MQ 应用程序可以在没有预订的情况下直接将消息发送到 MQTT 客户机; 请参阅 直接将消息发送到客户机。
2. 另一种方法是使用 `ClientIdentifier` 命名约定。 使所有 MQTT 订户使用其唯一 `ClientIdentifier` 作为主题来创建预订。 发布到 *ClientIdentifier*。 该出版物将发送到预订了主题 *ClientIdentifier* 的客户机。 通过使用此技术，您可以将发布发送到特定 MQTT 订户。

相关概念

IBM MQ Telemetry 简介

人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例

遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议

IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务

遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道

创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

IBM MQ Telemetry 安全性

保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

SSL 主要用于加密设备与遥测通道之间的通信，并认证正在连接到正确服务器的设备; 请参阅 使用 SSL 进行遥测通道认证。 您还可以使用 SSL 来检查是否允许客户机设备连接到服务器; 请参阅 MQTT 使用 SSL 的客户机认证。

JAAS 主要用于检查是否允许设备的用户使用服务器应用程序; 请参阅 MQTT 使用密码进行客户机认证。 JAAS 可以与 LDAP 配合使用，以使用单点登录目录来检查密码。

SSL 和 JAAS 可结合使用以提供双因子认证。 您可以将 SSL 使用的密码限制为符合 FIPS 标准的密码。

拥有至少数以万计的用户，提供个人安全概要文件并不总是切实可行的。 使用概要文件来授权个别用户访问 IBM MQ 对象也并非始终可行。 而是将用户分组到用于授权发布和预订主题以及向客户机发送发布的类中。

配置每个遥测通道以将客户机映射到公共客户机用户标识。 对在特定通道上连接的每个客户机使用公共用户标识; 请参阅 MQTT 客户机身份和授权。

授权用户组不会影响每个人的认证。 每个用户都可以在客户机或服务器上使用其 **用户名** 和 **密码**进行认证，然后在服务器上使用公共用户标识进行授权。

相关概念

IBM MQ Telemetry 简介
人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

IBM MQ Telemetry 全球化

MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

应用程序负责将字母数据转换为正确的代码页和数字数据转换为正确的数字编码。

MQTT Java 客户机具有方便的 `MqttMessage.toString` 方法。 此方法将消息有效内容视为以本地平台缺省字符集 (通常为 UTF-8) 进行编码。 它将有效内容转换为 Java 字符串。 Java 具有 String 方法 `getBytes`，用于将字符串转换为使用本地平台缺省字符集编码的字节数组。 两个 MQTT Java 程序在具有相同缺省字符集的平台之间交换消息有效内容中的文本，在 UTF-8 中轻松高效地执行此操作。

如果其中一个平台的缺省字符集不是 UTF-8，那么应用程序必须建立用于交换消息的约定。 例如，发布程序使用 `getBytes("UTF8")` 方法指定从字符串到 UTF-8 的转换。 要接收消息文本，订户假定消息以 UTF-8 字符集进行编码。

遥测 (MQXR) 服务将来自 MQTT 客户机消息的所有入局发布的编码描述为 UTF-8。 它设置 `MQMD.CodedCharSetId` 到 UTF-8， `RFH2.CodedCharSetId` 到 `MQCCSI_INHERIT`；请参阅 第 245 页的『IBM MQ Telemetry 与队列管理器的集成』。 出版物的格式设置为 `MQFMT_NONE`，因此通道或 `MQGET` 无法执行任何转换。

相关概念

IBM MQ Telemetry 简介
人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

IBM MQ Telemetry 的性能和可伸缩性

在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

容量规划

有关 IBM MQ Telemetry 的性能报告的信息，请从 Web 站点 IBM MQ 系列-性能报告中选择 IBM MQ Telemetry 性能评估报告。

连接

连接所涉及的成本包括

- 在处理器使用情况和时间方面设置连接本身的成本。
- 网络成本。
- 保持连接打开但不使用连接时使用的内存。

当客户保持连接时，会产生额外的负载。 如果连接保持打开状态，那么 TCP/IP 流和 MQTT 消息将使用网络来检查该连接是否仍然存在。 此外，对于保持打开的每个客户机连接，将在服务器中使用内存。

如果每分钟发送多条消息，请保持连接处于打开状态，以避免启动新连接的成本。 如果每 10-15 分钟发送不到一条消息，请考虑断开连接以避免使其保持打开状态的成本。 您可能希望将 SSL 连接保持打开状态但处于空闲状态的时间更长，因为设置的成本更高。

此外，请考虑客户的能力。 如果客户机上有存储转发工具，那么您可以对消息进行批处理，并断开发送批处理之间的连接。 但是，如果客户机已断开连接，那么客户机无法从服务器接收消息。 因此，应用程序的用途会影响决策。

如果系统有一个客户机发送许多消息 (例如文件传输) ，请不要等待每条消息的服务器响应。 而是发送所有消息并在结束时检查是否已接收到所有消息。 或者，使用 服务质量 (QoS)。

您可以根据消息来改变 QoS ，使用 QoS 0 来交付不重要的消息，并使用 QoS 2 来交付重要的消息。 如果 QoS 为 0 ，那么消息吞吐量可能是 QoS 为 2 的两倍左右。

命名约定

如果要为许多客户机设计应用程序，请实施有效的命名约定。 为了将每个客户机映射到正确的 `ClientIdentifier`，请使 `ClientIdentifier` 有意义。 良好的命名约定使管理员能够更轻松地确定哪些客户机正在运行。 命名约定可帮助管理员在 IBM MQ Explorer 中过滤一长串客户机，并帮助确定问题; 请参阅 客户机标识。

吞吐量

主题名称的长度会影响流经网络的字节数。 发布或预订时，消息中的字节数可能很重要。 因此，限制主题名称中的字符数。 当 MQTT 客户机预订主题 IBM MQ 时，为其提供格式的名称:

```
ClientIdentifier: TopicName
```

要查看 MQTT 客户机的所有预订，可以使用 IBM MQ MQSC **DISPLAY** 命令:

```
DISPLAY SUB(' ClientID1:*')
```

在 IBM MQ 中定义供 MQTT 客户机使用的资源

MQTT 客户机连接到 IBM MQ 远程队列管理器。 IBM MQ 应用程序有两种基本方法可将消息发送到 MQTT 客户机: 将缺省传输队列设置为 SYSTEM.MQTT.TRANSMIT.QUEUE 或使用队列管理器别名。 如果存在大量 MQTT 客户机，请定义队列管理器的缺省传输队列。 使用缺省传输队列设置可简化管理工作; 请参阅 配置分布式排队以向 MQTT 客户机发送消息。

通过避免预订来提高可伸缩性。

当 MQTT V3 客户机预订主题时，将通过 IBM MQ 中的遥测 (MQXR) 服务创建预订。 预订将客户机的发布路由到 SYSTEM.MQTT.TRANSMIT.QUEUE。 每个发布的传输头中的远程队列管理器名称都设置为进行预订的 MQTT 客户机的 ClientIdentifier 。 如果有许多客户机 (每个客户机都有自己的预订) ，那么这将导致在整个 IBM MQ 发布/预订集群或层次结构中维护许多代理预订。 有关不使用发布/预订，而是改为使用基于点到点的解决方案的信息，请参阅 直接向客户机发送消息。

管理大量客户机

要支持许多并发连接的客户机，请通过设置 JVM 参数 **-Xms** 和 **-Xmx** 来增加可用于遥测 (MQXR) 服务的内存。 请按照以下步骤操作:

1. 在遥测服务配置目录中找到 java.properties 文件; 请参阅 Windows 上的遥测 (MQXR) 服务配置目录 或 Linux 上的遥测服务配置目录 。
2. 遵循文件中的指示; 1 GB 的堆足以用于 50,000 个并发连接的客户机。

   ```
   # Heap sizing options - uncomment the following lines to set the heap to 1G
   #-Xmx1024m
   #-Xms1024m
   ```

3. 添加其他命令行参数以传递到 java.properties 文件中运行遥测 (MQXR) 服务的 JVM; 请参阅 将 JVM 参数传递到遥测 (MQXR) 服务。

要增加 Linux 上的打开文件描述符数，请将以下行添加到 /etc/security/limits.conf/，然后重新登录。

```
@mqm soft nofile 65000
@mqm hard nofile 65000
```

每个套接字都需要一个文件描述符。 遥测服务需要一些额外的文件描述符，因此此数目必须大于所需的打开套接字数。

队列管理器对每个非持久预订使用对象句柄。 要支持许多活动的非持久预订，请增加队列管理器中的最大活动句柄数; 例如:

```
echo ALTER QMGR MAXHANDS(999999999) | runmqsc qMgrName
```

图 *50:* 更改 *Windows* 上的最大句柄数

```
echo "ALTER QMGR MAXHANDS(999999999)" | runmqsc qMgrName
```

图 *51:* 更改 *Linux* 上的最大句柄数

其他注意事项

在规划系统需求时，请考虑重新启动系统所花费的时间长度。 计划的停机时间可能会影响排队等待处理的消息数。 配置系统，以便可以在可接受的时间内成功处理消息。 查看磁盘存储，内存和处理能力。 对于某些客户机应用程序，可能在客户机重新连接时废弃消息。 要废弃消息，请在客户机连接参数中设置 `CleanSession`；请参阅 清除会话。 或者，在 MQTT 客户机中使用尽力而为的服务质量 `0` 进行发布和预订；请参阅 服务质量。 从 IBM MQ 发送消息时使用 **非持久** 消息。 当系统或连接重新启动时，不会恢复具有这些服务质量的消息。

相关概念

IBM MQ Telemetry 简介
人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 支持的设备
MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

IBM MQ Telemetry 支持的设备

MQTT 客户机可以在从传感器和执行器到手持设备和车辆系统等一系列设备上运行。

MQTT 客户机很小，并且在受小内存和低处理能力限制的设备上运行。 MQTT protocol 可靠且具有较小的头，这适用于受低带宽，高成本和间歇性可用性约束的网络。

IBM MQ Telemetry 通过 MQTT 客户机应用程序与遥测设备通信。 这些应用程序使用以下资源，所有这些资源都实现 MQTT v3 协议:

- 以下客户机库:
  - *MQTT client for Java*，用于为 (例如) Android， OS X， Linux 或 Windows 设备构建本机应用程序。 使用此客户机库的应用程序可以在 Java 的所有变体上运行，从最小 CLDC (已连接的受限设备配置) /MIDP (移动信息设备概要文件) 到 CDC (已连接的设备配置) /Foundation， J2SE (Java Platform， Standard Edition) 和 J2EE (Java Platform， Enterprise Edition)。 还支持 IBM jclRM 定制类库。 Java ME 平台通常用于小型设备，如执行器，传感器，手机和其他嵌入式设备。 Java SE 平台通常安装在更高端的嵌入式设备上，例如台式机和服务器。
  - *MQTT client for Java*，用于为 (例如) iOS， OS X， Linux 或 Windows 设备构建本机应用程序。 此客户机库提供 C 参考实现以及针对 Windows 和 Linux 系统的预构建本机客户机。 C 参考实现使 MQTT 能够移植到各种设备和平台。 Intel 上的某些 Windows 系统 (包括 Windows 7， RedHat， Ubuntu 以及 ARM 平台 (例如 Eurotech Viper) 上的某些 Linux 系统) 实现运行 C 客户机的 Linux 版本，但 IBM 不提供对这些平台的服务支持。 如果您打算致电 IBM 支持中心，那么必须在受支持的平台上重现客户机问题。
  - *MQTT client for Java*，用于构建基于浏览器的 Web 应用程序。

MQTT 客户机库可从 Eclipse Paho 和 MQTT.org 免费获取。 请参阅 IBM MQ Telemetry Transport 样本程序。

相关概念

IBM MQ Telemetry 简介
人们，企业和政府越来越希望使用 IBM MQ Telemetry 与我们生活和工作的环境进行更智能的交互。 IBM MQ Telemetry 将各种设备连接到因特网和企业，并降低为智能设备构建应用程序的成本。

遥测用例
遥测是对数据的自动感应和度量以及对远程设备的自动控制。 重点在于数据从设备到中央控制点的传输。 遥测还包含向设备发送配置和控制信息。

将遥测设备连接至队列管理器

遥测连接协议
IBM MQ Telemetry 支持 TCP/IP IPv4 和 IPv6 以及 SSL。

遥测 (MQXR) 服务
遥测 (MQXR) 服务是作为 IBM MQ 服务管理的 TCP/IP 侦听器。 使用 MQ Explorer 向导或 **runmqsc** 命令创建服务。

遥测通道
创建遥测通道以创建具有不同属性 (例如 Java 认证和授权服务 (JAAS) 或 SSL 认证) 的连接，或者管理客户机组。

MQTT protocol
IBM MQ Telemetry Transport (MQTT) v3 协议旨在以低带宽或昂贵的连接在小型设备之间交换消息，并可靠地发送消息。 它使用 TCP/IP。

MQTT Client
MQTT 客户机应用程序负责从遥测设备收集信息，连接到服务器以及将信息发布到服务器。 它还可以预订主题，接收出版物以及控制遥测设备。

向 MQTT 客户机发送消息
IBM MQ 应用程序可以通过发布到客户机创建的预订或通过直接发送消息来发送 MQTT v3 客户机消息。 MQTT 客户机可以通过发布到其他客户机预订的主题来相互发送消息。

从 MQTT 客户机向 IBM MQ 应用程序发送消息
IBM MQ 应用程序可以通过预订主题从 MQTT v3 客户机接收消息。 MQTT 客户机使用遥测通道连接到 IBM MQ ，并通过发布到同一主题将消息发送到 IBM MQ 应用程序。

MQTT 发布/预订应用程序
使用基于主题的发布/预订来编写 MQTT 应用程序。

遥测应用程序
使用 IBM MQ 或 IBM Integration Bus 消息流编写遥测应用程序。

设备的 MQTT 守护程序
设备的 IBM MQ MQTT 守护程序是高级 MQTT V3 客户机应用程序。 用于存储和转发来自其他 MQTT 客户机的消息。 它像 MQTT 客户机一样连接到 IBM MQ ，但您也可以将其他 MQTT 客户机连接到该客户机。 您还可以将其连接到其他 MQTT 守护程序。

MQTT 无状态和有状态会话
MQTT 客户机可以创建与队列管理器的有状态会话。 当有状态 MQTT 客户机断开连接时，队列管理器会维护客户机创建的预订以及未完成的消息。 当客户机重新连接时，它将解析正在使用的消息。 它会发送排队等候传递的所有消息，并接收断开连接时针对预订发布的所有消息。

未连接 MQTT 客户机时
当客户机未连接时，队列管理器可以继续代表它接收发布。 它们将在客户机重新连接时转发给客户机。 如果客户机意外断开连接，那么客户机可以创建队列管理器代表客户机发布的 “Last will and testament”。

MQTT 客户机与 IBM MQ 应用程序之间的松耦合
MQTT 客户机与 IBM MQ 应用程序之间的发布流是松散耦合的。 出版物可能源自 MQTT 客户机或 IBM MQ 应用程序，并且没有设置顺序。 出版商和订阅者之间存在松散的联系。 它们通过出版物和订阅进行间接互动。 您还可以从 IBM MQ 应用程序直接将消息发送到 MQTT 客户机。

IBM MQ Telemetry 安全性
保护遥测设备的安全很重要，因为这些设备可能可移动，并且用于无法严格控制的场所。 您可以使用 VPN 来保护从 MQTT 设备到遥测 (MQXR) 服务的连接。 IBM MQ Telemetry 提供了另外两种安全性机制: SSL 和 JAAS。

IBM MQ Telemetry 全球化
MQTT v3 协议中的消息有效内容将编码为字节数组。 通常，处理文本的应用程序会在 UTF-8 中创建消息有效内容。 遥测通道将消息有效内容描述为 UTF-8，但不执行任何代码页转换。 发布主题字符串必须为 UTF-8。

IBM MQ Telemetry 的性能和可伸缩性
在管理大量客户机和提高 IBM MQ Telemetry 的可伸缩性时，请考虑以下因素。

相关任务

在设备的 MQTT 守护程序与 IBM MQ 之间传输消息
执行此任务以了解如何将命令发送到设备的 MQTT 守护程序。 您编写的命令将创建一个网桥，用于将消息从 IBM MQ 传输到守护程序，并将消息从守护程序传输到 IBM MQ。

相关参考

IBM MQ Telemetry 与队列管理器的集成
MQTT 客户机作为发布/预订应用程序与 IBM MQ 集成。 它可以发布或预订 IBM MQ 中的主题，创建新主题或使用现有主题。 由于 MQTT 客户机 (包括其自身) 或其他发布到其预订主题的 IBM MQ 应用程序，它从 IBM MQ 接收发布。 将应用规则来决定发布的属性。

安全性

在 IBM MQ 中，提供安全性的方法有几种: 授权服务接口; 用户编写的或第三方的通道出口; 使用安全套接字层 (SSL) 的通道安全性，通道认证记录 和消息安全性。

授权服务接口

对象权限管理器 (OAM) 提供了使用 MQI 调用，命令和访问对象的授权，缺省情况下已启用此授权。 通过 IBM MQ 用户组和 OAM 控制对 IBM MQ 实体的访问。 管理员可以根据需要使用命令行界面来授予或撤销权限。

有关创建授权服务组件的更多信息，请参阅 在 Windows UNIX 和 Linux 系统上设置安全性。

用户编写的通道出口或第三方通道出口

通道可以使用用户编写的通道出口或第三方通道出口。 有关更多信息，请参阅 消息传递通道的通道出口程序。

使用 SSL 的通道安全性

安全套接字层 (SSL) 协议提供业界标准的通道安全性，防止窃听，篡改和冒充。

SSL 使用公用密钥和对称方法来提供消息机密性和完整性以及相互认证。

有关 IBM MQ 中安全性的全面复审，包括有关 SSL 的详细信息，请参阅 安全性。 有关 SSL 的概述，包括本节中描述的命令的指针，请参阅 加密安全协议: SSL 和 TLS。

通道认证记录

使用通道认证记录对授予在通道级别连接系统的访问权进行精确控制。 有关更多信息，请参阅 通道认证记录。

消息安全性

使用 IBM MQ Advanced Message Security(这是 IBM MQ 的单独安装和许可组件) 对使用 IBM MQ 发送和接收的消息提供加密保护。 请参阅 IBM MQ Advanced Message Security。

相关信息

安全性
规划安全需求

客户机和服务器

介绍 IBM MQ 如何支持其应用程序的客户机/服务器配置。

IBM MQ MQI 客户机 是一个组件，允许在系统上运行的应用程序向在另一个系统上运行的队列管理器发出 MQI 调用。 来自调用的输出将发送回客户机，客户机会将其传递回应用程序。

IBM MQ 服务器 是向一个或多个客户机提供排队服务的队列管理器。 所有 IBM MQ 对象 (例如，队列) 仅存在于队列管理器机器 (IBM MQ 服务器机器) 上，而不存在于客户机上。 IBM MQ 服务器还可以支持本地 IBM MQ 应用程序。

IBM MQ 服务器与普通队列管理器之间的区别在于，服务器与每个客户机都有专用通信链路。 有关为客户机和服务器创建通道的更多信息，请参阅 配置分布式排队。

有关一般客户机的信息，请参阅 第 265 页的『IBM MQ MQI clients 概述』。

客户机/服务器环境中的 IBM MQ 应用程序

当链接到服务器时，客户机 IBM MQ 应用程序可以采用与本地应用程序相同的方式发出大多数 MQI 调用。 客户机应用程序发出 MQCONN 调用以连接到指定的队列管理器。 然后，此队列管理器将处理指定从连接请求返回的连接句柄的任何其他 MQI 调用。

您必须将应用程序链接到相应的客户机库。 请参阅 为 IBM MQ MQI clients 构建应用程序。

相关概念

第 270 页的『事务管理和支持』
事务管理简介以及 IBM MQ 如何支持事务。

第 271 页的『扩展队列管理器设施』
您可以使用用户出口， API 出口或可安装服务来扩展队列管理器设施。

IBM MQ MQI clients 概述

IBM MQ MQI client 是 IBM MQ 产品的组件，可以安装在没有运行队列管理器的系统上。

通过使用 IBM MQ MQI client，在与客户机相同的系统上运行的应用程序可以连接到在另一个系统上运行的队列管理器。 应用程序可以向该队列管理器发出 MQI 调用。 此类应用程序称为 IBM MQ MQI client 应用程序，队列管理器称为 服务器队列管理器。

IBM MQ MQI client 应用程序和服务器队列管理器使用 *MQI* 通道相互通信。 MQI 通道在客户机应用程序发出 **MQCONN** 或 **MQCONNX** 调用以连接到队列管理器时启动，并在客户机应用程序发出 **MQDISC** 调用以与队列管理器断开连接时结束。 MQI 调用流在 MQI 通道上的一个方向上的输入参数和输出参数在相反方向上的输入参数。

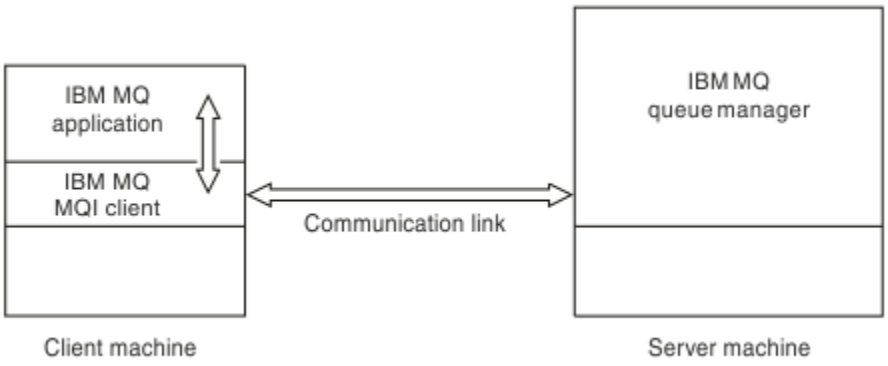


图 *52:* 客户机与服务器之间的链接

可以使用以下平台。 这些组合取决于您正在使用的 IBM MQ 产品，并在 第 267 页的『针对 IBM MQ 客户机的平台支持』中进行了描述。

IBM MQ MQI client	IBM MQ 服务器
UNIX 和 Linux Windows IBM i HP Integrity NonStop Server	UNIX 和 Linux Windows IBM i z/OS HP Integrity NonStop Server

MQI 可用于在客户机平台上运行的应用程序; 队列和其他 IBM MQ 对象保存在已安装在服务器上的队列管理器上。

要在 IBM MQ MQI client 环境中运行的应用程序必须首先与相关客户机库链接。 当应用程序发出 MQI 调用时， IBM MQ MQI client 会将请求定向到队列管理器，在该队列管理器中处理该请求，并从该队列管理器将应答发送回 IBM MQ MQI client。

应用程序与 IBM MQ MQI client 之间的链接是在运行时动态建立的。

您还可以使用 IBM MQ classes for .NET， IBM MQ classes for Java 或 IBM MQ classes for Java Message Service (JMS) 来开发客户机应用程序。 您可以在 IBM i IBM i， UNIX， Linux 和 Windows 平台上使用 Java 和 JMS 客户机。 此处未描述 Java 和 JMS 的使用。 有关如何安装，配置和使用 IBM MQ classes for Java 和 IBM MQ classes for JMS 的完整详细信息，请参阅 使用 IBM MQ classes for Java 和 使用 IBM MQ classes for JMS。

相关概念

第 266 页的『为何使用 IBM MQ 客户机?』
使用 IBM MQ 客户机是实现 IBM MQ 消息传递和排队的有效方法。

第 267 页的『如何设置 IBM MQ MQI client?』
遵循以下指示信息来设置客户机。

第 268 页的『什么是扩展事务客户机?』
IBM MQ 扩展事务客户机可以在外部事务管理器的控制下更新由另一个资源管理器管理的资源。

第 269 页的『客户机如何连接到服务器』
客户机使用 MQCONN 或 MQCONNX 连接到服务器，并通过通道进行通信。

为何使用 IBM MQ 客户机?

使用 IBM MQ 客户机是实现 IBM MQ 消息传递和排队的有效方法。

您可以使用在一台机器上运行的 MQI 和在另一台机器 (物理或虚拟) 上运行的队列管理器的应用程序。 这样做的好处是:

- 客户机上不需要完整的 IBM MQ 实现。
- 降低了客户机系统上的硬件需求。
- 减少了系统管理需求。
- 在客户机上运行的 IBM MQ 应用程序可以连接到不同系统上的多个队列管理器。
- 可以使用使用不同传输协议的备用通道。

相关参考

第 266 页的『哪些应用程序在 IBM MQ MQI client 上运行?』
客户机环境中支持完整 MQI。

第 267 页的『针对 IBM MQ 客户机的平台支持』
所有服务器平台上的 IBM MQ 都接受来自 IBM i IBM i， UNIX 或 Linux 系统和 Windows 上的 IBM MQ MQI clients 的客户机连接。

哪些应用程序在 *IBM MQ MQI client* 上运行?

客户机环境中支持完整 MQI。

这使几乎任何 IBM MQ 应用程序都能够配置为通过将 IBM MQ MQI client 上的应用程序链接到 MQIC 库而不是 MQI 库来在 IBM MQ MQI client 系统上运行。 例外情况如下所示：

- 带有信号的 MQGET
- 需要与其他资源管理器进行同步点协调的应用程序必须使用扩展事务客户机

如果启用预读，那么为了提高非持久消息传递性能，并非所有 MQGET 选项都可用。 该表显示了允许的选项以及是否可以在 MQGET 调用之间更改这些选项。

表 19: 启用预读时允许的 *MQGET* 选项

	在启用预读时允许，并且可以在 MQGET 调用之间进行更改	启用预读时允许，但不能在 MQGET 调用之间更改 [1]	启用预读时不允许使用的 MQGET 选项 [2]
MQGET MD 值	MsgId [3] CorrelId [3]	编码 CodedCharSetId	
MQGET MQGMO 选项	MQGMO_WAIT MQGMO_NO_WAIT MQGMO_FAIL_IF_QUIESCING MQGMO_BROWSE_FIRST [4] MQGMO_BROWSE_NEXT [4] MQGMO_BROWSE_MESSAGE _UNDER_CURSOR [4]	MQGMO_SYNCPOINT_IF _PERSISTENT MQGMO_NO_SYNCPOINT MQGMO_ACCEPT_TRUNCATED_MSG MQGMO_CONVERT MQGMO_LOGICAL_ORDER MQGMO_COMPLETE_MSG MQGMO_ALL_MSGS_AVAILABLE MQGMO_ALL_SEGMENTS_ AVAILABLE MQGMO_MARK_BROWSE_HANDLE MQGMO_MARK_BROWSE_CO_OP MQGMO_UNMARK_BROWSE_CO_OP MQGMO_UNMARK_BROWSE_HANDLE MQGMO_UNMARKED_BROWSE_MSG MQGMO_PROPERTIES_FORCE_MQRFH2 MQGMO_NO_PROPERTIES MQGMO_PROPERTIES_IN_HANDLE MQGMO_PROPERTIES_COMPATIBILITY	MQGMO_SET_SIGNAL MQGMO_SYNCPOINT MQGMO_MARK_SKIP 备份 (_BACKOUT) MQGMO_MSG_UNDER _CURSOR [4] MQGMO_LOCK MQGMO_UNLOCK
MQGMO 值		MsgHandle	

1. 如果在 MQGET 调用之间更改了这些选项，那么将返回 MQRC_OPTIONS_CHANGED 原因码。
2. 如果在第一个 MQGET 调用上指定这些选项，那么将禁用预读。 如果在后续 MQGET 调用上指定这些选项，那么将返回原因码 MQRC_OPTIONS_ERROR。
3. 客户机应用程序需要意识到，如果 MsgId 和 CorrelId 值在 MQGET 调用之间发生更改，那么具有先前值的消息可能已发送至客户机，并保留在客户机预读缓冲区中，直至被使用（或自动清除）为止。
4. 第一个 MQGET 调用确定在启用了预读时是否要从队列中浏览或获取消息。 如果应用程序尝试同时执行浏览与获取操作，将返回原因码 MQRC_OPTIONS_CHANGED。
5. MQGMO_MSG_UNDER_CURSOR 不能与预读配合使用。 在启用了预读时，可浏览或获取消息，但是不能同时执行这两个操作。

在 IBM MQ MQI client 上运行的应用程序可以同时连接到多个队列管理器，或者在 MQCONN 或 MQCONNX 调用上使用带有星号 (*) 的队列管理器名称 (请参阅 将 IBM MQ MQI client 应用程序连接到队列管理器 中的示例)。

针对 *IBM MQ* 客户机的平台支持

所有服务器平台上的 IBM MQ 都接受来自 IBM i IBM i， UNIX 或 Linux 系统和 Windows 上的 IBM MQ MQI clients 的客户机连接。

作为 基本产品和服务器 安装的 IBM MQ 可在以下平台上接受来自 IBM MQ MQI clients 的连接：

- HP Integrity NonStop Server
- IBM i IBM i
- UNIX 和 Linux 系统
- Windows

客户机连接在编码字符集标识 (CCSID) 和通信协议方面存在差异。

如何设置 IBM MQ MQI client?

遵循以下指示信息来设置客户机。

要设置 IBM MQ MQI client ， 必须已安装并运行 IBM MQ 服务器，您的客户机将连接到该服务器。 设置客户机所涉及的步骤包括：

1. 检查您是否有适合 IBM MQ MQI client 的平台，以及硬件和软件是否满足需求。 第 267 页的『针对 IBM MQ 客户机的平台支持』中描述了平台支持。
2. 决定如何在客户机工作站上安装 IBM MQ，然后遵循特定客户机和服务器平台组合的指示信息。 安装 IBM MQ 客户机中描述了安装。
3. 确保已配置并连接通信链路。 配置服务器与客户机之间的连接中描述了通信链路的配置。
4. 检查您的安装是否正常工作。 验证客户机安装中描述了如何验证安装。
5. 当您已验证 IBM MQ MQI client 安装时，请考虑是否必须保护客户机。 设置 IBM MQ MQI client 安全性中描述了客户机安全性。
6. 设置要在客户机上运行的 IBM MQ 应用程序所需的 IBM MQ MQI client 与服务器之间的通道。 定义 MQI 通道中描述了如何设置通道。 如果您使用的是 SSL，那么还有一些其他注意事项。 指定 MQI 通道使用 SSL 中描述了这些注意事项。 您可能需要使用 IBM MQ MQI client 配置文件或 IBM MQ 环境变量来设置通道。 IBM MQ 环境变量在 使用 IBM MQ 环境变量中进行了描述。
7. IBM MQ 应用程序在 开发应用程序中进行了完整描述。
8. 在 IBM MQ MQI client 环境中设计，构建和运行应用程序时，需要考虑队列管理器环境的一些差异。 有关这些差异的信息，请参阅:
   - 在客户机应用程序中使用消息队列接口 (MQI)
   - 为 IBM MQ MQI clients 构建应用程序
   - 将 IBM MQ MQI client 应用程序连接到队列管理器
   - 解决 IBM MQ MQI clients 的问题

什么是扩展事务客户机?

IBM MQ 扩展事务客户机可以在外部事务管理器的控制下更新由另一个资源管理器管理的资源。

如果您不熟悉事务管理的概念，请参阅 第 270 页的『事务管理和支持』。

请注意， XA 事务客户机现在作为 IBM MQ 的一部分提供。

客户机应用程序可以参与由它所连接的队列管理器管理的工作单元。 在工作单元中，客户机应用程序可以将消息放入该队列管理器所拥有的队列，并从中获取消息。 然后，客户机应用程序可以使用 **MQCMIT** 调用来落实工作单元，或者使用 **MQBACK** 调用来回退工作单元。 但是，在同一工作单元中，客户机应用程序无法更新其他资源管理器的资源，例如 Db2 数据库的表。 使用 IBM MQ 扩展事务客户机将除去此限制。

IBM MQ 扩展事务客户机是具有一些其他功能的 IBM MQ MQI client 。 使用此功能，客户机应用程序可以在同一工作单元中执行以下任务:

- 将消息放入它所连接的队列管理器所拥有的队列，并从中获取消息
- 更新 IBM MQ 队列管理器以外的资源管理器的资源

此工作单元必须由与客户机应用程序在同一系统上运行的外部事务管理器管理。 工作单元不能由客户机应用程序所连接到的队列管理器管理。 这意味着队列管理器只能充当资源管理器，而不能充当事务管理器。 这还意味着客户机应用程序只能使用外部事务管理器提供的应用程序编程接口 (API) 来落实或回退工作单元。 因此，客户机应用程序无法使用 MQI 调用 **MQBEGIN**， **MQCMIT** 和 **MQBACK**。

外部事务管理器使用连接到队列管理器的客户机应用程序所使用的相同 MQI 通道作为资源管理器与队列管理器进行通信。 但是，在发生故障后的恢复情况中，当没有应用程序在运行时，事务管理器可以使用专用 MQI 通道来恢复发生故障时队列管理器正在参与的任何不完整工作单元。

在此部分中，不具有扩展事务功能的 IBM MQ MQI client 称为 IBM MQ 基本客户机。 因此，您可以考虑 IBM MQ 扩展事务客户机由 IBM MQ 基本客户机组成，并添加扩展事务功能。

注: IBM i IBM i 上的 IBM MQ MQI client 不支持 IBM MQ 扩展事务功能。

相关参考

第 269 页的『针对扩展事务客户机的平台支持』
扩展事务客户机可用于支持基本客户机的所有分布式平台和 IBM i 平台。 客户机不可用于 z/OS。

针对扩展事务客户机的平台支持

扩展事务客户机可用于支持基本客户机的所有分布式平台和 IBM i 平台。 客户机不可用于 z/OS。

使用扩展事务客户机的客户机应用程序只能连接到以下 IBM MQ 8.0 产品的队列管理器:

- IBM MQ for AIX
- IBM MQ for HP-UX
- IBM WebSphere MQ for HP Integrity NonStop Server
- IBM MQ for IBM i
- IBM MQ for Linux
- IBM MQ for Solaris
- IBM MQ for Windows

z/OS 虽然没有在 z/OS 上运行的扩展事务客户机，但是使用扩展事务客户机的客户机应用程序可以连接到在 z/OS 上运行的队列管理器。

对于每个平台，扩展事务客户机的硬件和软件需求与 IBM MQ 基本客户机的硬件和软件需求相同。 扩展事务客户机支持编程语言 (如果它受 IBM MQ 基本客户机和您正在使用的事务管理器支持)。

有关所有平台 (HP Integrity NonStop Server 除外) 的外部事务管理器的信息，请参阅 System Requirements for IBM MQ 8.0 Web 页面，并遵循相应平台的软件产品兼容性报告 (SPCR) 工具链接。

对于 HP Integrity NonStop Server，请参阅 在 HP Integrity NonStop Server 上规划 IBM MQ 客户机环境 。

客户机如何连接到服务器

客户机使用 MQCONN 或 MQCONNX 连接到服务器，并通过通道进行通信。

在 IBM MQ 客户机环境中运行的应用程序必须保持客户机与服务器之间的活动连接。

连接由发出 MQCONN 或 MQCONNX 调用的应用程序进行。 客户机和服务器通过 *MQI* 通道进行通信，或者在使用共享对话时，每个对话共享一个 MQI 通道实例。 当调用成功时， MQI 通道实例或对话将保持连接，直到应用程序发出 MQDISC 调用为止。 这是应用程序需要连接到的每个队列管理器的情况。

相关概念

第 269 页的『同一机器上的客户机和队列管理器』
当您的机器还安装了队列管理器时，还可以在 IBM MQ MQI client 环境中运行应用程序。

第 269 页的『不同平台上的客户』
以下是 IBM MQ MQI client 和服务器系统的另一个示例。 在此示例中，服务器在不同平台上与三个 IBM MQ MQI clients 通信。

第 270 页的『使用不同版本的客户机和服务器软件』
如果您正在使用先前版本的 IBM MQ 产品，请确保服务器支持从客户机的 CCSID 进行代码转换。

同一机器上的客户机和队列管理器

当您的机器还安装了队列管理器时，还可以在 IBM MQ MQI client 环境中运行应用程序。

在这种情况下，您可以选择链接到队列管理器库或客户机库，但请记住，如果链接到客户机库，那么仍需要定义通道连接。 这在应用程序的开发阶段很有用。 您可以在不依赖他人的情况下在自己的机器上测试程序，并确信将其移至独立 IBM MQ MQI client 环境时仍有效。

不同平台上的客户

以下是 IBM MQ MQI client 和服务器系统的另一个示例。 在此示例中，服务器在不同平台上与三个 IBM MQ MQI clients 通信。

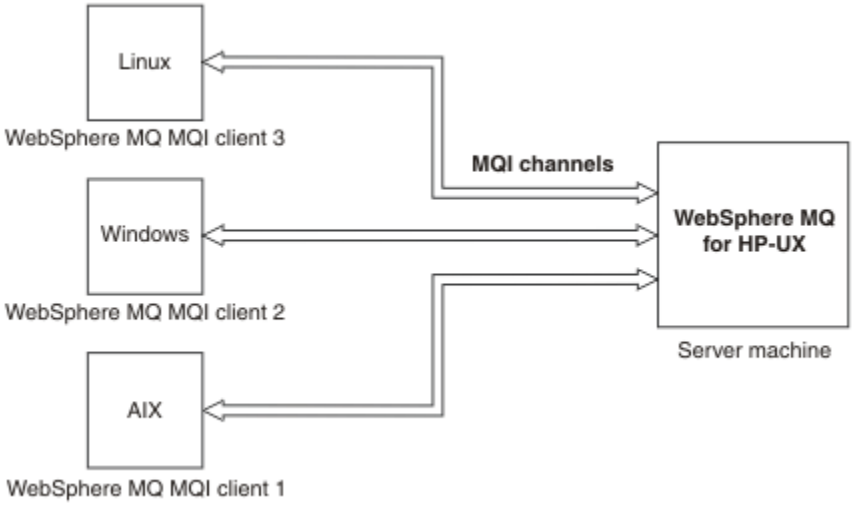


图 *53: IBM MQ* 服务器连接到不同平台上的客户机

其他更复杂的环境是可能的。 例如， IBM MQ 客户机可以连接到多个队列管理器，也可以连接到作为队列共享组一部分的任意数量的队列管理器。

使用不同版本的客户机和服务器软件

如果您正在使用先前版本的 IBM MQ 产品，请确保服务器支持从客户机的 CCSID 进行代码转换。

IBM WebSphere MQ 7.0 或更高版本的客户机可以连接到所有受支持的队列管理器版本。 如果要连接到较低版本的队列管理器，那么不能在客户机上的 IBM MQ 应用程序中使用来自较高版本的产品的功能部件和结构。

IBM WebSphere MQ 7.0 或更高版本的队列管理器接受来自所有受支持的客户机版本的连接。

有关更多信息，请参阅 开发应用程序 中支持的编程语言。

事务管理和支持

事务管理简介以及 IBM MQ 如何支持事务。

资源管理器 是一个计算机子系统，它拥有和管理可由应用程序访问和更新的资源。 以下是资源管理器的示例:

- IBM MQ 队列管理器，其资源是其队列
- Db2 数据库及其表中包含的资源

当应用程序更新一个或多个资源管理器的资源时，可能需要确保某些更新作为一个组成功完成，或者没有任何更新完成。 这种要求的原因是，如果其中一些更新成功完成，但其他更新未成功完成，那么业务数据将处于不一致状态。

以这种方式管理的资源的更新据说发生在 工作单元或 事务中。 应用程序可以将一组更新分组到工作单元中。

在工作单元期间，应用程序向资源管理器发出请求以更新其资源。 当应用程序发出落实所有更新的请求时，工作单元结束。 在落实更新之前，所有更新都不会对访问相同资源的其他应用程序可见。 或者，如果应用程序决定由于任何原因而无法完成工作单元，那么它可以发出请求，以回退其请求的所有更新，直至该点为止。 在这种情况下，任何更新都不会对其他应用程序可见。 这些更新通常在逻辑上相关，并且必须全部成功才能保留数据完整性。 如果一个更新成功，而另一个更新失败，那么数据完整性将丢失。

当工作单元成功完成时，会将其指定为 *commit*。 一旦落实，在该工作单元内进行的所有更新都将永久且不可逆。 但是，如果工作单元失败，那么将改为 回退所有更新。 此过程称为 同步点协调，在此过程中，将落实工作单元或以完整性回退工作单元。

落实或回退工作单元中的所有更新的时间点称为 同步点。 工作单元内的更新据说 在同步点控制内发生。 如果应用程序请求 在同步点控制之外的更新，那么资源管理器会立即落实该更新，即使存在正在进行的工作单元也是如此，并且以后无法回退该更新。

管理工作单元的计算机子系统称为 事务管理器或 点协调程序。

本地工作单元是其中更新的资源仅限于 IBM MQ 队列管理器资源的工作单元。 此处同步点协调由队列管理器本身使用单阶段落实过程提供。

全局 工作单元是其中属于其他资源管理器 (例如，符合 XA 的数据库) 的资源也会更新的工作单元。 在此，必须使用两阶段落实过程，并且工作单元可以由队列管理器本身进行协调，也可以由另一个符合 XA 的事务管理器 (例如 IBM TXSeries 或 BEA Tuxedo) 在外部进行协调。

事务管理器负责确保对工作单元中资源的所有更新都成功完成，或者没有任何更新完成。 应用程序向事务管理器发出落实或回退工作单元的请求。 事务管理器的示例为 CICS 和 WebSphere Application Server，尽管这两个事务管理器都具有其他功能。

某些资源管理器提供自己的事务管理功能。 例如， IBM MQ 队列管理器可以管理涉及对其自己的资源的更新和对 Db2 表的更新的工作单元。 队列管理器不需要单独的事务管理器来执行此功能，尽管如果是用户需求，可以使用此功能。 如果使用单独的事务管理器，那么会将其称为 外部事务管理器。

要让外部事务管理器管理工作单元，事务管理器与参与工作单元的每个资源管理器之间必须有标准接口。 此接口允许事务管理器和资源管理器相互通信。 其中一个接口是 *XA* 接口，它是许多事务管理器和资源管理器支持的标准接口。 XA 接口由开放式组在 分布式事务处理: *XA* 规范中发布。

当多个资源管理器参与工作单元时，事务管理器必须使用 两阶段落实 协议来确保工作单元中的所有更新都成功完成或都未完成，即使发生系统故障也是如此。 当应用程序向事务管理器发出请求以落实工作单元时，事务管理器将执行以下操作:

阶段 1 (准备落实)
事务管理器要求参与工作单元的每个资源管理器确保有关其资源的预期更新的所有信息都处于可恢复状态。 资源管理器通常通过将信息写入日志并确保将信息写入硬盘来执行此操作。 当事务管理器从每个资源管理器接收到有关其资源的预期更新的信息处于可恢复状态的通知时，阶段 1 将完成。

阶段 2 (落实)
当阶段 1 完成时，事务管理器将做出不可撤销的决策来落实工作单元。 它要求参与工作单元的每个资源管理器落实对其资源的更新。 当资源管理器接收到此请求时，它必须落实更新。 在此阶段，它无法选择将它们退出。 当事务管理器从每个资源管理器接收到其已将更新落实到其资源的通知时，阶段 2 完成。

XA 接口使用两阶段落实协议。

有关更多信息，请参阅 事务支持方案。

IBM MQ 还提供对 Microsoft Transaction Server (COM +) 的支持。 使用 Microsoft Transaction Server (COM +) 提供有关如何设置 IBM MQ 以利用 COM + 支持的信息。

扩展队列管理器设施

您可以使用用户出口， API 出口或可安装服务来扩展队列管理器设施。

用户出口

用户出口提供了一种机制，供您将自己的代码插入到队列管理器函数中。 支持的用户出口包括:

通道出口
这些出口会改变通道的运行方式。 通道出口-消息传递通道的出口程序中描述了通道出口。

数据转换出口
这些出口创建可放入应用程序中的源代码片段，以将数据从一种格式转换为另一种格式。 编写数据转换出口中描述了数据转换出口。

集群工作负载出口
此出口执行的函数由该出口的提供程序定义。 调用定义信息在 MQ_CLUSTER_WORKLOAD_EXIT-调用描述中提供。

API 出口

通过 API 出口，可以编写用于更改 IBM MQ API 调用（如 MQPUT 和 MQGET）行为的代码，然后直接在这些调用之前或之后插入该代码。 插入是自动的; 队列管理器在已注册的点处驱动退出代码。 有关 API 出口的更多信息，请参阅 使用和编写 API 出口。

可安装服务

可安装服务具有具有多个入口点的正规化接口 (API)。

可安装服务的实现称为 服务组件。 您可以使用 IBM MQ 随附的组件，也可以编写自己的组件以执行所需的功能。

目前，提供了以下可安装服务:

授权服务
授权服务允许您构建自己的安全设施。

实现服务的缺省服务组件是对象权限管理器 (OAM)。 缺省情况下， OAM 处于活动状态，您不必执行任何操作来对其进行配置。 您可以使用授权服务接口来创建其他组件以替换或扩充 OAM。 有关 OAM 的更多信息，请参阅 在 Windows 上设置安全性， UNIX 和 Linux 系统。

名称服务
名称服务使应用程序能够通过识别远程队列 (就像它们是本地队列一样) 来共享队列。

您可以编写自己的名称服务组件。 例如，如果您打算将名称服务与 IBM MQ 配合使用，那么可能需要执行此操作。 要使用名称服务，您必须具有用户编写的组件或由其他软件供应商提供的组件。 缺省情况下，名称服务处于不活动状态。

IBM MQ Java 语言接口

IBM MQ 提供了两个备用应用程序编程接口 (API) 以在 Java 应用程序中使用: IBM MQ classes for Java Message Service 和 IBM MQ classes for Java。

IBM 支持且是开放式标准的活动参与者，并且在消息传递区域中， API 标准是 Java Message Service (JMS)。 IBM MQ 8.0 实现了 JMS 2.0 标准，该标准引入了新的简化 API 以及诸如共享预订之类的功能。 此外， WebSphere Application Server Liberty 还支持具有缺省消息传递提供程序和 IBM MQ 的 JMS 2.0 。

在 IBM MQ 中，Java 应用程序中可以使用以下两个备选 API:

IBM MQ classes for JMS
IBM MQ classes for Java Message Service (JMS) 是 IBM MQ 随附的 JMS 提供程序。 Java Platform, Enterprise Edition Connector Architecture (JCA) 提供一种标准方式将运行在 Java EE 环境中的应用程序连接到企业信息系统 (EIS)，如 IBM MQ 或 Db2。

IBM MQ classes for Java
IBM MQ classes for Java 使您可以在 Java 环境中使用 IBM MQ。 IBM MQ classes for Java 允许 Java 应用程序作为 IBM MQ 客户机连接到 IBM MQ，或直接连接到 IBM MQ 队列管理器。

从 IBM MQ 8.0 开始，使用 Java 7 构建 IBM MQ classes for Java 和 IBM MQ classes for JMS 。

Java 7 运行时环境支持运行较早的类文件版本。

相关信息
为什么应该使用 IBM MQ classes for JMS?
为什么应该使用 IBM MQ classes for Java?
使用 JMS 2.0 功能
JMS 模型

IBM MQ classes for JMS

IBM MQ classes for JMS 是 IBM MQ 随附的 JMS 提供程序。 IBM MQ classes for JMS 实现 javax.jms 包中定义的接口，并且还提供对 JMS API 的两组扩展。 Java Platform， Standard Edition (Java SE) 和 Java Platform， Enterprise Edition (Java EE) 应用程序都可以使用 IBM MQ classes for JMS。

JMS 规范定义了一组可供应用程序用来执行消息传递操作的接口。 从 版本 8.0 开始， IBM MQ 支持 JMS 标准的 JMS 2.0 版本 (请参阅 IBM MQ 8.0 中的新增内容)。 此实现提供标准 API 的所有功能，但需要的接口更少，并且更便于使用。 有关更多信息，请参阅 JMS 模型 和 JMS 2.0 规范 (位于 Java.net)。

javax.jms 包指定了 JMS 接口的详细信息，JMS 提供者针对特定消息传递产品实现了这些接口。 IBM MQ classes for JMS 是 JMS 提供程序，用于实现 IBM MQ 的 JMS 接口，并且还为 JMS API 提供以下两组扩展:

- IBM MQ JMS 扩展
- IBM JMS 扩展

对于使用 javax.jms 接口或任意一组 JMS 扩展来创建的连接工厂、队列或主题对象，它可以使用其中任何 API 进行寻址；即，可以将其强制转换为任何接口。 为保持最大程度的应用程序可移植性，请使用符合您需求的最通用的 API。

IBM MQ JMS 扩展

IBM MQ classes for JMS 还提供了针对 JMS API 的扩展。 先前的 IBM MQ classes for JMS 发行版中包含了在 MQConnectionFactory、MQQueue 和 MQTopic 对象中实现的扩展。 这些对象包含特定于 IBM MQ 的属性和方法。 这些对象可以是受管对象，也可以由应用程序在运行时动态创建这些对象。 从 IBM WebSphere MQ 7.0 开始，IBM MQ classes for JMS 会维护这些扩展，您可以继续按原样使用任何利用这些扩展的应用程序。 这些扩展称为 IBM MQ JMS 扩展。 请注意，在此系列文档中，应用程序在运行时动态创建的对象不会被视为受管对象。

IBM JMS 扩展

除了 IBM MQ JMS 扩展之外，从 IBM WebSphere MQ 7.0 开始， IBM MQ classes for JMS 提供了一组更通用的 JMS API 扩展。 这些扩展称为 IBM JMS 扩展，能够实现以下广泛目标:

- 提高各个 IBM JMS 提供程序的一致程度
- 更容易编写两个 IBM 消息传递系统之间的桥接应用程序
- 更容易将应用程序从一个 IBM JMS 提供程序移植到另一个提供程序

这些扩展的主要目标是在运行时动态创建和配置连接工厂和目标，但是这些扩展还提供了与消息传递不直接相关的功能，例如问题确定功能。

对于使用 javax.jms 接口或任意一组 JMS 扩展来创建的连接工厂、队列或主题对象，它可以使用其中任何 API 进行寻址；即，可以将其强制转换为任何接口。 为保持最大程度的应用程序可移植性，请使用符合您需求的最通用的 API。

JMS 模型

JMS 模型定义了一组接口， Java 应用程序可以使用这些接口来执行消息传递操作。 IBM MQ classes for JMS 作为 JMS 提供程序，定义 JMS 对象如何与 IBM MQ 概念相关。 JMS 规范认为某些 JMS 对象是受管对象。

JMS 规范和 javax.jms 包一起定义了一组可供 Java 应用程序用来执行消息传递操作的接口。

IBM MQ 8.0 支持 JMS 标准的 JMS 2.0 版本，该版本引入了简化的 API ，同时还保留了经典 API (从 V 1.1 开始)。

简化的 API

JMS 2.0 引入了简化的 API ，同时还保留了 JMS 1.1 中特定于域的接口和独立于域的接口。 简化的 API 减少了发送和接收消息所需的对象数，由以下接口构成:

ConnectionFactory
ConnectionFactory 是供 JMS 客户机用来创建连接的受管对象。 此接口也用于标准 API。

JMS 上下文
此对象结合了标准 API 的 Connection 和 Session 对象。 可以通过复制底层连接来从其他 JMSContext 对象创建 JMSContext 对象。

JMSProducer
JMSProducer 由 JMSContext 创建，并用于向队列或主题发送消息。 JMSProducer 对象会导致创建发送消息所需的对象。

JMSConsumer
JMSConsumer 由 JMSContext 创建，并用于从主题或队列接收消息。

简化的 API 具有以下影响：

- JMSContext 对象始终自动启动底层连接。
- JMSProducer 和 JMSConsumer 现在可以直接处理消息体，而无需使用消息的 getBody 方法来获取整个消息对象。
- 在发送“消息体”（即消息内容）之前，可以使用方法链在 JMSProducer 对象上设置消息属性。 JMSProducer 将负责创建发送消息所需的所有对象。 通过使用 JMS 2.0，可以按如下所示设置属性和发送消息：

```
context.createProducer().
setProperty("foo", "bar").
setTimeToLive(10000).
setDeliveryMode(NON_PERSISTENT).
setDisableMessageTimestamp(true).
send(dataQueue, body);
```

JMS 2.0 还引入了可用于在多个使用者之间共享消息的共享预订。 所有 JMS 1.1 预订均视为非共享预订。

标准 API

以下列表汇总了标准 API 的主要 JMS 接口：

Destination
目标是应用程序发送消息的位置和/或应用程序从中接收消息的源。

ConnectionFactory
ConnectionFactory 对象封装了连接的一组配置属性。 应用程序使用连接工厂来创建连接。

Connection
Connection 对象将应用程序的活动连接封装到消息传递服务器中。 应用程序使用连接来创建会话。

Session
会话是用于发送和接收消息的单线程上下文。 应用程序使用会话来创建消息、消息生产者和消息使用者。 会话将进行事务处理或不进行事务处理。

消息
Message 对象封装了应用程序发送或接收的消息。

MessageProducer
应用程序使用消息生产者向目标发送消息。

MessageConsumer
应用程序使用消息使用者接收向目标发送的消息。

第 275 页的图 54 显示了这些对象及其关系。

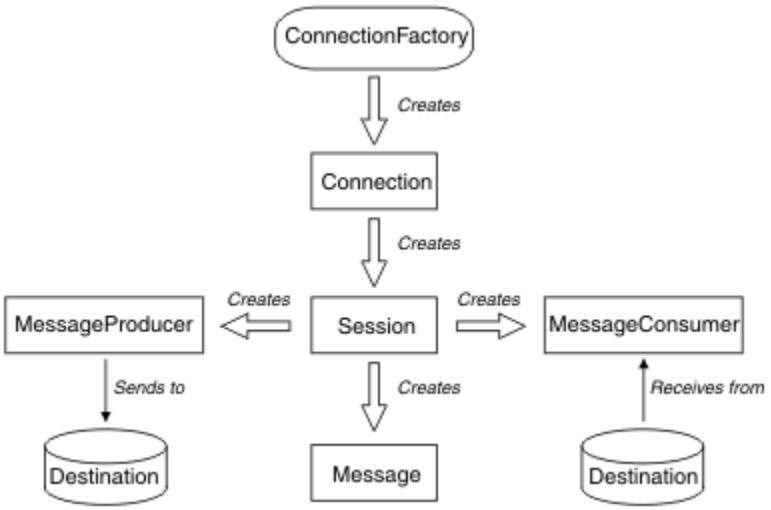


图 *54: JMS* 对象及其关系

此图显示了以下主要接口：ConnectionFactory、Connection、Session、MessageProducer、MessageConsumer、Message 和 Destination。 应用程序使用连接工厂来创建连接，并使用连接来创建会话。 然后，应用程序可以使用会话来创建消息、消息生产者和消息使用者。 应用程序使用消息生产者向目标发送消息，并使用消息使用者接收向目标发送的消息。

Destination、ConnectionFactory 或 Connection 对象可以由多线程应用程序的不同线程并行使用，但 Session、MessageProducer 或 MessageConsumer 对象不能由不同线程并行使用。 确保 Session、MessageProducer 或 MessageConsumer 对象不会被并行使用的最简单方法是为每个线程创建单独的 Session 对象。

JMS 支持以下两种类型的消息传递：

- 点到点消息传递
- 发布/预订消息传递

这两种类型的消息传递也称为消息传递域，您可以在应用程序中结合使用这两种类型的消息传递。 在点到点域中，目标是队列，在发布/预订域中，目标是主题。

对于 JMS 之前的 JMS 1.1 版本，点到点域的编程使用的是一组接口和方法，而发布/预订域的编程则使用的是另一组接口和方法。 二者相似，但相互独立。 从 JMS 1.1 起，您可以使用一组支持这两个消息传递域的公共接口和方法。 通用接口为每个消息传递域提供独立于域的视图。 第 275 页的表 20 列出了独立于 JMS 域的接口及对应的域特定接口。

表 *20:* 独立于 *JMS* 域的接口和特定于域的接口

独立于域的接口	**点到点域的域特定接口**	**发布/预订域的域特定接口**
ConnectionFactory	QueueConnectionFactory	TopicConnectionFactory
Connection	QueueConnection	TopicConnection
Destination	队列	Topic
Session	QueueSession	TopicSession
MessageProducer	QueueSender	TopicPublisher
MessageConsumer	QueueReceiver QueueBrowser	TopicSubscriber

JMS 2.0 保留了所有特定于域的接口，因此现有应用程序仍可以使用这些接口。 但对于新的应用程序，请考虑使用 1.1 中独立于域的接口或 2.0 中的简化 API。

在 IBM MQ classes for JMS 中，JMS 对象在以下方面与 IBM MQ 概念相关：

- Connection 对象具有派生自连接工厂（用于创建连接）属性的属性。 这些属性控制应用程序如何连接到队列管理器。 这些属性的示例包括队列管理器名称以及运行队列管理器的系统的主机名或 IP 地址（针对以 CLIENT 方式连接到队列管理器的应用程序）。
- Session 对象封装了 IBM MQ 连接句柄，因此定义了会话的事务范围。
- MessageProducer 对象和 MessageConsumer 对象各自封装了一个 IBM MQ 对象句柄。

在使用 IBM MQ classes for JMS 时，IBM MQ 的所有标准规则都适用。 尤其需要注意，应用程序可以向远程队列发送消息，但只能从应用程序所连接到的队列管理器拥有的队列中接收消息。

JMS 规范认为 ConnectionFactory 和 Destination 对象都是受管对象。 管理员在中央存储库中创建和维护受管对象，而 JMS 应用程序可使用 Java 命名和目录接口 (JNDI) 来检索这些对象。

在 IBM MQ classes for JMS 中，Destination 接口的实现是 Queue 和 Topic 的抽象超类，因此 Destination 实例是 Queue 对象或 Topic 对象。 独立于域的接口将队列或主题视为目标。 MessageProducer 或 MessageConsumer 对象的消息传递域由目标是队列还是主题来确定。

因此，在 IBM MQ classes for JMS 中，以下类型的对象可以是受管对象：

- ConnectionFactory
- QueueConnectionFactory
- TopicConnectionFactory
- 队列
- Topic
- XAConnectionFactory
- XAQueueConnectionFactory
- XATopicConnectionFactory

IBM MQ classes for JMS 体系结构

IBM MQ classes for JMS 具有最初在 IBM WebSphere MQ 7.0 中引入的分层体系结构。

在 IBM WebSphere MQ 7.0 之前，IBM WebSphere MQ classes for JMS 的实现完全特定于 IBM WebSphere MQ。 提供消息传递系统的其他 IBM 产品中也包含 JMS 提供者，但是这些 JMS 提供者与 IBM MQ classes for JMS 实现几乎或完全没有共同点。

从 IBM WebSphere MQ 7.0 开始，IBM MQ classes for JMS 具有分层体系结构，如图第 277 页的图 55 中所示。 最上层代码是可供任何 IBM JMS 提供程序使用的公共层。 当应用程序调用 JMS 方法时，不特定于消息传递系统的调用的任何处理都由公共层执行，这也提供对调用的一致响应。 对特定于消息传递系统的调用的任何处理将委派到下一层。 在下图中，IBM MQ 消息传递提供者以及另外两个消息传递提供者（消息传递提供者 A 和消息传递提供者 B）都显示在下一层中。

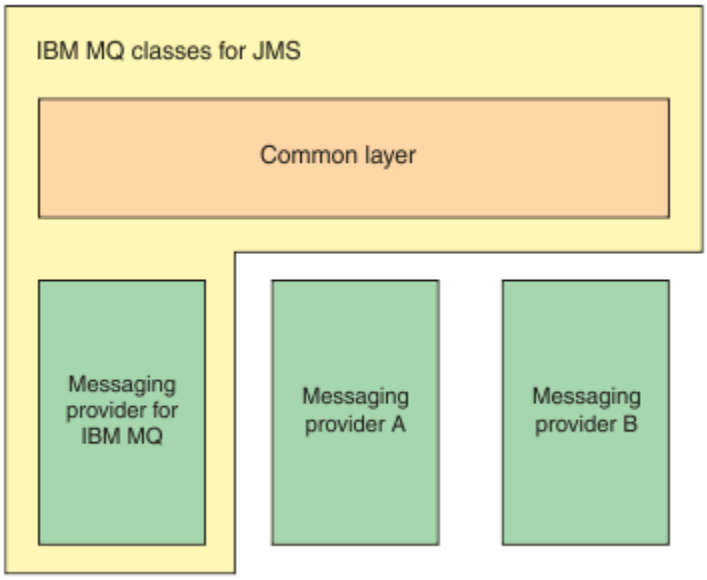


图 *55: IBM JMS* 提供者的分层体系结构

分层体系结构实现了以下目标:

- 提高各个 IBM JMS 提供者行为的一致性
- 更容易编写两个 IBM 消息传递系统之间的桥接应用程序
- 更容易将应用程序从一个 IBM JMS 提供程序移植到另一个提供程序

受管对象支持

IBM MQ classes for JMS 支持使用受管对象。

JMS 应用程序中的逻辑流以 ConnectionFactory 和 Destination 对象开头。 应用程序使用 ConnectionFactory 对象来创建 Connection 对象，该对象表示从应用程序到消息传递服务器的活动连接。 应用程序使用 Connection 对象来创建 Session 对象，它是用于生成和使用消息的单线程上下文。 然后，应用程序可以使用 Session 对象和 Destination 对象来创建 MessageProducer 对象，应用程序使用该对象将消息发送到指定的目标。 目标是消息传递系统中的队列或主题，并由 Destination 对象封装。 应用程序还可以使用 Session 对象和 Destination 对象来创建 MessageConsumer 对象，应用程序使用该对象来接收已发送到指定目标的消息。

JMS 规范认为 ConnectionFactory 和 Destination 对象都是受管对象。 管理员在中央存储库中创建和维护受管对象，而 JMS 应用程序可使用 Java 命名和目录接口 (JNDI) 来检索这些对象。 受管对象存储库可以小至单个文件，大至轻量级目录访问协议 (LDAP) 目录。

IBM MQ classes for JMS 支持使用受管对象。 应用程序可以使用通过 IBM MQ 公开的 IBM MQ classes for JMS 的所有功能，而无需将任何特定于 IBM MQ 的信息硬编码到应用程序本身中。 这种安排使应用程序在一定程度上独立于底层 IBM MQ 配置。 为实现这种独立性，应用程序可以使用 JNDI 来检索已存储为受管对象的连接工厂和目标，并且仅使用 javax.jms 包中定义的接口来执行消息传递操作。 管理员可以使用 IBM MQ JMS 管理工具或 IBM MQ Explorer 在中央存储库中创建和维护受管对象。 但是，应用程序服务器通常为受管对象提供自己的存储库，并提供用于创建和维护这些对象的自有工具。 因此，Java EE 应用程序可以使用 JNDI 来从应用程序服务器存储库或中央存储库中检索受管对象。

相关信息

配置 JMS 资源

Java EE 平台上支持的通信类型

在 Java EE 平台上，IBM MQ classes for JMS 在应用程序组件和 IBM MQ 队列管理器之间支持两种类型的通信。

在应用程序组件和 IBM MQ 队列管理器之间支持以下两种类型的通信：

- 出站通信
- 入站通信

出站通信

通过直接使用 JMS API，应用程序组件可以创建与队列管理器的连接，然后发送和接收消息。

例如，应用程序组件可以是应用程序客户机、servlet、Java Server Page (JSP)、企业 Java bean (EJB) 或消息驱动的 bean (MDB)。 在此类型的通信中，应用程序服务器容器在消息传递操作支持中仅提供低级功能，例如连接池和线程管理。

入站通信

如果使用的是入站通信，那么到达目标的消息将传递到 MDB，然后由 MDB 负责处理该消息。

Java EE 应用程序使用 MDB 以异步方式处理消息。 MDB 用作 JMS 消息侦听器，并由用于定义消息处理方式的 onMessage() 方法实现。 已将 MDB 部署到应用程序服务器的 EJB 容器中。 MDB 的精确配置方式取决于所使用的应用程序服务器，但配置信息必须指定要连接到的队列管理器、如何连接到队列管理器、要监视哪个目标中的消息以及 MDB 的事务行为。 然后，MDB 容器将使用这些信息。 当满足 MDB 选择条件的消息到达指定的目标时，EJB 容器会使用 IBM MQ classes for JMS 来从队列管理器中检索该消息，然后调用其 onMessage() 方法将该消息传递到 MDB。

与 IBM MQ classes for Java 的关系

从 IBM WebSphere MQ 7.0 开始，IBM WebSphere MQ classes for JMS 和 IBM WebSphere MQ classes for Java 实现为同级。 此实现不同于先前发行版，在先前发行版中，IBM WebSphere MQ classes for JMS 的实现依赖于 IBM WebSphere MQ classes for Java。

对于低于 IBM WebSphere MQ 7.0 的版本，IBM WebSphere MQ classes for JMS 几乎完全实现为基于 IBM WebSphere MQ classes for Java 的代码层。 这种安排会让应用程序开发者产生一些混淆，因为 MQEnvironment 类中的设置字段或调用方法可能导致对使用 IBM MQ classes for JMS 编写的代码运行时行为产生不想要的意外影响。 此外，在 JMS API 并非基于 IBM MQ classes for Java 的自然拟合的区域中，IBM MQ classes for JMS 的实现具有一些约束，并且这些约束导致了一些与运行时性能相关的问题。

从 IBM WebSphere MQ 7.0 开始，IBM MQ classes for JMS 的实现不再依赖于 IBM MQ classes for Java。 IBM MQ classes for Java 和 IBM MQ classes for JMS 现在是使用 MQI 通用 Java 接口的同级。 这种安排提供了更大的性能优化空间，意味着 MQEnvironment 类中的设置字段或调用方法对使用 IBM MQ classes for JMS 编写的代码运行时行为没有影响。 第 279 页的图 56 显示了 IBM WebSphere MQ classes for JMS 的先前发行版和 IBM WebSphere MQ 7.0 之前的发行版中的 IBM MQ classes for JMS 与 IBM MQ classes for Java 之间的关系以及此关系在后续发行版中的更改方式。

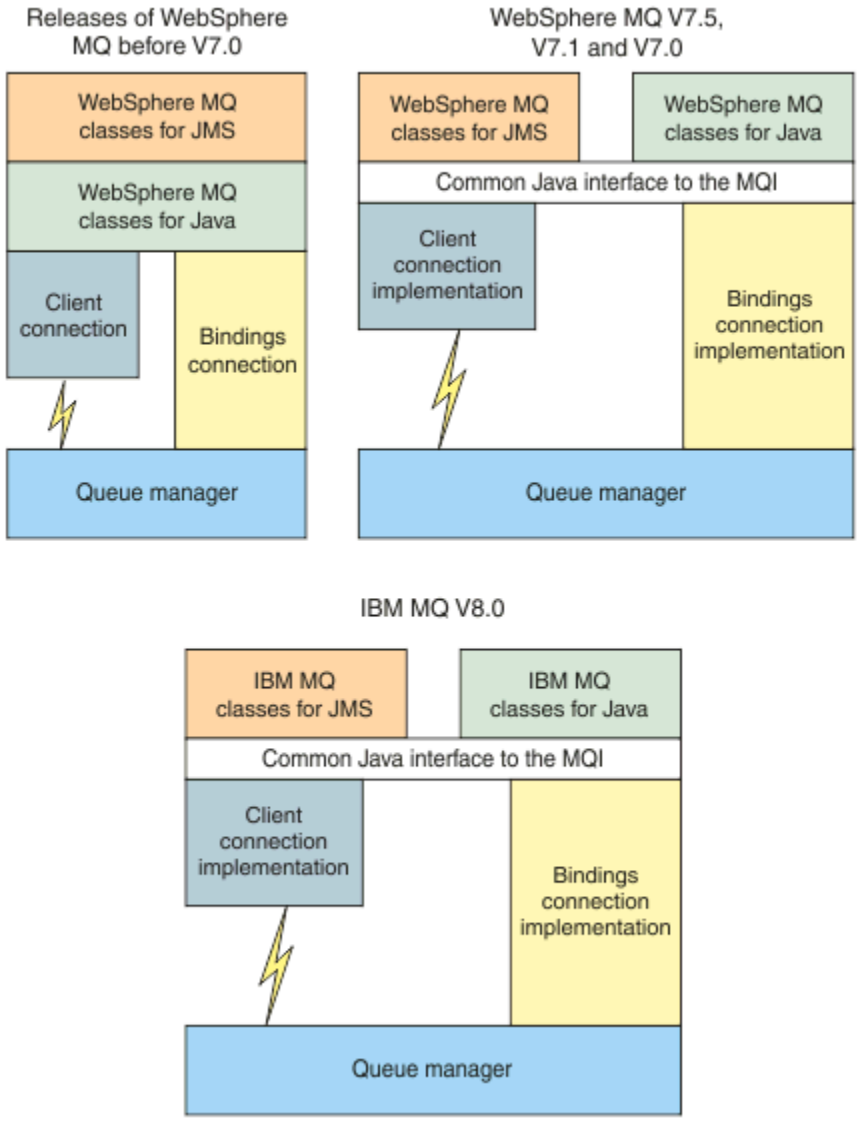


图 *56: IBM MQ classes for JMS* 与 *IBM MQ classes for Java* 之间的关系

为保持与 IBM WebSphere MQ 7.0 之前的发行版的兼容性，用 Java 编写的通道出口类仍可以使用 IBM MQ classes for Java 接口，即使从 IBM MQ classes for JMS 调用这些通道出口类也是如此。 但是，使用 IBM MQ classes for Java 接口意味着您的应用程序仍依赖于 IBM MQ classes for Java JAR 文件 `com.ibm.mq.jar`。 如果不想在类路径中包含 `com.ibm.mq.jar`，可以改为使用 `com.ibm.mq.exits` 包中的接口集。

从 IBM WebSphere MQ 7.0 开始，您可以使用 IBM MQ Explorer 来创建和配置 JMS 受管对象。

IBM MQ 消息传递提供者方式

IBM MQ 消息传递提供者具有三种运行方式：正常方式、带有限制的正常方式和迁移方式。

IBM MQ 消息传递提供者具有以下三种运行方式：

- IBM MQ 消息传递提供者正常方式
- IBM MQ 消息传递提供者带有限制的正常方式
- IBM MQ 消息传递提供者迁移方式

IBM MQ 消息传递提供者正常方式使用 IBM MQ 队列管理器的所有功能来实现 JMS。 此方式已优化为使用 JMS 2.0 API 和功能。

IBM MQ 消息传递提供程序带有限制的正常方式使用 JMS 2.0 API，但不使用新的 IBM MQ 8.0 功能（例如共享预订、延迟传递或异步发送）。

IBM MQ 消息传递提供者迁移方式基于 IBM WebSphere MQ 6.0 功能，仅使用先前在 IBM WebSphere MQ 6.0 队列管理器中可用的功能来实现 JMS。 您可以连接到使用 IBM MQ 消息传递提供程序迁移方式的 IBM WebSphere MQ 7.0 或更高版本的队列管理器，但不能使用任何 版本 7.0 优化。 此方式允许连接到以下任一队列管理器版本：

1. 以 BINDINGS 或 CLIENT 方式连接到 IBM WebSphere MQ 7.0 或更高版本的队列管理器，但此方式仅使用先前可供 IBM WebSphere MQ 6.0 队列管理器使用的那些功能。
2. 以 CLIENT 方式连接到 IBM WebSphere MQ 6.0 或更低版本的队列管理器。

如果要使用 IBM MQ Enterprise Transport 连接到 IBM Integration Bus ，请使用 IBM MQ 消息传递提供程序迁移方式。 如果使用 IBM MQ Real-Time Transport，那么会自动选择 IBM MQ 消息传递提供程序迁移方式，因为您已在连接工厂对象中显式选择了相应属性。 使用 IBM MQ Enterprise Transport 与 IBM Integration Bus 的连接遵循 用于选择 IBM MQ 消息传递提供程序方式的规则中描述的方式选择的一般规则。

相关信息

配置 JMS PROVEIDERVERSION 属性

IBM MQ.NET 受管客户机的 SSL 和 TLS 支持

IBM MQ.NET 完全受管客户机提供基于 Microsoft.NET SSLStreams 套件的安全套接字层 (SSL) 和传输层安全性 (TLS) 支持。 这与基于 GSKit 的其他 IBM MQ 客户机不同。

您可以开发 IBM MQ.NET 应用程序以在受管方式或非受管方式下运行。

- 在受管方式下， .NET 应用程序在 .NET CLR (公共语言运行时) 中工作，而无需任何跨平台调用 (例如调用 C MQI)。
- 在非受管方式下，将针对底层 MQI 操作调用 C MQI。 基本上，非受管方式接口包含基于 C MQI 的 .NET 包装程序类。

受管 IBM MQ.NET 客户机使用 Microsoft.NET Framework 库来实现 SSL 和 TLS 安全套接字协议。 Microsoft 中的系统.NET.Security.SSLStream 类用于在 IBM MQ.NET 中实现安全性 (SSL/TLS)。

非受管 IBM MQ.NET 客户机方式已支持基于 C MQI (和 GSKit) 的 SSL 功能。 即， SSL 操作由 C MQI 处理。 在这种情况下， GSKit 将实现 SSL 和 TLS 安全套接字协议。

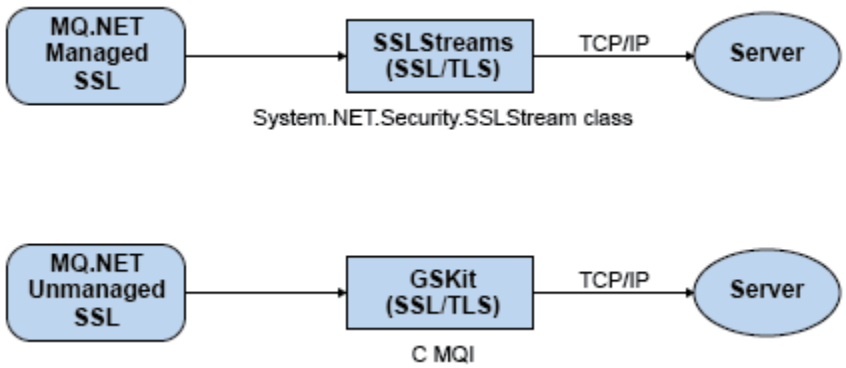


图 *57: IBM MQ.NET* 受管和非受管 *SSL/TLS* 比较

下表汇总了受管实现与非受管实现之间的差异:

表 21: 受管实施与非受管实施之间的差异			
模式	**协议**	**实施**	**注释**
IBM MQ.NET 管理的 SSL	SSL TLS	系统.NET.Security.SSLStream 类 SSLStream 类通过已连接的 TCP 套接字作为流运行	SSL 3.0， TLS 1.0 TLS1.2 (仅适用于 Microsoft.NET Framework v4.5)
IBM MQ.NET 非受管 SSL	SSL TLS	GSKIT 和 C-MQI	SSL 和 TLS 安全套接字协议

相关信息
.NET 的安全套接字层 (SSL) 和传输层安全性 (TLS) 支持

HP Integrity NonStop Server 的 IBM MQ 客户机

了解 HP Integrity NonStop Server 平台的 IBM MQ 客户机。

IBM MQ 现在支持 HP Integrity NonStop Server 平台的客户机。

概述

有关 IBM MQ 客户机 (包括 HP Integrity NonStop Server 平台的客户机) 的概述，请参阅 第 265 页的『IBM MQ MQI clients 概述』。

有关 IBM MQ Client for HP Integrity NonStop Server 平台的技术概述，请参阅 第 284 页的『IBM MQ Client for HP Integrity NonStop Server 技术概述』。

有关 HP Integrity NonStop Server 支持的环境和功能部件的 IBM MQ 客户机的详细信息，请参阅 第 284 页的『IBM MQ Client for HP Integrity NonStop Server 支持的环境和功能部件』。

规划

有关规划 IBM MQ Client for HP Integrity NonStop Server 环境的帮助，请参阅 Planning your IBM MQ client environment on HP Integrity NonStop Server 。

安装

有关安装 HP Integrity NonStop Server 的 IBM MQ 客户机的帮助。

- 选择要安装的内容，请参阅 HP Integrity NonStop Server 的 IBM MQ 客户机组件 。
- 规划安装，请参阅 在 HP Integrity NonStop Server 上规划安装
  - 文件系统
- 硬件和软件需求，请参阅 HP Integrity NonStop Server 系统上的硬件和软件需求。
- 验证您是否具有正确的软件，请参阅 验证系统软件先决条件。
- 准备系统，请参阅 在 HP Integrity NonStop Server 上设置用户和组 。
- 安装客户机，请参阅 在 HP Integrity NonStop Server 系统上安装 IBM MQ 客户机。
- 验证安装，请参阅 验证客户机安装。
- 卸载，请参阅 在 HP Integrity NonStop Server 上卸载 IBM MQ 。

HP Integrity NonStop Server 客户机命令

以下命令适用于 HP Integrity NonStop Server OSS 和监护器环境的 IBM MQ 客户机:

- dspmqtrc
- dspmqver
- endmqtrc
- mqrc
- runmqras
- runmqsc
- runmqtmc
- strmqtrc

以下命令适用于 IBM MQ Client for HP Integrity NonStop Server OSS 环境:

- dspmqtrc

向 DISPLAY CHSTATUS 命令 Product Identifier values 表添加了新产品标识 MQNC。

安全性

要保护 IBM MQ Client for HP Integrity NonStop Server 环境，请参阅:

- 有关 IBM MQ client for HP Integrity NonStop Server 如何向添加到 Planning authentication for a client application 的队列管理器标识自身的信息。
- 在 HP Integrity NonStop Server 上设置安全性
  - OpenSSL
  - 熵守护程序
- IBM MQ 中的 SSL 和 TLS 安全协议
- 在 HP Integrity NonStop Server 上使用 SSL 或 TLS
  - 证书管理
  - 个人证书库
  - 证书信任库
  - 传递短语隐藏文件
  - 证书撤销列表文件

事务管理设施

有关事务管理设施 (TMF) 的信息，请参阅以下部分和主题。

- 在 HP Integrity NonStop Server 上规划 IBM MQ 客户机环境
  - 准备 HP Integrity NonStop Server 环境
  - IBM MQ 和 HP NonStop TMF
  - 使用 HP NonStop TMF
    - 使用全局工作单元
    - 避免长时间运行的事务
    - 有关在添加到 即将到期的全局工作单元的预先配置的不活动时间间隔之后使全局工作单元到期的队列管理器配置的信息。
- 配置 HP Integrity NonStop Server
  - 网关进程概述
  - 配置 Gateway 以在 Pathway 下运行

  - TMF 和 TMF/Gateway 节
  - 配置客户机初始化文件
  - 向通道授予许可权
- 管理 HP Integrity NonStop Server
  - 从 Pathway 手动启动 TMF/Gateway
  - 从 Pathway 停止 TMF/Gateway
- IBM MQ Client for HP Integrity NonStop Server 故障诊断

开发应用程序

有关为 HP Integrity NonStop Server 平台上的 IBM MQ 客户机开发应用程序的信息，请参阅：

- 在 HP Integrity NonStop Server 上构建过程应用程序
  - OSS 以及监护器头和公共库
  - 在 HP Integrity NonStop Server 中准备 C 程序
  - 准备 COBOL 程序
  - 准备 pTAL 程序
- 有关在 pTAL 中编码的信息，请参阅 在 pTAL 中编码。
- 有关为 HP Integrity NonStop Server 的 IBM MQ 客户机准备 JMS 程序的信息，请参阅 为 HP Integrity NonStop Server 的 IBM MQ 客户机准备 JMS 程序 。

新消息

以下是 HP Integrity NonStop Server 上 IBM MQ 客户机的新消息：

- AMQ5xxx: 可安装服务
  - AMQ5370
  - AMQ5371
  - AMQ5372
  - AMQ5373
  - AMQ5374
  - AMQ5375
  - AMQ5376
  - AMQ5377
  - AMQ5378
  - AMQ5379
  - AMQ5380
  - AMQ5390
  - AMQ5391
  - AMQ5392
  - AMQ5393
  - AMQ5394
  - AMQ5395
  - AMQ5396
  - AMQ5397
  - AMQ5398
  - AMQ5399

- AMQ9xxx: 远程
  - AMQ9816
  - AMQ9817
  - AMQ9818
  - AMQ9819
  - AMQ9820
  - AMQ9821
  - AMQ9823
  - AMQ9824

已修改的 API 原因码

以下现有 API 原因码现在包含 HP Integrity NonStop Server:

- 2354 (0932) (RC2354): MQRC_UOW_ENLISTMENT_ERROR
- 2355 (0933) (RC2355): MQRC_UOW_MIX_NOT_SUPPORTED
- 2072 (0818) (RC2072): MQRC_SYNCPOINT_NOT_AVAILABLE
- 2003 (07D3) (RC2003): MQRC_BACKED_OUT

样本

有关 HP Integrity NonStop Server 上 IBM MQ 客户机的样本程序所演示的方法的信息，请参阅 HP Integrity NonStop Server 的 IBM MQ 客户机样本 。

故障诊断和支持

有关 HP Integrity NonStop Server 上 IBM MQ 客户机的故障诊断和支持信息，请参阅以下主题:

- IBM MQ Client for HP Integrity NonStop Server 故障诊断
- HP Integrity NonStop Server 上的错误日志
- 在 HP Integrity NonStop Server 上使用跟踪
- FFST: IBM WebSphere MQ for HP Integrity NonStop Server

IBM MQ Client for HP Integrity NonStop Server 技术概述

HP Integrity NonStop Server 操作系统的技术概述。

HP Integrity NonStop Server 是专为实现最高可用性而设计的操作系统，即使存在多个硬件或软件故障，也不会出现计划内或计划外的停机时间。 它是线性可扩展的，例如，如果增加 20% 的硬件，那么可以获得 20% 的可用性能。 为了保持数据完整性，操作系统有自己的事务管理器和事务文件系统。

HP Integrity NonStop Server 操作系统通常由以下用户使用:

- 金融机构，例如对于 ATM 网络，网上银行支持，信贷授权，证券交易所交换机，交易以及银行到银行的交易。
- 例如，用于 Web 商店后端，库存和过程控制的制造。
- 例如，用于交换，紧急和其他网络服务的电信。

IBM MQ Client for HP Integrity NonStop Server 支持的环境和功能部件

提供有关 HP Integrity NonStop Server 平台的 IBM MQ 客户机的详细信息，并描述特定于 HP Integrity NonStop Server 系统的受支持客户机 API 和环境以及客户机功能。

受支持的客户机 API 和环境

IBM MQ Client for HP Integrity NonStop Server 支持以下执行环境:

表 22:		
	OSS	**监护器**
C	✓	✓
JMS	✓	
COBOL	✓	✓
pTAL	✓	✓

功能摘要

一些客户机功能方面特定于主机操作系统。 以下摘要描述了特定于 HP Integrity NonStop Server 的 IBM MQ 客户机的客户机功能的各个方面:

- C (本机)， PTAL， COBOL (本机)
  - 网络协议: TCP (IPv4 和 IPv6)
  - 传输类型: 仅限客户机
  - 传输安全性: SSL/TLS
  - 事务支持: 由事务管理设施 (TMF) 协调的两阶段落实 (需要连接到位于 IBM WebSphere MQ 7.1 或更高版本的队列管理器)
  - 寻址方式: 32 位
- Java Message Service (JMS)
  - 网络协议: TCP (IPv4 和 IPv6)
  - 传输类型: 仅客户机 (不支持绑定，直接和直接 HTTP)
  - 传输安全性: SSL/TLS
  - 事务支持: 单阶段落实
  - 执行: 独立 (不支持 Application Support Facility (ASF) 和 Java 连接器体系结构 (JCA))
  - 出口: 仅 Java 语言 (不支持使用其他语言编写的本机出口)
  - IBM MQ 头和 PCF: 不支持以下类: com.ibm.mq.headers.* 和 com.ibm.mq.pcf.*

IBM MQ for z/OS 概念

IBM MQ for z/OS 使用的某些概念对于 z/OS 平台是唯一的。 例如，仅随 IBM MQ for z/OS 提供了日志记录机制，存储管理技术，恢复处置单元和队列共享组。 请使用本主题以获取有关这些概念的更多信息。

相关概念

第 286 页的『z/OS 上的队列管理器』
必须先安装 IBM MQ for z/OS 产品并启动队列管理器，然后才能让应用程序在 z/OS 系统上使用 IBM MQ 。 队列管理器拥有并管理 IBM MQ 所使用的资源集。

第 287 页的『z/OS 上的通道启动程序』
通道启动程序 提供并管理用于启用 IBM MQ 分布式排队的资源。 IBM MQ 使用 消息通道代理程序 (MCA) 将消息从一个队列管理器发送到另一个队列管理器。

第 289 页的『术语和任务』
使用本主题作为术语以及特定于 IBM MQ for z/OS 的任务的简介。

第 291 页的『共享队列和队列共享组』
您可以使用共享队列和队列共享组来实现 MQ 资源的高可用性。

第 328 页的『组内排队』
本部分描述了组内排队，这是 z/OS 平台独有的 IBM MQ for z/OS 函数。 此功能仅对定义到队列共享组的队列管理器可用。

第 339 页的『存储管理』
IBM MQ for z/OS 需要永久和临时数据结构，并使用页集和内存缓冲区来存储此数据。 这些主题提供了有关 IBM MQ 如何利用这些页集和缓冲区的更多详细信息。

第 343 页的『日志记录』
IBM MQ 维护发生数据更改和重要事件时的 日志 。 如果需要，可以使用这些日志将数据恢复到先前状态。

第 361 页的『恢复和重新启动』
使用本主题中的链接可了解 IBM MQ for z/OS 用于重新启动和恢复的功能。

第 374 页的『z/OS 上的安全概念』
使用本主题来了解安全性对于 IBM MQ 的重要性，以及在系统上没有足够的安全性设置所产生的影响。

第 379 页的『可用性』
IBM MQ for z/OS 具有许多用于实现高可用性的功能。 本主题描述了可用性的一些注意事项。

第 383 页的『恢复单元处置』
某些事务应用程序在连接到队列共享组 (QSG) 中的队列管理器时，可以使用 GROUP (而不是 QMGR) 恢复处置单元，方法是在它们连接时指定 QSG 名称而不是队列管理器名称。 这将通过除去重新连接到 QSG 中的同一队列管理器的需求，使事务恢复更加灵活和稳健。

相关参考
第 352 页的『定义系统』
IBM MQ for z/OS 使用许多缺省对象定义，并提供样本 JCL 来创建这些缺省对象。 使用本主题来了解这些缺省对象以及样本 JCL。

第 382 页的『监控和统计信息』
IBM MQ for z/OS 具有一组用于监视队列管理器和收集统计信息的工具。

z/OS 上的队列管理器

必须先安装 IBM MQ for z/OS 产品并启动队列管理器，然后才能让应用程序在 z/OS 系统上使用 IBM MQ 。 队列管理器拥有并管理 IBM MQ 所使用的资源集。

队列管理器

队列管理器 是向应用程序提供消息传递服务的程序。 使用消息队列接口（MQI）的应用程序可以将消息放置到队列并可从队列中获取消息。 队列管理器确保消息可以发送至正确的队列或传递至另一个队列管理器。 队列管理器处理向它发出的 MQI 调用以及提交给它的命令（无论从哪个源）。 队列管理器为每个调用或命令生成合适的完成代码。

队列管理器管理的资源包括:

- 用于保存 IBM MQ 对象定义和消息数据的页集
- 用于在队列管理器发生故障时恢复消息和对象的日志
- 处理器存储器
- 不同应用程序环境 (CICS， IMS 和批处理) 可通过其访问 IBM MQ API 的连接
- IBM MQ 通道启动程序，允许 z/OS 系统上的 IBM MQ 与其他系统之间进行通信

队列管理器具有名称，应用程序可以使用此名称连接到该队列管理器。

第 287 页的图 58 说明了队列管理器，其中显示了与不同应用程序环境的连接以及通道启动程序。

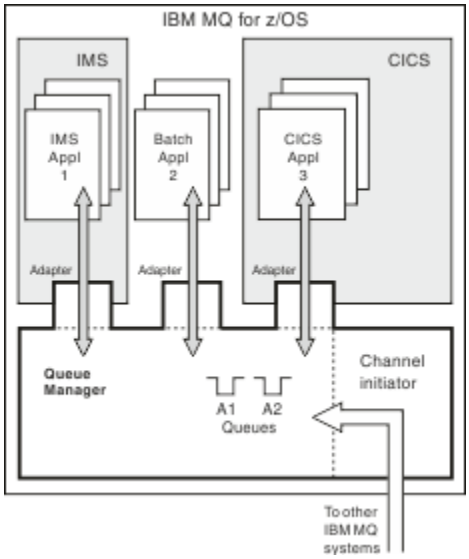


图 *58: IBM MQ for z/OS* 概述

z/OS 上的队列管理器子系统

在 z/OS 上， IBM MQ 作为在 IPL 时启动的 z/OS 子系统运行。 在子系统中，通过执行 JCL 过程来启动队列管理器，该过程指定 z/OS 数据集，这些数据集包含有关日志的信息，并保存对象定义和消息数据 (页集)。 子系统和队列管理器具有相同的名称，最多四个字符。 网络中的所有队列管理器都必须具有唯一名称，即使它们位于不同的系统，综合系统或平台上也是如此。

z/OS 上的通道启动程序

通道启动程序 提供并管理用于启用 IBM MQ 分布式排队的资源。 IBM MQ 使用 消息通道代理程序 (MCA) 将消息从一个队列管理器发送到另一个队列管理器。

要将消息从队列管理器 A 发送到队列管理器 B ，队列管理器 A 上的 发送 MCA 必须设置到队列管理器 B 的通信链路。 必须在队列管理器 B 上启动 接收 MCA ，才能从通信链路接收消息。 此单向路径由发送 MCA ，通信链路和接收 MCA 组成，称为 通道。 发送 MCA 从传输队列中获取消息并将它们从通道发送到接收 MCA。 接收 MCA 接收消息并将其放入目标队列。

在 IBM MQ for z/OS 中，发送和接收 MCA 都在通道启动程序 (通道启动程序也称为 移动者) 内运行。 通道启动程序在队列管理器的控制下作为 z/OS 地址空间运行。 只能有单个通道启动程序连接到队列管理器，并且它在与队列管理器相同的 z/OS 映像中运行。 可以同时在通道启动程序中运行数千个 MCA 进程。

第 288 页的图 59 显示了综合系统中的两个队列管理器。 每个队列管理器都有一个通道启动程序和一个本地队列。 由 AIX 和 Windows 上的队列管理器发送的消息将放置在本地队列上，由应用程序从中检索这些消息。 应答消息由类似的路由返回。

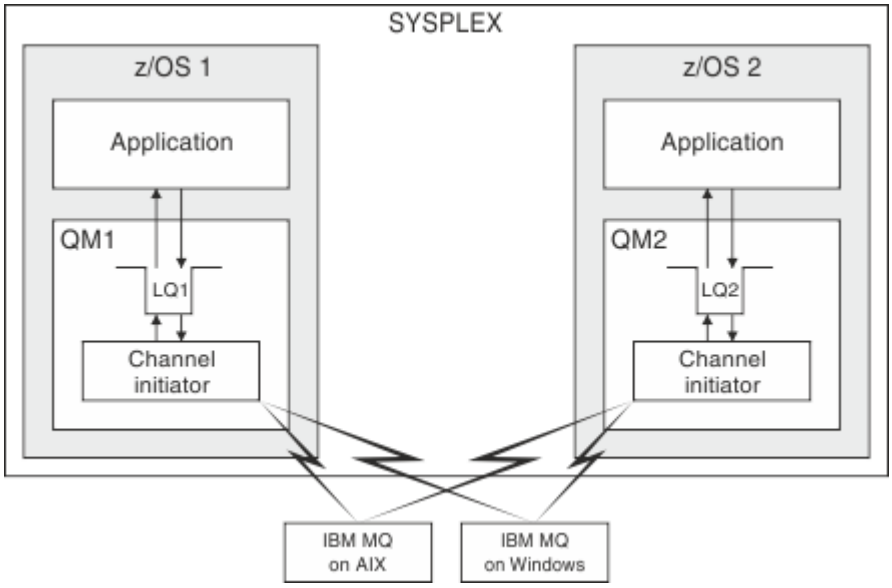


图 *59:* 队列管理器之间的通信

通道启动程序还包含与通道管理相关的其他进程。 这些过程包括:

侦听器
这些进程在诸如 TCP 之类的通信子系统上侦听入站通道请求，并在接收到入站请求时启动指定的 MCA。

主管
这将管理通道启动程序地址空间，例如，它负责在发生故障后重新启动通道。

名称服务器
这用于将 TCP 名称解析为地址。

SSL 任务
这些用于执行加密和解密以及检查证书撤销列表。

通道启动程序的 SMF 记录

通道启动程序 (CHINIT) 可以生成 SMF 统计信息记录和记帐记录以及有关任务和通道的信息。

CHINIT 可以生成具有以下类型信息的 SMF 统计信息记录和记帐记录:

- 任务: 分派器，适配器，域名服务器 (DNS) 和 SSL。 这些任务构成了所谓的 CHINIT 统计信息。
- 通道: 提供类似于 DIS CHSTATUS 命令提供的记帐信息。 这称为通道记帐。

在 z/OS 以外的平台上， IBM MQ 通过将 PCF 消息写入 SYSTEM.ADMIN.STATISTICS.QUEUE。 有关如何在除 z/OS 以外的平台上记录统计信息的更多信息，请参阅 通道统计信息消息数据 。

统计信息数据

您可以使用此信息来查找以下信息:

- 您是否需要更多的 CHINIT 任务，例如 SSL TCB 的数量以及这些任务使用的 CPU 数量。
- 针对这些任务的请求的平均时间。
- 时间间隔内针对 DNS 和 SSL 任务的最长持续时间请求以及发生此请求的时间。 您可以将此时间与通道可能迂到的问题相关联。

记帐数据

您可以使用此信息来监视通道使用情况并找出以下信息:

- 吞吐量最高的通道。

- 发送消息的速率以及发送数据的速率 (以 MB/ 秒计)。
- 实现的批处理大小。 如果实现的批处理大小接近为通道指定的批处理大小，那么通道可能接近其发送消息的限制。

使用 START TRACE 和 STOP TRACE 命令来控制记帐跟踪和统计信息跟踪的收集。 您可以在通道和队列管理器上使用 STATCHL 和 STATACLS 选项来控制通道是否生成 SMF 数据。

注: 您不必将 OPMODE 设置为 NEWFUNC ，就可以为通道启动程序启用 SMF 记录收集。

术语和任务

使用本主题作为术语以及特定于 IBM MQ for z/OS 的任务的简介。

管理 IBM MQ for z/OS 所需的某些条款和任务特定于 z/OS 平台。 以下列表包含其中一些术语和任务。

- 共享队列
- 页集和缓冲池
- 日志记录
- 定制队列管理器环境
- 重新启动和恢复
- 安全
- 可用性
- 处理对象
- 监控和统计信息
- 应用程序环境

共享队列

队列可以是 非共享队列，由 队列共享组拥有且仅可供一个队列管理器访问，也可以是由队列共享组拥有的共享。 队列共享组由多个队列管理器组成，这些队列管理器在单个 z/OS 综合系统中运行，可以同时访问相同的 IBM MQ 对象定义和消息数据。 在队列共享组中，可共享对象定义存储在共享 Db2 数据库中。 共享队列消息保存在一个或多个耦合设施结构 (CF 结构) 中。 如果消息数据过大而无法直接存储在结构中 (大小超过 63 KB) ，或者如果消息足够大，以至于安装定义的规则选择它进行卸载，那么消息控制信息仍存储在耦合设施条目中，但消息数据会卸载到共享消息数据集 (SMDS) 或共享 Db2 数据库中。 共享消息数据集，共享 Db2 数据库和耦合设施结构是由组中所有队列管理器共同管理的资源。

页集和缓冲池

将消息放入非共享队列时，队列管理器会将数据存储在页集上，以便在后续操作从同一队列获取消息时可以检索数据。 如果从队列中除去消息，那么保存数据的页集中的空间稍后将释放以供复用。 随着队列上保留的消息数增加，因此页集中使用的空间量增加，并且随着队列上的消息数减少，页集中使用的空间也会减少。

为了降低将数据写入页集和从页集读取数据的性能成本，队列管理器会将更新缓冲到处理器存储器中。 用于缓冲页集访问的存储量通过称为 缓冲池的 IBM MQ 对象进行控制。

有关页集和缓冲池的更多信息，请参阅 存储管理。

日志记录

对页集上保留的对象的任何更改以及对持久消息的操作都将记录为日志记录。 这些日志记录将写入称为 活动日志的日志数据集。 活动日志数据集的名称和大小保存在称为 引导数据集 (BSDS) 的数据集中。

当活动日志数据集填满时，队列管理器将切换到另一个日志数据集，以便日志记录可以继续，并将完整活动日志数据集的内容复制到 归档日志 数据集。 有关这些操作的信息 (包括归档日志数据集的名称) 保存在引导

数据集中。 从概念上讲，队列管理器会循环使用一组活动日志数据集; 当填充活动日志时，会将日志数据卸载到归档日志中，并且活动日志数据集可供复用。

有关日志和引导数据集的更多信息，请参阅 第 343 页的『日志记录』。

定制队列管理器环境

启动队列管理器时，将读取一组控制队列管理器操作方式的初始化参数。 此外，将读取包含 IBM MQ 命令的数据集，并执行其包含的命令。 通常，这些数据集包含运行 IBM MQ 所需的系统对象的定义，您可以定制这些数据集以定义或初始化操作环境所需的 IBM MQ 对象。 读取这些数据集后，会将它们定义的任何对象存储在页集或 Db2 中。

有关初始化参数和系统对象的更多信息，请参阅 第 352 页的『定义系统』。

恢复和重新启动

在 IBM MQ 操作期间的任何时候，处理器存储器中可能存在尚未写入页集的更改。 这些更改将写出到队列管理器中的后台任务最近最少使用的页集。

如果队列管理器异常终止，那么队列管理器重新启动的恢复阶段可以恢复丢失的页集更改，因为持久消息数据保存在日志记录中。 这意味着 IBM MQ 可以恢复持久消息数据和对象更改，直到发生故障为止。

如果作为队列共享组成员的队列管理器迁到耦合设施故障，那么仅当您已备份耦合设施结构时，才能恢复该队列上的持久消息。

有关恢复和重新启动的更多信息，请参阅 第 361 页的『恢复和重新启动』。

安全性

您可以使用外部安全性管理器，例如 Security Server (以前称为 RACF) 以保护 IBM MQ 拥有和管理的资源不受未经授权的用户访问。 您还可以使用安全套接字层 (SSL) 来实现通道安全性。 SSL 包含在 IBM MQ 产品中。

有关 IBM MQ 安全性的更多信息，请参阅 第 374 页的『z/OS 上的安全概念』。

可用性

IBM MQ 的一些功能旨在提高队列管理器或通信子系统发生故障时的系统可用性。 有关这些功能部件的更多信息，请参阅第 379 页的『可用性』。

处理对象

当队列管理器正在运行时，可以通过 z/OS 控制台界面或通过使用 TSO 下的 ISPF 服务的管理实用程序来处理 IBM MQ 对象。 这两种机制都使您能够定义，变更或删除 IBM MQ 对象。 您还可以控制和显示各种 IBM MQ 和队列管理器功能的状态。

您还可以使用 IBM MQ 版本 7 资源管理器 (图形用户界面) 来处理 IBM MQ 对象，该图形用户界面提供了使用队列，队列管理器和其他对象的可视方式。

有关这些工具的更多信息，请参阅 发出命令。

监控和统计信息

有多个工具可用于监视队列管理器和通道启动程序。 您还可以收集统计信息以用于性能评估和记帐目的。

有关这些设施的更多信息，请参阅 第 382 页的『监控和统计信息』。

应用程序环境

当队列管理器已启动时，应用程序可以连接到该队列管理器并使用 IBM MQ API 启动该队列管理器。 这些应用程序可以是 CICS， IMS，批处理或 WebSphere Application Server 应用程序。 IBM MQ 应用程序还可以使用 CICS 和 IMS 网桥访问 CICS 和 IMS 系统上不知道 IBM MQ 的应用程序。

有关这些设施的更多信息，请参阅 第 385 页的『IBM MQ 和其他 z/OS 产品』。

有关编写 IBM MQ 应用程序的信息，请参阅以下文档:

- 开发应用程序
- 使用 C++
- 使用 IBM MQ classes for Java

共享队列和队列共享组

您可以使用共享队列和队列共享组来实现 MQ 资源的高可用性。

共享队列和队列共享组是 z/OS 平台上 IBM MQ for z/OS 独有的功能。

本部分描述了属性和优点，并提供了有关多个队列管理器如何共享相同队列以及这些队列上的消息的信息。

相关概念

第 291 页的『什么是共享队列?』
共享队列 是一种本地队列类型。 该队列上的消息可由综合系统中的一个或多个队列管理器访问。

第 292 页的『什么是队列共享组?』
可以访问相同共享队列的一组队列管理器称为队列共享组。 队列共享组的每个成员对同一组共享队列都具有访问权。

第 293 页的『共享队列消息在何处挂起?』
共享队列上的每条消息都由 z/OS 耦合设施列表结构中的一个条目表示。 如果消息数据过大而无法容纳在同一条目中，那么会将其卸载到共享消息数据集 (SMDS) 或 Db2。

第 306 页的『使用共享队列的优点』
共享队列允许 IBM MQ 应用程序可伸缩，高度可用，并允许实现工作负载均衡。

第 322 页的『分布式排队和队列共享组』
分布式排队和队列共享组是可用于提高应用程序系统可用性的两种方法。 使用本主题来查找有关这些方法的更多信息。

第 326 页的『使用共享队列影响工作负载分布』
使用本主题来了解影响队列共享组中共享队列的工作负载分布的因素。

相关参考

第 327 页的『在何处查找有关这些概念的更多信息』
使用本主题中的表来查找有关本信息中心中其他主题的更多信息。

什么是共享队列?

共享队列 是一种本地队列类型。 该队列上的消息可由综合系统中的一个或多个队列管理器访问。

队列共享组

可以访问同一组共享队列的队列管理器构成一个称为 队列共享组的组。

任何队列管理器都可以访问消息

队列共享组中的任何队列管理器都可以访问共享队列。 这意味着您可以将消息放入一个队列管理器上的共享队列，并从另一个队列管理器中获取同一消息。 这为队列共享组中不需要队列管理器之间的通道处于活动状态的通信提供了快速机制。

IBM WebSphere MQ 7.1 和更高版本支持将消息卸载到 Db2 或共享消息数据集 (SMDS)。 可配置任何大小的消息的卸载。

在较早版本的 IBM MQ 中，大型消息 (> 63 KB) 将占位符存储在耦合设施 (4 K) 中，其消息数据存储在 Db2 中。

第 292 页的图 60 显示了三个队列管理器和一个耦合设施，构成队列共享组。 所有三个队列管理器都可以访问耦合设施中的共享队列。

应用程序可以连接到队列共享组中的任何队列管理器。 由于队列共享组中的所有队列管理器都可以访问所有共享队列，因此应用程序不依赖于特定队列管理器的可用性; 队列共享组中的任何队列管理器都可以为队列提供服务。

这提供了更大的可用性，因为队列共享组中的所有其他队列管理器都可以在其中一个队列管理器出现问题时继续处理该队列。

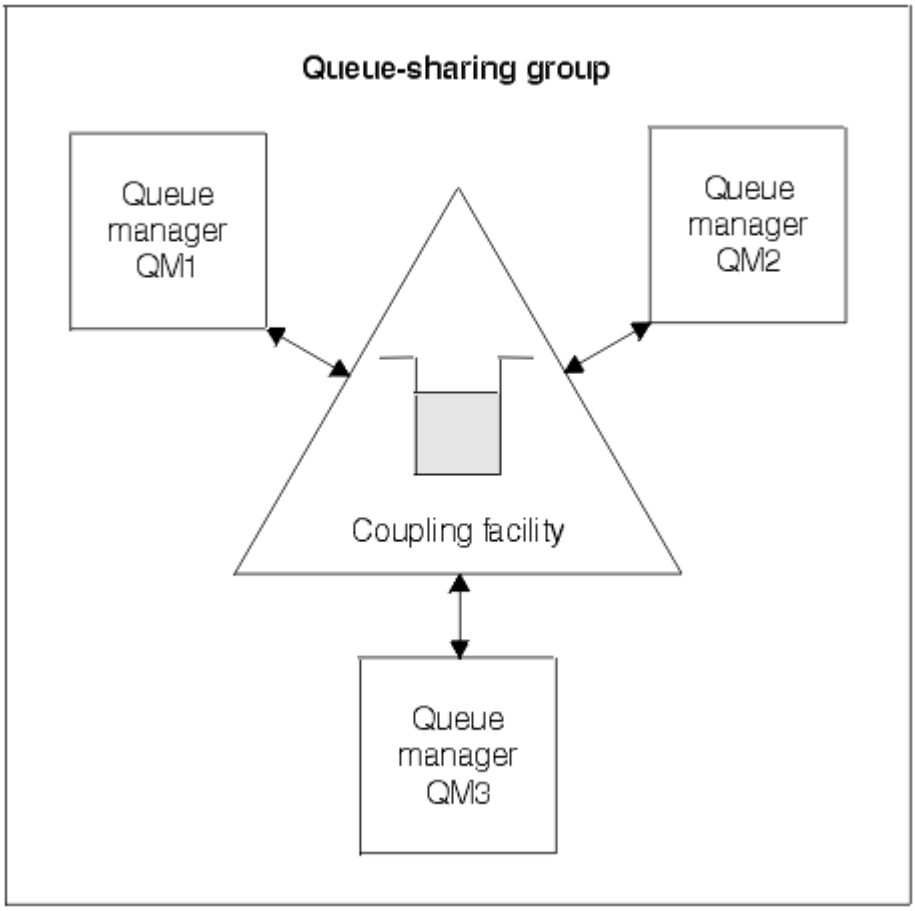


图 *60:* 队列共享组

队列定义由所有队列管理器共享

共享队列定义存储在 Db2 数据库表 OBJ_B_QUEUE 中。 因此，您只需要定义一次队列，然后队列共享组中的所有队列管理器都可以访问该队列。 这意味着要创建的定义较少。

相反，非共享队列的定义存储在拥有该队列的队列管理器的页集 0 上 (如 页集 中所述)。

如果已在定义队列管理器的页集中定义了具有该名称的队列，那么不能定义共享队列。 同样，如果存在同名的共享队列，那么不能在队列管理器页集上定义队列的本地版本。

什么是队列共享组?

可以访问相同共享队列的一组队列管理器称为队列共享组。 队列共享组的每个成员对同一组共享队列都具有访问权。

队列共享组具有最多为四个字符的名称。 该名称在网络中必须是唯一的，并且必须与所有队列管理器名称不同。

第 293 页的图 61 说明了包含两个队列管理器的队列共享组。 每个队列管理器都有一个通道启动程序及其自己的本地页集和日志数据集。

队列共享组的每个成员还必须连接到 Db2 系统。 Db2 系统必须全部位于同一个 Db2 数据共享组中，以便队列管理器可以访问用于保存共享对象定义的 Db2 共享存储库。 这些定义是任何类型的 IBM MQ 对象 (例如，队列和通道) 的定义，仅定义一次，然后组中的任何队列管理器都可以使用这些定义。 这些定义称为 全局 定义，在 专用定义和全局定义中进行了描述。

多个队列共享组可以引用特定数据共享组。 您可以指定 Db2 子系统的名称以及队列管理器在启动时在 IBM MQ 系统参数中使用的数据共享组。

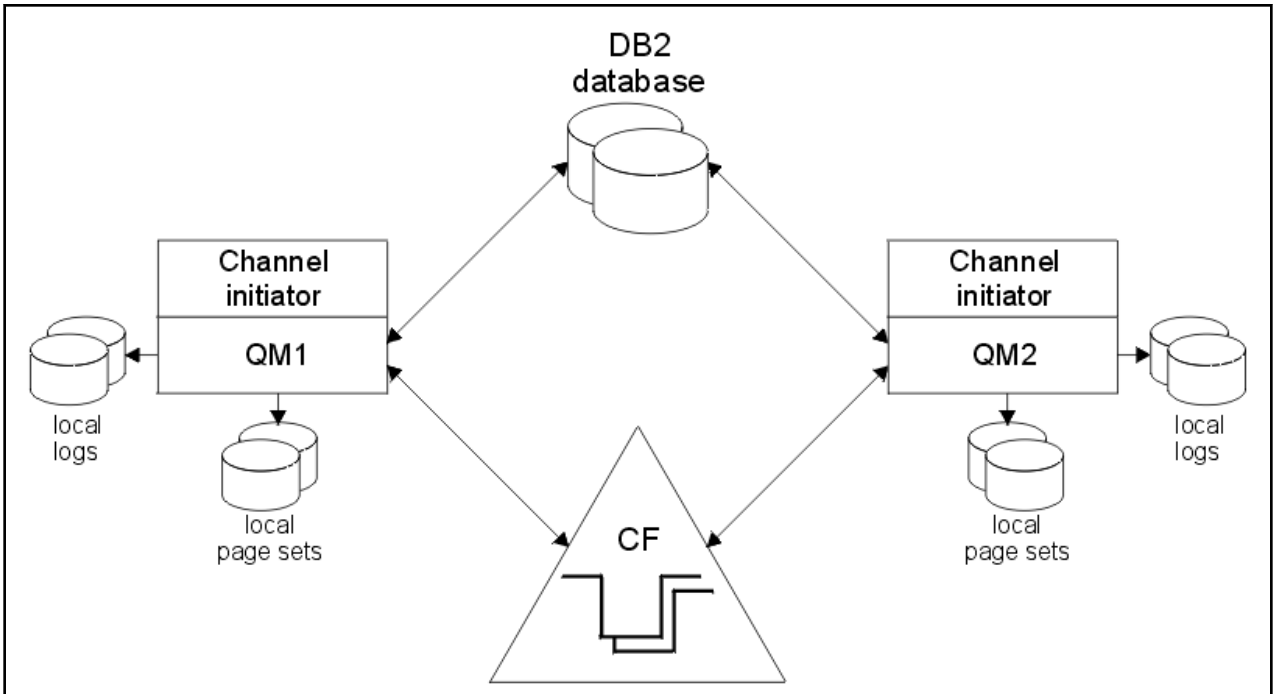


图 *61:* 队列共享组中队列管理器的组件

当队列管理器已加入队列共享组时，它有权访问为该组定义的共享对象，并且您可以使用该队列管理器在该组中定义新的共享对象。 如果在组中定义了共享队列，那么可以使用此队列管理器将消息放入这些共享队列并从中获取消息。 组中的任何队列管理器都可以检索共享队列上保留的消息。

您可以输入一次 MQSC 命令，并使其在队列共享组中的所有队列管理器上执行，就像在每个队列管理器上分别输入一样。 *command scope* 属性用于此目的。 将命令定向到不同的队列管理器中描述了此属性。

当队列管理器作为队列共享组的成员运行时，必须能够区分专用于该队列管理器的 IBM MQ 对象和全局定义的可供队列共享组中所有队列管理器使用的 IBM MQ 对象。 队列共享组处置 属性用于此目的。 此属性在 专用和全局定义中进行了描述。

您可以定义一组安全概要文件，用于控制对组中任何位置的 IBM MQ 对象的访问。 这意味着要定义的概要文件数量大大减少。

队列管理器只能属于一个队列共享组，并且该组中的所有队列管理器都必须位于同一综合系统中。 您可以在启动时在系统参数中指定队列管理器所属的队列共享组。

共享队列消息在何处挂起?

共享队列上的每条消息都由 z/OS 耦合设施列表结构中的一个条目表示。 如果消息数据过大而无法容纳在同一条目中，那么会将其卸载到共享消息数据集 (SMDS) 或 Db2。

如果已将 CF 结构配置为使用系统类内存 (SCM)，那么 IBM MQ 可以在不进行其他配置的情况下使用此结构。 请参阅 IBM MQ V8 功能部件和增强功能，第 8 章。

共享队列消息存储器

放入共享队列的消息不会存储在页集上，也不会使用缓冲池。

共享队列中的消息在 z/OS 耦合设施 (CF) 中的列表结构上具有条目。 同一综合系统中的许多队列管理器可以使用 CF 列表结构来访问这些消息。

小型共享队列消息的消息数据通常包含在耦合设施条目中。 对于较大的消息，消息数据可以存储在共享消息数据集 (SMDS) 中，也可以作为一个或多个二进制大对象 (BLOB) 存储在由 Db2 数据共享组共享的 Db2 表中。 超过 63 KB 的消息数据始终会卸载到 SMDS 或 Db2。 还可以选择以相同方式卸载较小的消息，以节省耦合设施结构中的空间。 请参阅第 295 页的『为共享消息指定卸载选项』，以获取更多详细信息。

放置在共享队列上的消息将在耦合设施结构中引用，直到 MQGET 检索到这些消息为止。 耦合设施操作用于:

- 搜索下一条可检索的消息
- 锁定共享队列上未落实的消息
- 通知相关队列管理器已落实消息的到达

对持久消息执行的 MQPUT 和 MQGET 操作记录在执行该操作的队列管理器的日志中。 这将最大程度地降低耦合设施发生故障时数据丢失的风险。

耦合设施

共享队列上保存的消息在耦合设施中引用。 耦合设施位于综合系统中的任何 z/OS 映像外部，通常配置为在不同的电源上运行。 因此，耦合设施对软件故障具有弹性，您可以对其进行配置，以使其对硬件故障或电源中断具有弹性。 这意味着存储在耦合设施中的消息高度可用。

IBM MQ 使用的每个耦合设施列表结构都专用于特定队列共享组，但一个耦合设施可以保存多个队列共享组的结构。 不同队列共享组中的队列管理器无法共享数据。 队列共享组中最多 32 个队列管理器可以同时连接到耦合设施列表结构。

单个耦合设施列表结构最多可包含 512 个共享队列。 存储在结构中的消息数据总量受结构容量的限制。 但是，通过 **CFLEVEL(5)**，您可以使用卸载参数来卸载小于 63 KB 的消息的数据，从而增加可以存储在结构中的消息数，尽管每条消息仍需要至少一个耦合设施条目加上至少 512 字节的数据。

列表结构的大小受以下因素限制:

- 它必须位于单个耦合设施中。
- 它可能与 IBM MQ 和其他产品的其他结构共享可用的耦合设施存储器。

耦合设施列表结构可以具有与其关联的存储类内存。 在某些情况下，当与共享队列配合使用时，此存储类内存很有用。 请参阅第 307 页的『将存储类内存与共享队列配合使用』以获取更多信息。

规划 CF 结构大小

如果需要有关 CF 结构大小调整的指导，那么可以使用 MP16: IBM MQ for z/OS 容量规划和调整 supportpac。 您还可以使用基于 Web 的工具 CFSizer，该工具由 IBM 提供以帮助处理 CF 大小。

CF 结构对象

队列管理器对耦合设施结构的使用在 CF 结构 (CFSTRUCT) IBM MQ 对象中指定。

这些结构对象存储在 Db2 中。

使用与耦合设施结构相关的 z/OS 命令或定义时，需要队列共享组名称的前四个字符。 但是， IBM MQ CFSTRUCT 对象始终存在于单个队列共享组中，因此其名称不包含队列共享组名称的前四个字符。 例如，从 SQ03 开始在队列共享组中定义的 CFSTRUCT (MYDATA) 将使用耦合设施列表结构 SQ03MYDATA。

CF 结构具有用于确定其功能能力的 CFLEVEL 属性:

- 1 或 2-可用于小于 63 KB 的非持久消息
- 3-可用于小于 63 KB 的持久和非持久消息
- 4-可用于最大为 100 MB 的持久和非持久消息
- 5-可用于最大为 100 MB 的持久和非持久消息，并选择性地卸载到共享消息数据集 (SMDS) 或 Db2。

耦合设施的备份和恢复

可以使用 IBM MQ 命令 BACKUP CFSTRUCT 来备份耦合设施列表结构。 这会将当前在 CF 结构中的持久消息的副本放入进行备份的队列管理器的活动日志数据集中，并将备份的记录写入 Db2。

如果耦合设施发生故障，那么可以使用 IBM MQ 命令 RECOVER CFSTRUCT。 这将使用来自 Db2 的备份记录从 CF 结构的备份中查找和复原持久消息。 将使用队列共享组中所有队列管理器的日志来重放自上次备份以来的任何活动，然后将 CF 结构恢复到故障之前的位置。

请参阅 BACKUP CFSTRUCT 和 RECOVER CFSTRUCT 命令以获取更多详细信息。

相关概念

第 295 页的『为共享消息指定卸载选项』
您可以选择共享队列消息的消息数据存储在 Db2 表或共享消息数据集 (SMDS) 中的位置。 您还可以根据消息的大小以及耦合设施结构 (CF) 的当前使用情况来选择卸载哪些消息。

第 297 页的『管理共享消息数据集 (SMDS) 环境』
如果选择共享消息数据集以卸载大型消息，那么还必须了解 IBM MQ 用于管理这些数据集的信息以及用于处理此信息的命令。 使用本主题来了解如何管理共享消息数据集。

为共享消息指定卸载选项

您可以选择共享队列消息的消息数据存储在 Db2 表或共享消息数据集 (SMDS) 中的位置。 您还可以根据消息的大小以及耦合设施结构 (CF) 的当前使用情况来选择卸载哪些消息。

可以从耦合设施中卸载共享队列的消息数据，并将其存储在 Db2 表或称为 共享消息数据集 (SMDS) 的 IBM MQ 受管数据集中。

对于大于耦合设施条目大小 63 KB 的消息，将消息数据卸载到 SMDS 可能比卸载到 Db2 具有显着的性能改进。

仍使用耦合设施结构中的列表条目来管理每个共享队列消息，但当将消息数据卸载到 SMDS 时，该耦合设施条目仅包含一些控制信息以及对存储消息的相关磁盘块的引用列表。 使用此机制意味着每条消息所需的耦合设施元素存储量仅为消息的实际大小的一小部分。

选择共享队列消息的存储位置

通过 **CFSTRUCT** 定义上的 **OFFLOAD(SMDS|DB2)** 参数来控制 SMDS 或 Db2 共享消息存储器的选择。 **OFFLOAD(SMDS)** 是缺省值。

此参数还要求 **CFSTRUCT** 使用 **CFLEVEL(5)** 或更高版本。 只有 IBM WebSphere MQ 7.1 或更高版本的队列管理器才能连接到此级别的 CF 结构。

仅当队列共享组中的所有队列管理器都处于 IBM WebSphere MQ 7.1 或更高版本时，才可以将结构更改为 **CFLEVEL(5)** 。

OFFLOAD 参数仅在 **CFLEVEL(5)** 中有效。 请参阅 DEFINE CFSTRUCT 以获取更多详细信息。

OFFLOAD(DB2) 主要用于迁移目的。

选择卸载哪些共享队列消息

根据消息数据的大小以及耦合设施结构的当前使用情况，将消息数据卸载到 SMDS 或 Db2 。 有三个规则，每个规则指定一对匹配的参数。 这些参数是对应的耦合设施结构使用率阈值百分比 (**OFFLDnTH**) 和消息大小限制 (**OFFLDnSZ**)。

这三个规则的当前实现是使用以下关键字对指定的:

- OFFLD1TH 和 OFFLD1SZ
- OFFLD2TH 和 OFFLD2SZ
- OFFLD3TH 和 OFFLD3SZ

规则对	缺省值	描述
规则对 1	OFFLD1TH(70) 和 OFFLD1SZ(32K)	如果耦合设施结构针对超过 32 KB 的消息超过 70% 的完全卸载数据
规则对 2	OFFLD2TH(80) 和 OFFLD2SZ(4K)	如果耦合设施结构对超过 4 KB 的消息的满卸载数据超过 80%
规则对 3	OFFLD3TH(90) 和 OFFLD3SZ(0K)	如果耦合设施结构对超过 0 KB (所有消息) 的消息的完全卸载数据超过 90%

如果卸载规则具有 OFFLD *x* SZ 值 64K，那么表示该规则无效。 在这种情况下，仅当另一个卸载规则生效时，或者如果消息大于 63.75 KB，那么将卸载消息，太大而无法存储在结构中。

卸载的每条消息仍需要耦合设施中的 0.75 KB 存储空间。

可以为每个结构指定的三个卸载规则旨在如下所示使用。

- 性能
  - 当应用程序结构中有大量空间时，仅当消息数据太大而无法存储在结构中时，或者如果它超过某个较低的消息大小阈值，那么应该卸载消息数据，以便将其存储在结构中的性能值不值得其需要的结构空间量。
  - 如果需要特定消息大小阈值，那么通常使用第一个卸载规则来指定该阈值。
- 容量
  - 当应用程序结构中的空间非常小时，应该卸载最大数量的消息数据，以便最佳地利用剩余空间。
  - 第三个卸载规则通常用于指示当结构接近已满时，应该卸载大多数消息，因此应用程序结构中的条目通常具有最小大小 (大约需要 0.75K 字节)。
  - 应该根据应用程序结构大小和最大预期任务列表数来选择使用阈值参数。 例如，如果最大预期待办事项是 1M 消息，那么此数量的消息所需的结构存储量约为 0.75G 字节。 这意味着例如，如果结构约为 10G 字节，那么卸载所有消息的使用阈值必须设置为 92% 或更低。
  - 结构空间分为元素和条目，尽管总体可能有足够的空间，但其中一个空间可能先于另一个空间耗尽。 系统提供 AUTOALTER 功能以在必要时调整比率，但这不是很敏感，因此实际可用的空间量可能略小一些。 因此，最好以不超过最大结构空间的 90% 为目标，因此在前一个示例中，用于卸载所有消息的使用阈值最好设置为 80% 左右。
- 缓冲转换:
  - 随着耦合设施结构中剩余空间量的减少，性能特性发生较大突然变化是不可取的。 耦合设施管理在所使用的条目与元素的典型比率中突然发生阈值更改也是不可取的。
  - 第二个卸载规则通常用于在性能和容量偏差卸载规则之间提供一些中间缓冲。 当耦合设施结构中使用的空间超过中间阈值时，可以将其设置为导致卸载活动显着增加。 这意味着剩余空间用得更慢，给耦合设施自动改变处理更多时间，以适应更高的使用水平。

如果不能扩展耦合设施结构，并且需要存储至少一些预定数量的消息，那么可以根据需要修改第三规则，以确保所有消息的数据卸载在适当的阈值处开始，以确保为该预定数量的消息保留空间。

例如，如果耦合设施结构大小为 4 GB，并且预先确定的消息数为 1 百万，那么需要 1,000,000 * 0.75 KB，即 768 MB，即 4 GB 的 18.75%。 在这种情况下，需要将卸载所有消息的阈值设置为 80% 左右，而不是 90%。 这将提供参数 OFFLD3TH(80) 和 OFFLD3SZ(0K)。 其他卸载参数也需要调整。

如果发现卸载非常小的消息具有显着的性能影响，但对于较大的消息，相对影响较小，那么可以降低其他规则的使用阈值，以更早卸载更大的消息，从而在结构中为需要卸载之前的小消息留出更多空间。

例如，如果频繁出现超过 32KB 的消息，但卸载这些消息所耗用的时间性能 (根据 RMF 统计信息或应用程序性能确定) 与将它们保留在耦合设施中所耗用的时间性能非常相似，那么第一个规则的阈值可设置为 0% 以卸载所有此类消息。 这将提供参数 OFFLD1TH(0) 和 OFFLD1SZ(32K)。 同样需要调整其他卸载参数。

如果存在许多围绕特定中间大小的消息 (例如 16 KB 和 6 KB)，那么更改第二个规则的消息大小选项可能有用，以便在相当低的使用阈值下卸载较大的消息，从而节省大量空间，但较小的消息仍然只存储在耦合设施中。

管理共享消息数据集 *(SMDS)* 环境

如果选择共享消息数据集以卸载大型消息，那么还必须了解 IBM MQ 用于管理这些数据集的信息以及用于处理此信息的命令。 使用本主题来了解如何管理共享消息数据集。

SMDS 对象

在可通过队列共享组中的任何队列管理器更新的共享 SMDS 对象中跟踪每个共享消息数据集的属性和状态。

对于可访问每个耦合设施应用程序结构的每个队列管理器，都有一个共享消息数据集。 共享消息数据集由拥有队列管理器名称 (使用 SMDS 关键字指定) 和应用程序结构名称 (使用 CFSTRUCT 关键字指定) 标识。

注: 为结构定义 SMDS 数据集时，每个队列管理器必须有一个数据集。

SMDS 对象存储在数组中 (组中每个队列管理器有一个条目)，该数组构成 Db2 中存储的相应 CFSTRUCT 对象的扩展。

没有用于 DEFINE 或 DELETE SMDS 对象的命令，因为它是作为 CFSTRUCT 对象的一部分创建或删除的，但有一个用于 ALTER 它以更改单个拥有队列管理器的设置的命令。

有关 SMDS 命令的更多信息，请参阅 第 305 页的『SMDS 相关命令』

SMDSCONN 信息

共享消息数据集可能处于正常状态，但一个或多个队列管理器无法连接到该数据集，例如，由于安全性定义或直接访问设备连接存在问题。 因此，每个队列管理器都必须跟踪每个共享消息数据集的连接状态和可用性信息，例如，指示当前是否可以连接到该数据集，如果不是，那么说明原因。

SSMDSCONN 信息表示与共享消息数据集的队列管理器连接。 对于共享消息数据集本身，它由队列管理器标识，该队列管理器拥有与 CFSTRUCT 名称组合的共享消息数据集 (在共享对象本身的 SMDS 关键字上指定)。

没有用于标识连接队列管理器的参数，因为针对特定队列管理器的命令只能引用同一队列管理器的 SMDSCONN 信息。

SSMDSCONN 信息条目保留在拥有队列管理器的主存储器中，并在重新启动队列管理器时重新创建。 但是，如果来自单个队列管理器的连接已显式停止，那么此信息也会作为标志存储在相应的 CFSTRUCT 或 SMDS 对象中的连接数组中，以便它在队列管理器重新启动期间持久存在。

状态和可用性信息

状态信息指示资源或连接的状态 (例如，是否尚未使用，正在正常使用或需要恢复)。 通常使用 STATUS 关键字进行描述。 可能的值取决于对象的类型。

通常会自动更新状态信息，例如，在使用资源或连接时检测到错误时。 但是，在某些情况下，还可以使用命令来更新状态，以允许队列管理器无法自动确定正确状态的情况。

可用性信息指示是否可以使用资源或连接，通常主要由状态信息确定。 对于共享消息数据集支持中使用的资源或连接类型，将实现三个级别的可用性:

可用

这意味着该资源可供正常使用。 这并不一定意味着它目前正在使用中 (可以根据 STATUS 值来确定)。 对于数据集，如果它需要重新启动处理，那么这将允许拥有队列管理器打开该数据集，但其他队列管理器必须等到该数据集重新处于 ACTIVE 状态。

由于错误而不可用
这意味着由于错误，资源已自动变为不可用，并且在执行某种形式的修复或恢复处理之前，预期不会再次可用。 但是，允许在没有操作员干预的情况下再次尝试使其可用。 此尝试也可以由用于将资源标记为 "已启用" 的命令或用于以指示恢复处理已完成的方式更改状态的命令触发。

使资源不可用的原因通常从相关 STATUS 值可见，但在某些情况下可能有其他原因使资源不可用，在这种情况下，会提供单独的 REASON 值来指示原因。

由于操作员命令而不可用
这意味着命令已显式禁用对资源的访问。 只能通过使用命令再次将其启用来使其可用。

SMDS 可用性

对于共享 SMDS 对象，可用性由 ACCESS 关键字描述，可能的值为 ENABLED， SUSPENDED 和 DISABLED。

可以对组中的任何队列管理器中的相关共享对象使用 **RESET SMDS** 命令来更新可用性，以设置 ACCESS (ENABLED) 或 ACCESS (DISABLED)。

如果可用性先前为 ACCESS (SUSPENDED)，那么将其更改为 ACCESS (ENABLED) 将触发新的尝试使用共享消息数据集，但如果先前的错误仍然存在，那么可用性将重置回 ACCESS (SUSPENDED)。

SMDSCONN 可用性

对于本地 SMDSCONN 信息条目，可用性由 AVAIL 关键字描述，可能的值为 NORMAL， ERROR 或 STOPPED。 可以使用针对特定队列管理器的 **START SMDSCONN** 或 **STOP SMDSCONN** 命令来更新可用性，以启用或禁用其连接。

如果可用性先前为 AVAIL (ERROR)，那么将其更改为 AVAIL (NORMAL) 将触发新的尝试使用共享消息数据集，但如果先前的错误仍然存在，那么可用性将重置为 AVAIL (ERROR)。

共享消息数据集共享状态和可用性

使用共享状态信息在组中管理每个共享消息数据集的可用性，可使用带有 TYPE (SMDS) 的 **DISPLAY CFSTATUS** 命令来显示这些信息。 这将显示已为每个结构激活数据集的每个队列管理器的状态信息。 每个数据集可以处于下列其中一种状态:

找不到
这意味着尚未激活相应的数据集。 仅当指定了特定队列管理器时，才会显示此状态，因为当选择了所有队列管理器时，将跳过尚未激活的数据集。

新建
正在首次打开并初始化数据集，准备使其处于活动状态。

活动的
这意味着数据集完全可用，应该由结构的所有活动队列管理器分配和打开。

失败
这意味着数据集完全不可用 (恢复处理除外)，并且必须由所有队列管理器关闭和取消分配。

恢复中
这意味着正在对此数据集进行介质恢复 (使用 RECOVER CFSTRUCT)。

已恢复
这指示已发出命令以将失败的数据集切换回活动状态，但需要进一步重新启动处理 (尚未完成)，因此只有拥有此数据集的队列管理器才能打开此数据集以进行重新启动处理。

空
数据集不包含任何消息。 如果拥有队列管理器在数据集不包含任何消息时正常关闭该数据集，那么该数据集将处于此状态。 当要废弃先前数据集内容时，也可以将其置于 EMPTY 状态，因为应用程序结构已清空 (使用带有 TYPE PURGE 的 **RECOVER CFSTRUCT**，或者仅对于不可恢复的结构，通过删除该结构的先前实例)。 下次数据集由其拥有的队列管理器打开时，空间映射将重置为空，并且状态将更改为 ACTIVE。 由于不再需要先前的数据集内容，因此可以将处于此状态的数据集替换为新分配的数据集，例如，更改空间分配或将其移动到另一个卷。

命令输出包括启用恢复日志记录的日期和时间 (如果有) 以及数据集失败的日期和时间 (如果当前未处于活动状态)。

可以通过 **RESET SMDS** 命令将共享消息数据集置于 FAILED 状态，也可以在检测到以下任何类型的错误时自动将其置于 FAILED 状态:

- 拥有的队列管理器无法分配或打开数据集。
- 在任何队列管理器成功打开数据集头之后，验证该数据集头失败。
- 当拥有队列管理器正在读取或写入数据时，将发生永久 I/O 错误。
- 当另一个队列管理器从已成功完成打开处理和验证的数据集中读取数据时，将发生永久 I/O 错误。

当数据集处于 FAILED 或 INRECOVER 状态时，它不可用于正常使用，因此如果可用性状态为 ACCESS (ENABLED)，那么它将更改为 ACCESS (SUSPENDED)。

如果数据集已进入 FAILED 状态，但不需要介质恢复 (例如，由于数据仍然有效但存储设备暂时脱机)，那么可以使用 **RESET SMDS** 命令来请求将状态直接更改为 RECOVERY 状态。

当数据集在完成恢复处理时或由于 **RESET SMDS** 命令而进入 RECOVERY 状态时，那么在完成重新启动处理后，即可再次使用该数据集。 如果它处于 ACCESS (SUSPENDED) 状态，那么它将自动切换回 ACCESS (ENABLED) 状态，这允许拥有队列管理器执行重新启动处理。 重新启动处理完成时，状态将更改为 ACTIVE，然后所有其他队列管理器都可以再次连接到数据集。

共享消息数据集连接状态和可用性

每个队列管理器维护其与自身和组中其他队列管理器所拥有的每个共享消息数据集的连接的本地状态和可用性信息。 可以使用 **DISPLAY SMDSCONN** 命令显示此信息。

如果它无法访问处于 ACTIVE 状态的共享消息数据集 (属于另一个队列管理器)，那么它会从其自己的角度将连接标记为不可用。

如果该错误明确指示数据集本身存在问题，那么队列管理器还会自动更改共享状态以指示数据集现在处于 FAILED 状态。 但是，如果该错误可能是由环境问题 (例如，未授权打开数据集) 引起的，那么队列管理器会发出错误消息并将数据集视为不可用，但不会修改共享数据集状态。 如果环境错误仍然是数据集的问题 (例如，它已分配在某些队列管理器无法访问的设备上)，那么操作员可以使用指定 STATUS (FAILED) 的 RESET SMDS 命令来允许根据需要恢复或修复数据集。

如果无法建立与共享消息数据集的连接，但该数据集似乎有效，那么可以通过对拥有的队列管理器发出 **START SMDSCONN** 命令来触发使用该数据集的新尝试。

如果操作需要临时终止特定队列管理器与数据集之间的连接，但数据集本身未损坏，那么可以使用 **STOP SMDSCONN** 命令来关闭和取消分配数据集。 如果数据集正在使用中，那么队列管理器将正常关闭该数据集 (尽管该数据集中的任何数据请求都将被拒绝，并带有返回码)。 如果它是拥有的数据集，那么队列管理器将在 CLOSE 处理期间保存空间映射，从而避免需要重新启动处理。

如果需要从所有队列管理器中临时取出数据集 (例如移动该数据集) 但未损坏，那么最好将 **STOP SMDSCONN** 用于具有选项 CMDSCOPE (*) 的相关数据集，以首先停止使用该数据集的队列管理器，因为这将避免在数据集重新投入服务时需要重新启动处理。 相反，如果将数据集标记为 FAILED，那么这将告知队列管理器必须立即停止使用该数据集，这意味着不会保存空间映射，并且需要通过重新启动处理重建空间映射。

如果重新启动队列管理器，那么将重试对先前处于 ACCESS (SUSPENDED) 状态的任何共享消息数据集的访问。

共享消息数据集恢复日志记录

记录持久共享消息以进行介质恢复。 这意味着可以在耦合设施结构或共享消息数据集发生任何故障后恢复消息，前提是恢复日志仍然完整。 出于灾难恢复目的，还可以从另一个站点的恢复日志中重新创建持久消息。

将消息数据写入共享消息数据集时，将单独记录写入数据集的每个块，后跟写入耦合设施的消息条目 (包括数据映射)。 恢复过程始终恢复耦合设施结构，但不需要恢复个别共享消息数据集，除非数据集状态为 FAILED，或者状态为 ACTIVE，但数据集头记录不再有效，这指示数据集已重新创建。 如果数据集的状态为 ACTIVE 并且数据集头仍然有效，或者如果数据集的状态为 EMPTY，那么不会选择恢复数据集，这指示发生故障时未在其中存储任何消息。

共享消息数据集备份

当使用 BACKUP CFSTRUCT 来备份应用程序结构中的共享消息时，将同时备份存储在共享消息数据集中的持久消息的任何数据，与先前存储在数据库中的持久共享消息一样。

共享消息数据集恢复

如果共享消息数据集已损坏或丢失，那么需要将其置于 FAILED 状态以阻止队列管理器使用该数据集，直到修复该数据集为止。 这通常会自动发生，但也可以使用指定 STATUS (FAILED) 的 **RESET SMDS** 命令来完成。

如果共享消息数据集包含任何持久消息，那么可以使用 RECOVER CFSTRUCT 命令来恢复这些消息。 此命令首先从最近的 BACKUP CFSTRUCT 命令复原该共享消息数据集的任何持久消息数据，然后应用自该时间以来记录的所有更改。 如果自首次激活数据集以来未执行任何 **BACKUP CFSTRUCT** 命令，那么会将其重置为空，然后应用自激活以来的所有更改。

如果 CFSTRUCT 内容和所有共享消息数据集都不可用 (例如，在灾难恢复情况下) ，那么它们都可以通过单个 **RECOVER CFSTRUCT** 命令进行恢复。

如果共享消息数据集已损坏，但 CFSTRUCT 的恢复未处于活动状态，或者包含最新 BACKUP CFSTRUCT 的日志不可用或不可用，那么无法恢复卸载到该数据集的消息。 在这种情况下，可以使用带有参数 TYPE (PURGE) 的 **RECOVER CFSTRUCT** 命令将共享消息数据集标记为空，并从该数据集中存储数据的结构中删除任何消息。

发出 **RECOVER CFSTRUCT** 命令时，共享消息数据集状态将从 FAILED 更改为 INRECOVER。 如果恢复成功完成，那么状态会自动更改为 RECOVERY ，否则会更改回 FAILED。

当数据集更改为恢复状态时，这将告知拥有队列管理器它现在可以尝试打开数据集并执行重新启动处理。

共享消息数据集恢复和同步点

共享消息数据集恢复过程将重新应用所有完整日志记录的更改，直到日志结束，而不考虑同步点。

如果在同步点内进行了更改，那么针对 CFSTRUCT 的重新启动或恢复处理可能会导致回退未落实的请求，因此某些已恢复的更改可能实际未使用，但无论如何都不会对恢复这些更改造成任何影响。

还可能是未落实的 MQPUT 消息可能已写入结构，但相应数据可能未写入数据集或日志 (因为只有在同步点处理开始时才会强制完成 I/O)。 这是无害的，因为重新启动处理将回退结构中的消息条目，因此它引用未恢复的数据这一事实无关紧要。

共享消息数据集重新启动处理

如果与 CFSTRUCT 的队列管理器连接正常终止，那么在数据集关闭之前，队列管理器会将每个共享消息数据集的可用块空间映射写入数据集内的检查点区域。 然后，可以在连接重新启动时再次读取空间映射，前提是 CFSTRUCT 和共享消息数据集都不需要在下一次重新启动之前进行任何恢复处理。

但是，如果队列管理器异常终止，或者结构或数据集需要任何恢复处理，那么在重新启动与该结构的队列管理器连接时，需要执行其他处理以动态重建空间映射。

如果数据集本身不需要恢复，那么队列管理器重新启动只需扫描结构的当前内容以查找对当前队列管理器拥有的消息数据的引用，并将相关数据块标记为空间映射中拥有的数据块。 在重建空间映射时，其他队列管理器可以继续使用该结构并读取重新启动队列管理器所拥有的数据。

恢复后重新启动共享消息数据集

如果必须从备份恢复共享消息数据集，那么存储在数据集中的所有非持久消息都将丢失，如果使用 TYPE (PURGE) 恢复数据集，那么存储在数据集中的所有消息都将丢失。 在恢复完成之前，数据集将标记为 FAILED 或 INRECOVER ，因此任何尝试从另一个队列管理器读取其中一条受影响的消息都会返回错误代码，指示数据集暂时不可用。

当数据集已恢复时，状态将更改为已恢复，这允许拥有的队列管理器打开该数据集以进行重新启动处理，但该数据集仍对其他队列管理器不可用。 队列管理器重新启动会扫描结构以重新构建空间映射以获取任何剩余

消息。 扫描还会检查已丢失数据的消息，并从结构中删除这些消息 (如果需要，将其标记为 "已丢失" ，稍后将其删除)。

当此重新启动扫描完成时，数据集状态将自动从 "已恢复" 更改为 "活动" ，此时其他队列管理器可以重新开始使用该数据集。

共享消息数据集使用信息

DISPLAY USAGE 命令现在还显示有关任何当前打开的共享消息数据集的共享消息数据集空间和缓冲池使用情况的信息。 如果指定了新选项 TYPE (SMDS) 或现有选项 TYPE (ALL) ，那么将显示此信息。

共享消息数据性能和容量注意事项

监视数据集使用情况

DISPLAY USAGE 命令可以使用选项 **TYPE(SMDS)** 显示每个拥有的共享消息数据集的当前已满百分比。

当共享消息数据集达到 90% 满时，队列管理器通常会自动展开共享消息数据集，前提是选项 **DSEXPAND(YES)** 对 SMDS 定义有效。 当 SMDS 选项设置为 **DSEXPAND(YES)** 或 SMDS 选项设置为 **DSEXPAND(DEFAULT)** 且 CFSTRUCT 缺省选项设置为 **DSEXPAND(YES)** 时，这将适用。

如果由于创建数据集时未指定辅助分配大小而导致扩展尝试失败 (给出消息 IEC070I ，原因码为 203) 队列管理器使用大约为当前大小的 20% 的覆盖辅助分配来重复扩展请求。

扩展数据集时，新的数据集扩展数据块将作为扩展处理的一部分进行格式化，对于非常大的扩展数据块，这可能需要数十秒甚至几分钟。 在格式化完成后，新空间可供使用，并且目录已更新为显示新的高使用率控制区间。

如果正在非常迅速地创建新消息，那么在扩展处理完成之前，现有数据集可能会变满。 在这种情况下，无法分配空间的任何请求都将临时暂挂，直到扩展尝试完成并且新空间可供使用为止。 如果扩展成功，那么将自动重试该请求。

如果由于缺少可用空间或由于已达到最大扩展数据块而导致扩展尝试失败，那么将发出一条消息，说明失败原因，然后将受影响的 SMDS 的覆盖选项自动更改为 **DSEXPAND(NO)** 以防止进一步的扩展尝试。 在这种情况下，存在数据集变满的风险，在这种情况下可能需要进一步的操作，如 数据集变满中所述。

监视应用程序结构使用情况

可以使用 MVS **DISPLAY XCF,STRUCTURE** 命令显示应用程序结构的使用级别，该命令指定应用程序结构的全名 (包括队列共享组前缀)。 IXC360I 响应消息显示元素和条目的当前使用情况。

当结构使用情况超过 CFRM 策略中指定的 **FULLTHRESHOLD** 值时，系统将发出消息 IXC585E ，并可能执行自动 **ALTER** 操作 (如果指定) ，这可能会改变条目与元素的比率或增大结构大小。

优化缓冲池大小

共享缓冲池中的每个缓冲区都用于读取或写入多达逻辑块大小的一条消息的连续页范围。 如果消息溢出到更多块中，那么单独块中的每个页面范围都需要单独的缓冲区。

在写或读操作之后包含消息数据的缓冲区将保留在存储器中，并使用最近最少使用的 (LRU) 高速缓存方案进行复用，以便稍后再次读取相同数据的请求不需要转至磁盘。 当在同一系统上运行的应用程序写入共享消息并在不久后读回共享消息时，这将提供重大优化。 如果出于选择目的浏览了另一个队列管理器拥有的消息，然后进行了检索，那么这也避免了从磁盘重新读取消息的需要。

这意味着每个应用程序结构所需的缓冲区数是每个并发 API 请求的一个，该请求为该应用程序结构读取或写入大量消息，外加一些额外的缓冲区，这些缓冲区将用于保存最近访问的数据，以优化后续读访问。

对于共享缓冲池，如果没有足够的缓冲区，那么如果没有立即可用的缓冲区，那么 API 请求将直接等待。 但是，应该避免这种情况，因为它可能导致性能显着下降。

来自共享缓冲池的 **DISPLAY USAGE** 命令的统计信息显示在当前统计信息时间间隔内是否有任何缓冲区等待，并且还显示最小可用缓冲区数 (或负值表示随时等待缓冲区的最大线程数) ，已保存数据的缓冲区数以及缓冲区请求在 LRU 链上成功找到已保存数据的次数的百分比 (“LRU 命中数”) 而不是必须读取它 (“LRU 未命中”)[1].

- 如果存在任何等待，那么应该增加缓冲区数。
- 如果有许多未使用的缓冲区，那么缓冲区的数目可能会减少，以便在区域中有更多的存储空间用于其他用途。
- 如果有许多缓冲区包含已保存的数据，但对该已保存的数据执行的读取操作所占的比例非常小，那么如果存储器可以更好地用于其他用途，那么缓冲区数量可能会减少。 但是，缓冲区的数目不应减少超过可用缓冲区的最低数目，因为这可能会触发等待，并且最好是足够高，使最低可用缓冲区计数通常远高于零。

删除共享消息数据集

DELETE CFSTRUCT 命令 (仅当结构中的所有共享队列都为空且已关闭时才允许) 不会删除共享消息数据集本身，但可以在此命令完成后以通常的方式将其删除。 如果要将同一数据集复用为共享消息数据集，那么必须首先对其进行重新格式化以将其重置为空状态。

共享消息数据集的异常情况

在正常使用期间可能会发生一些异常情况，即使不存在软件或硬件错误也是如此。

数据集变满

如果数据集变满但无法扩展，或者扩展尝试失败，那么使用相应队列管理器将大消息写入相应应用程序结构的应用程序将收到错误 2192，即 MQRC_STORAGE_MEDIUM_FULL (也称为 MQRC_PAGESET_FULL)。

由于应该处理数据的应用程序中发生故障，导致大量积压消息累积，因此数据集可能变满。 如果是这样，进一步扩展数据集只会是一个临时解决方案，让处理应用程序尽快再次运行很重要。

如果可以提供更多空间，那么可以使用 **ALTER SMDS** 命令来设置 **DSEXPAND(YES)** 或 **DSEXPAND(DEFAULT)** (假定已设置 YES 或假定为 CFSTRUCT 定义的 **DSEXPAND** 缺省值) 以触发重试。 但是，如果失败的原因是已达到最大扩展数据块数，那么将拒绝新的扩展尝试并显示一条消息，并且将再次设置 **DSEXPAND(NO)** 。 在这种情况下，进一步扩展它的唯一方法是重新分配它，这涉及使它暂时不可用，如下所述。

需要移动或重新分配数据集

如果需要移动或扩展数据集但在正常使用中，那么可以临时将其取出以允许移动或重新分配数据集。 尝试在数据集不可用时使用该数据集的任何 API 请求都将接收到原因码 MQRC_DATA_SET_NOT_AVAILABLE 。

1. 使用 **RESET SMDS** 命令将数据集标记为 **ACCESS(DISABLED)**。 这将导致它正常关闭并由所有当前连接的队列管理器取消分配。
2. 根据需要移动或重新分配数据集，将旧内容复制到新分配的数据集，例如使用 "访问方法服务" (AMS) **REPRO** 命令。

   在将旧数据复制到新数据集之前，请勿尝试对新数据集进行预格式化，因为这将导致将复制的数据追加到格式化数据集的末尾。
3. 使用 **RESET SMDS** 命令再次将数据集标记为 **ACCESS(ENABLED)** ，以使其重新投入使用。

   如果旧内容小于新数据集的大小，那么打开新数据集时将自动对其余空间进行预格式化。

   如果旧内容大于新数据集的大小，那么队列管理器必须扫描耦合设施结构中的消息并重建空间映射，以确保没有任何活动数据丢失。 如果找到对位于新扩展数据块外部的数据块的任何引用，那么该数据集将标记为 **STATUS(FAILED)** ，并且必须通过将该数据集替换为正确的大小之一并将旧数据集再次复制到该数据块中或者使用 **RECOVER CFSTRUCT** 来恢复任何持久消息来进行修复。

耦合设施结构空间小

如果耦合设施结构空间不足，导致消息 IXC585E，那么 值得检查是否已设置卸载规则以确保在此情况下卸载最大数据量。 否则，可以使用 **ALTER CFSTRUCT** 命令来修改卸载规则。

[1] (Hits / (Hits+Misses))* 100

共享消息数据集的错误情境

有一些问题需要注意，这些问题只能由错误引起，不能在正常的操作情况下发生。

无法打开拥有的数据集

如果拥有共享消息数据集的队列管理器无法将其分配或打开，或者数据集属性不受支持，那么队列管理器将设置相应的 **SMDSCONN** 状态值 **ALLOCFAIL** 或 **OPENFAIL**，并将 **SMDSCONN** 可用性设置为 **AVAIL(ERROR)**。 它还将 SMDS 可用性设置为 **ACCESS(SUSPENDED)**。 更正错误后，使用 **RESET SMDS** 命令设置 **ACCESS(ENABLED)** 以触发重试，或者向拥有队列管理器发出 **START SMDSCONN** 命令。

无法打开只读数据集

如果队列管理器无法分配或打开另一个队列管理器拥有并标记为 **STATUS(ACTIVE)**的共享消息数据集，那么它假定这可能是由于其与数据集 (由 **SMDSCONN** 对象表示) 的连接存在特定问题，而不是数据集本身存在问题。

它将 **SMDSCONN** 标记为 **STATUS(ALLOCFAIL)** 或 **STATUS(OPENFAIL)** (视情况而定)，并将 **SMDSCONN** 可用性标记为 AVAIL (ERROR)，以防止进一步尝试使用它。

如果可以在不影响数据集本身状态的情况下更正问题，请使用 **START SMDSCONN** 命令来触发重试。

如果问题证明是数据集本身的问题，那么可以使用 **RESET SMDS** 命令将数据集标记为 **STATUS(FAILED)**，直到它已恢复为止。 恢复数据集后，将状态更改回 **STATUS(ACTIVE)** 的操作将导致通知其他队列管理器。 如果 **SMDSCONN** 标记为 **AVAIL(ERROR)**，那么它将自动更改回 **AVAIL(NORMAL)** 以触发打开数据集的新尝试。

数据集头已损坏

如果已成功打开数据集，但头信息的格式不正确，那么队列管理器将关闭并取消分配数据集，并将状态集设置为 **STATUS(FAILED)**，将可用性设置为 **ACCESS(SUSPENDED)**。 这允许使用 **RECOVER CFSTRUCT** 来恢复内容。

如果由于数据集包含来自另一个用途的残差数据并且未随后进行预格式化而产生错误，请对数据集进行预格式化，并使用 **RESET SMDS** 命令将状态更改为 **STATUS(RECOVERED)**。

否则，必须恢复数据集。

数据集意外为空

如果队列管理器打开标记为 **STATUS(ACTIVE)** 的数据集，但发现该数据集未初始化或新预格式化但以其他方式有效，那么队列管理器将关闭并取消分配共享消息数据集，然后将状态设置为 **STATUS(FAILED)**，并将可用性设置为 **ACCESS(SUSPENDED)**。

数据集存在永久 I/O 错误

如果数据集在成功 **OPEN** 处理后发生永久 I/O 错误，那么它可能需要恢复。 队列管理器会将数据集标记为 **STATUS(FAILED)**，以便所有当前连接的队列管理器都将关闭并取消分配该数据集。

数据集有可恢复的 I/O 错误

如果数据集存在硬件问题，那么这可能导致可恢复 I/O 错误，这些错误不会反映回队列管理器，但会导致性能显着下降，并且还会指示在不久的将来发生永久 I/O 错误的风险。

在这种情况下，可以通过使用 **RESET SMDS** 命令将数据集标记为 **STATUS(FAILED)**来使其脱机以进行恢复。 这将导致所有队列管理器将其关闭并取消分配，例如，可以将其移至新卷，然后再使其再次可用。

以这种方式使数据集不可用时，不会保存空间映射，因此队列管理器连接重新启动处理将需要扫描耦合设施结构以查找数据集中的消息并重建空间映射，然后才能使数据集再次可用。 作为替代方法，如果共享消息数据集仍然可用，那么可以通过使用 **RESET SMDS** 命令将数据集标记为 **ACCESS(DISABLED)**，使其更轻轻地不可用，直到它准备好再次可用为止。

数据集内容不正确

队列管理器无法直接检测到数据集包含不正确的数据或不是最新的数据，例如，因为必须从备份复原包含该数据集的卷。 但是，它会执行完整性检查，这使得任何此类错误都不太可能导致应用程序看到不正确的消息数据。

为了完整性检查目的，在将消息数据传递给用户程序之前，数据集中的每个消息块都以相应耦合设施条目标识的副本作为前缀，包括唯一的时间戳记，每当读取消息块时都会检查该时间戳记。 如果消息块前缀与条目标识不匹配 (并且在平均时间内未删除耦合设施条目)，那么假定消息块已损坏且不可用。

如果损坏的消息是持久的，那么数据集将标记为 **STATUS(FAILED)**，并且必须使用 **RECOVER CFSTRUCT** 命令来恢复结构内容。 如果损坏的消息是非持久消息，那么无法恢复该消息，因此将发出诊断消息并删除相应的耦合设施消息条目。

如果打开数据集时没有可用的已保存空间映射，那么将通过扫描耦合设施结构以获取对数据集中数据的引用来重建该映射。 在此扫描期间，队列管理器将执行多项操作:

1. 队列管理器确定当前保留在数据集中的最新消息 (如果有) 的位置。
2. 然后，队列管理器从数据集读取该消息，以确保块前缀与消息条目标识匹配

这些操作确保队列管理器检测数据集处于较低级别的任何情况，并将数据集标记为 FAILED。 但是，如果数据集是从先前副本复原的，并且自那时以来未添加任何新消息，或者自该副本之后添加的所有消息都已被读取和删除，那么此检查将容忍这种情况。

在数据集正常关闭的情况下，为了防止下级数据，队列管理器执行许多操作:

1. 当数据集正常关闭时，队列管理器会将空间映射时间戳记的副本保存在 Db2 内的 SMDS 对象中。
2. 然后，当再次打开数据集时，队列管理器会检查空间映射时间戳记是否相同

如果时间戳记不匹配，那么这表明可能已使用数据集的下级副本，因此队列管理器将忽略现有空间映射并对其进行重建，这将仅在未实际丢失消息数据的情况下成功。

注: 这些完整性检查不保证在所有理论上可能的情况下检测到低级别或损坏的数据集。 例如，如果消息块的启动有效，但其余数据已被部分覆盖，那么它们将不会检测到这种情况。

共享消息数据集的恢复方案

本部分描述了共享消息数据集恢复方案。

未丢失任何数据的数据集恢复

在某些情况下，可以复原失败数据集的正确内容，而无需实际恢复。 一个示例是，数据集包含来自先前使用的残差数据，并且未再次进行预格式化，可以通过对其进行预格式化来进行修正。 另一种情况是移动了数据集，但在复制数据的过程中发生了错误，可以通过正确再次复制数据来修正此错误。

在这种情况下，可以使用 **RESET SMDS** 命令设置 **STATUS(RECOVERED)** 来再次提供更正后的数据集。 如果可用性当前为 **ACCESS(SUSPENDED)**，那么将自动将其设置回 **ACCESS(ENABLED)**。

当通知拥有队列管理器已恢复数据集时，它会扫描结构内容以重构空间映射，然后将状态更改为 **STATUS(ACTIVE)**。 然后，其他队列管理器可以再次开始读取数据集。

具有 TYPE (NORMAL) 的数据集恢复

如果数据集的内容已丢失，但应用程序结构是使用 **RECOVER(YES)** 定义的，并且相应的恢复日志可用，那么可以使用 **RECOVER CFSTRUCT** 命令来恢复存储在该结构中的任何持久消息，包括卸载到共享消息数据集的持久消息数据。 此命令使用 **BACKUP CFSTRUCT** 命令记录的信息以及自备份时间以来记录的对持久消息的所有更改来复原当前状态。

RECOVER CFSTRUCT 命令始终将耦合设施结构中的所有持久消息与存储在 Db2 中的卸载消息数据一起恢复。 对于存储在共享消息数据集中的卸载数据，仅当每个数据集已标记为 **STATUS(FAILED)** 时，或者在恢复处理打开时发现它意外为空或无效时，才会选择对其进行恢复处理。 标记为活动且通过验证检查的任何共享消息数据集都不需要恢复，因为现有消息数据已正确，但头已更新以指示在恢复后将需要重建任何已保存的空间映射。

仅当结构已标记为失败时，才可以进行恢复处理，因为需要通过恢复处理来重构结构的完整内容。 但是，如果至少一个共享消息数据集已标记为失败，那么 **RECOVER CFSTRUCT** 命令将在必要时自动将结构标记为失败，以允许继续进行恢复处理。

可以从队列共享组中的任何队列管理器执行恢复，前提是已授予其对相关数据集的写访问权。

仅备份和记录持久消息，因此正常恢复处理将复原所有持久消息，但将导致结构中的任何非持久消息丢失。

恢复完成后，为恢复选择的任何数据集将自动更改为 **STATUS(RECOVERED)**，如果可用性为 **ACCESS(SUSPENDED)**，那么将更改为 **ACCESS(ENABLED)**。 队列管理器通过扫描耦合设施中的消息来重建每个数据集的空间映射，然后将数据集标记为 **STATUS(ACTIVE)**，以便可以再次使用该数据集。

使用 TYPE (PURGE) 进行数据集恢复

对于可恢复结构，如果数据集内容已丢失，但由于某种原因无法恢复，例如，由于恢复日志不可用或恢复需要太长时间，那么可以将 **RECOVER CFSTRUCT** 命令与 **TYPE(PURGE)** 配合使用以将结构恢复到可用状态。 这会将结构重置为空状态，并将所有关联的数据集标记为 **STATUS(EMPTY)**。

删除应用程序结构

如果使用 MVS **SETXCF FORCE** 命令删除了不可恢复的应用程序结构，或者由于结构故障，那么在下次连接该结构时，将发出消息 CSQE028I 以声明已重置该结构，并且已废弃所有现有消息，并且任何现有数据集也将自动重置为 **STATUS(EMPTY)**。 此操作将使不可恢复的结构在该结构或任何关联数据集中丢失数据后再次可用。

如果删除了可恢复的应用程序结构，那么将以与该结构发生故障相同的方式处理该结构。

数据集恢复失败

如果 **RECOVER CFSTRUCT** 由于某些原因 (例如，由于日志数据集不再可用，或者由于正在进行恢复时队列管理器已终止) 而无法完成，那么将在头中标记至少已启动恢复的任何数据集，以显示已尝试部分恢复，并且该数据集将保持 **STATUS(FAILED)** 状态。

在这种情况下，选项是重复原始恢复请求，或者改为使用 **TYPE(PURGE)** 进行恢复，从而废弃现有数据。

如果尝试将数据集标记为 **STATUS(RECOVERED)** 而未实际恢复该数据集，那么下次打开该数据集时，队列管理器将看到该头指示未完成恢复，并将其再次标记为 **STATUS(FAILED)**。

场外灾难恢复

对于站外灾难恢复，只能使用包含 CFSTRUCT 定义和关联 SMDS 状态信息的日志和 Db2 共享对象来重新创建持久共享消息。

设置包含定义的 Db2 表后，可以将应用程序结构和共享消息数据集设置为空。 当队列管理器连接到它们并发现它们意外为空时，它会将它们标记为失败，之后可以使用单个 **RECOVER CFSTRUCT** 命令来恢复所有受影响结构的所有持久消息。

SMDS 相关命令

本主题描述并提供对与共享消息数据集相关的命令的访问权。

显示和更改与大型消息卸载 (**OFFLOAD** 和卸载规则) 和共享消息数据集 (**DSGROUP, DSBLOCK, DSBUFS, DSEXPAND)**) 相关的 **CFSTRUCT** 选项:

- DISPLAY CFSTRUCT
- DEFINE CFSTRUCT
- 变更 CFSTRUCT
- 删除 CFSTRUCT

显示 **CFSTRUCT** 与大型消息卸载相关的状态 **(OFFLDUSE)**:

- 显示 CFSTATUS

显示和变更覆盖数据集选项 (**DSEXPAND** 和 **DSBUFS**) 对于个别队列管理器:

- 显示 SMDS
- 变更 SMDS

显示或修改队列共享组中数据集的状态和可用性:

- 显示 CFSTATUS TYPE (SMDS)
- 重置 SMDS

显示队列管理器的 SMDS 数据集空间使用情况和缓冲区使用情况信息:

- DISPLAY USAGE TYPE (SMDS)

显示或修改连接的状态和可用性 (**SMDSCONN**) 到单个队列管理器中的数据集:

- 显示 SMDSCONN
- 开始 SMDSCONN
- STOP SMDSCONN

备份和恢复共享消息，必要时包括 SMDS 中的大型消息数据:

- 备份 cfstruct
- RECOVER CFSTRUCT

使用共享队列的优点

共享队列允许 IBM MQ 应用程序可伸缩，高度可用，并允许实现工作负载均衡。

共享队列的优点

共享队列体系结构 (克隆的服务器从单个共享队列中提取工作) 具有一些有用的属性:

- 通过添加服务器应用程序的新实例，甚至添加具有队列管理器 (在队列共享组中) 和应用程序副本的新 z/OS 映像，它是可扩展的。
- 高度可用。
- 它根据队列共享组中每个队列管理器的可用处理容量自然执行 *pull* 工作负载均衡。

使用共享队列实现高可用性

以下示例说明如何使用共享队列来提高应用程序可用性。

请考虑以下 IBM MQ 场景: 在网络中运行的客户机应用程序想要发出在 z/OS 上运行的服务器应用程序的请求。 客户机应用程序构造请求消息并将其放在请求队列上。 然后，客户机等待来自服务器的应答，该应答发送到请求消息的消息描述符中指定的应答队列。

IBM MQ 管理请求消息从客户机到 z/OS 上的服务器输入队列的传输以及从服务器到客户机的响应的传输。 通过将服务器的输入队列定义为共享队列，可以在队列共享组中的任何队列管理器上检索放入队列的任何消息。 这意味着您可以在综合系统中的每个 z/OS 映像上配置队列管理器，并且通过将它们全部连接到同一个队列共享组，其中任何一个都可以访问服务器输入队列上的消息。

服务器的输入队列上的消息仍可用，即使其中一个队列管理器异常终止或由于管理原因必须将其停止也是如此。 您可以使整个 z/OS 映像脱机，并且消息仍可用。

要利用共享队列上消息的此可用性，请在综合系统中的每个 z/OS 映像上运行服务器应用程序实例，以提供更高的服务器应用程序容量和可用性，如 第 307 页的图 62 中所示。

服务器应用程序的一个实例从共享队列中检索请求消息，并根据内容执行其处理，生成作为 IBM MQ 消息发送回客户机的结果。 响应消息指定给在请求消息的消息描述符中指定的应答队列和应答队列管理器。

有许多选项可用于配置返回路径。 有关这些选项的更多信息，请参阅 第 322 页的『分布式排队和队列共享组』。

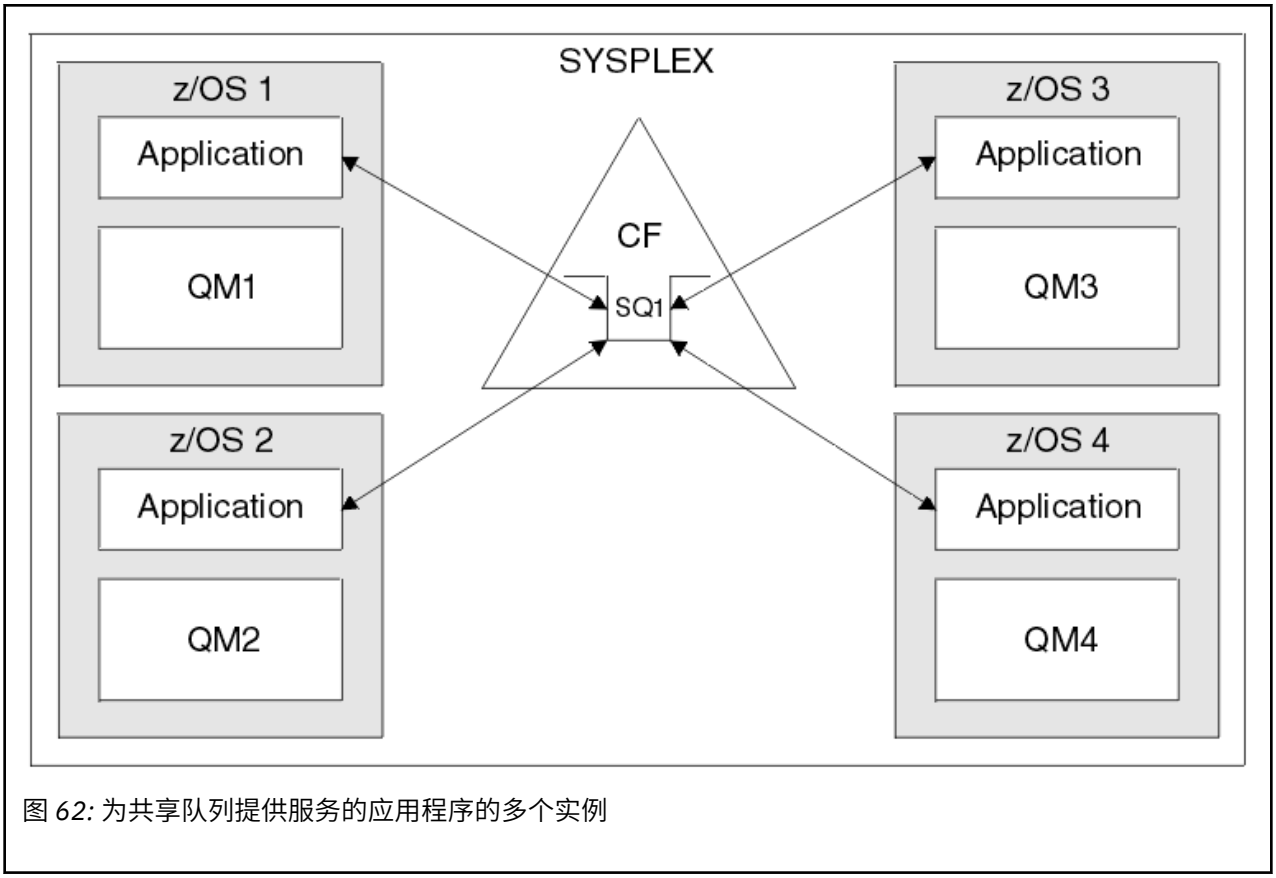


图 *62:* 为共享队列提供服务的应用程序的多个实例

对等恢复

为了进一步增强队列共享组中消息的可用性， IBM MQ 会检测组中的另一个队列管理器是否与耦合设施异常断开连接，并在可能的情况下完成仍处于暂挂状态的该队列管理器的工作单元。 此功能称为 对等恢复 。

假设队列管理器在应用程序已从同步点中的队列检索到请求消息但尚未放入响应消息或已落实工作单元的位置异常终止。 队列共享组中的另一个队列管理器检测到故障，并回退正在失败的队列管理器上执行的未完成工作单元。 这意味着将请求消息放回到请求队列中，并可供其他某个服务器实例处理，而无需等待失败的队列管理器重新启动。

如果 IBM MQ 无法自动解析工作单元，那么您可以手动解析共享部分，以使队列共享组中的另一个队列管理器能够继续处理该工作。

将存储类内存与共享队列配合使用

当与 IBM MQ for z/OS 共享队列配合使用时，使用存储类内存 (SCM) 可能是有利的。

z13， zEC12 和 zBC12 机器允许安装 Flash Express 卡。 这些卡包含闪存固态驱动器 (SSD)。 安装后，可将卡中的闪存分配给通常称为 SCM 的一个或多个 LPAR。

SCM 在 I/O 等待时间和成本方面介于实存储器和直接访问存储设备 (DASD) 之间。 由于 SCM 没有移动部件，因此与 DASD 相比，它的 I/O 延迟要低得多。

SCM 也比实存储器便宜得多。 因此，可以以相对较低的成本安装大量存储器; 例如，一对 Flash Express 卡包含 1424 GB 可用存储器。

这些特征意味着当必须在短时间内从实存储器中获取大量数据时， SCM 很有用，因为该数据可以比写入 DASD 更快地写入 SCM。 使用包含 IBM MQ 共享队列的耦合设施 (CF) 列表结构时，此特定点非常有用。

为什么列表结构会填充

定义 CF 结构时，将使用描述该结构的最大大小的 SIZE 属性对其进行配置。 由于 CF 结构始终永久驻留在实存储器中，因此在 CF 上定义的结构的 SIZE 属性总和应该小于分配给 CF 的实存储器量。

因此，有恒定的压力将任何给定结构的 SIZE 值保持在最小的可能值，以便更多的结构可以拟合到 CF 中。 但是，确保结构足够大以达到其目的可能会产生相互冲突的压力，因为使结构太小意味着它可能被填满，从而干扰使用它的应用程序或子系统。

强烈需要根据其预期使用情况来准确调整结构的大小。 但是，这项任务很难完成，因为工作负载会随着时间的推移而发生变化，并且要考虑它们的波动并不容易。

IBM MQ 共享队列使用 CF 列表结构来存储消息。 IBM MQ 调用 CF 结构，其中包含消息和应用程序结构。

使用存储在 IBM MQ CFSTRUCT 对象中的信息来引用应用程序结构。 将小于 63 KB 的消息放入共享队列时，该消息将完全作为单个列表条目以及零个或多个列表元素存储在应用程序结构中。

由于 IBM MQ 共享队列使用列表结构，因此所描述的压力也会影响共享队列。 在这种情况下，可以存储在共享队列中的最大消息数是下列各项的函数:

- 队列上消息的大小
- 结构的最大大小
- 结构中可用的条目数和元素数

因为多达 512 个共享队列可以使用相同的结构，并有效竞争条目和元素，这使得问题变得更加复杂

IBM MQ 队列用于在应用程序之间传输数据，因此常见情况是应用程序将消息放入队列，而应该获取这些消息的伙伴应用程序未在运行。

发生此情况时，队列上的消息数会随时间推移而增加，直到发生以下一种或多种情况:

- 放置应用程序停止放置消息。
- 获取应用程序开始获取消息。
- 队列上的现有消息将开始到期，并且将从队列中除去。
- 队列达到其最大深度，在此情况下， MQRC_Q_FULL 原因码将返回到放置应用程序。
- 包含共享队列的结构达到其最大大小，或者包含该结构的 CF 耗尽可用存储器。 在任一情况下，都会将 MQRC_STORAGE_MEDIUM_FULL 原因码返回到放置应用程序。

在最后三种情况下，队列已满。 此时，放置应用程序存在问题，因为其消息无处可去。 放置应用程序通常通过使用以下一个或多个解决方案来解决此问题:

- 重复重试放入消息 (可选) ，并在两次重试之间延迟。
- 将消息放在其他位置，例如数据库或文件。 稍后可访问这些消息，并将其正常放入队列中。
- 如果消息是非持久消息，请将其废弃。

但是，对于某些应用程序类 (例如，具有大量入局消息或无法访问文件系统的应用程序类) ，这些解决方案不实用。 确实需要确保队列永远不会或极不可能填满，这与共享队列特别相关。

SMDS 和卸载规则

IBM WebSphere MQ 7.1 中引入的卸载规则提供了一种降低应用程序结构填充可能性的方法。

每个应用程序结构都有三个与之关联的规则，使用三对关键字指定:

- OFFLD1SZ 和 OFFLD1TH
- OFFLD2SZ 和 OFFLD2TH
- OFFLD3SZ 和 OFFLD3TH

每个规则都指定必须满足的条件，才能将消息数据卸载到与应用程序结构关联的存储机制。 当前有两种类型的存储机制可用:

- Db2
- 虚拟存储器访问方法 (VSAM) 线性数据集组， IBM MQ 调用共享消息数据集 (SMDS)。

以下示例显示了 MQSC 命令，该命令使用 DEFINE CFSTRUCT 命令来创建名为 LIST1 的应用程序结构。

此结构具有缺省卸载规则，并使用 SMDS 作为卸载机制。 这意味着当结构满 70% (OFFLD1TH) 时，所有 32 KB 或更大 (OFFLD1SZ) 的消息都将卸载到 SMDS。

同样，当结构为 80% 已满 (OFFLD2TH) 时，将卸载所有 4 KB 或更大 (OFFLD2SZ) 的消息。 当结构为 90% 已满 (OFFLD3TH) 时，将卸载所有消息 (OFFLD3SZ)。

```
DEFINE CFSTRUCT(LIST1)
CFLEVEL(5)
OFFLOAD(SMDS)
OFFLD1SZ(32K) OFFLD1TH(70)
OFFLD2SZ(4K) OFFLD2TH(80)
OFFLD3SZ(0K) OFFLD3TH(90)
```

卸载消息存储在卸载介质中，而指向消息的指针存储在结构中。 虽然卸载规则会减少结构填充的机会，但由于在结构中的消息数据耗尽，因此某些数据仍会写入每个消息的结构中。 即，指向已卸载消息的指针。

此外，卸载规则会带来性能成本。 将消息写入结构比较快，主要由向 CF 发送写入请求所花费的时间支配。 实际写入结构的速度很快，以实际存储速度发生。

将消息写入 SMDS 要慢得多，因为它包括写入消息指针的结构，以及将消息数据写入 SMDS。 第二个写操作以 DASD 速度完成，并有可能增加等待时间。 如果使用 Db2 作为卸载机制，那么性能成本要高得多。

存储类内存如何与 *IBM MQ for z/OS* 配合使用

将存储类内存 (SCM) 与 IBM MQ for z/OS 共享队列配合使用的概述。

处于 CFLEVEL 19 或更高级别的耦合设施 (CF) 可以将 SCM 分配给它。 然后，可以将该 CF 中定义的结构配置为使用 SCM 来减少结构填充的机会 (称为结构完全条件)。 当配置为使用 SCM 的结构在系统确定的点之后填满时， CF 开始将数据从该结构移动到 SCM 中，这将释放该结构中用于新数据的空间。

注: 由于 SCM 本身可以填充，因此将 SCM 分配给结构只会降低结构完全情况的可能性，但不会完全消除发生一次情况的可能性。

通过在耦合设施资源管理器 (CFRM) 策略中同时指定 **SCMALGORITHM** 和 **SCMMAXSIZE** 关键字 (包含该结构的定义) ，将结构配置为使用 SCM。

请注意，在指定这些关键字并应用 CFRM 策略之后，必须重建或取消分配结构以使其生效。

SCMALGORITHM

由于 SCM 的输入/输出速度比实存储器的输入/输出速度慢，因此 CF 使用一种算法，该算法针对结构的预期使用进行定制，以减少写入 SCM 或从中读取 SCM 的影响。

该算法由结构的 CFRM 策略中的 **SCMALGORITHM** 关键字使用 *KEYPRIORITY1* 值进行配置。 请注意，您应该仅将 *KEYPRIORITY1* 值用于 IBM MQ 共享队列所使用的列表结构。

KEYPRIORITY1 算法的工作原理是假定大多数应用程序将按优先级顺序从共享队列获取消息; 即，当应用程序获取消息时，它将获取具有最高优先级的最旧消息。

当结构开始填充超过系统定义的阈值 90% 时， CF 将开始异步迁移下一个可能性最小的消息。 这些是最近放入队列的优先级较低的消息。

将消息从结构异步迁移到 SCM 称为 "预登台"。

预登台可降低使用 SCM 的性能成本，因为这会降低在对 SCM 进行同步输入/输出期间应用程序被阻塞的机会。

除了预登台之外， *KEYPRIORITY1* 算法还会在有足够可用空间时异步将来自 SCM 的消息返回到结构中。 对于 *KEYPRIORITY1* 算法，这意味着当结构小于或等于满 70% 时。

将来自 SCM 的消息引入到结构中的操作称为 "预取"。

预取降低了应用程序尝试获取已预登台到 SCM 的消息的可能性，并且需要等待 CF 同步将消息返回到结构中。

SCMMAXSIZE

SCMMAXSIZE 关键字定义可由结构使用的 SCM 的最大数量。 由于需要时 SCM 由 CF 分配给结构，因此可以指定大于可用 SCM 总量的 **SCMMAXSIZE** 。 这被称为 "过度承诺"。

要点: 切勿过度落实 SCM。 如果这样做，依赖它的应用程序将不会获得它们期望的行为。 例如，使用共享队列的 IBM MQ 应用程序可能会获得意外的 MQRC_STORAGE_MEDIUM_FULL 原因码。

CF 使用各种数据结构来跟踪其对 SCM 的使用。 这些数据结构驻留在分配给 CF 的实存储器中，因此会减少结构可使用的实存储器量。 这些数据结构所使用的存储器被称为 "扩充空间"。

使用 SCM 配置结构时，会将少量实存储器从 CF 分配到称为固定扩充空间的结构。 即使结构从未实际使用任何 SCM，也会分配此值。 随着来自该结构的数据存储到 SCM 中，将从 CF 中的备用实存储器分配额外的动态扩充空间。

从 SCM 中除去数据时，会将动态扩充空间返回到 CF。 扩充空间 (固定或动态) 从不从分配给结构的实存储器中获取。

除了扩充存储器外，当结构配置为使用 SCM 时，该结构使用的控制存储量也会增加。 这意味着在没有配置 SCM 的情况下，使用 SCM 配置的列表结构可以包含比相同大小的结构更少的条目和元素。

要了解 SCM 对新结构或现有结构的影响，请使用 CFSizer 工具。

需要注意的最后一个要点是，在将数据从结构移动到 SCM 中，并且已使用动态扩充空间之后，不能手动或自动更改结构。

即，分配给该结构的存储量不能增加或减少，该结构所使用的入元素比率不能改变，以此类推。 要使结构再次可更改，该结构不得具有任何存储在 SCM 中的数据，并且不得使用动态扩充存储器。

为何使用 *SCM*
紧急存储和提高性能是将 SCM 与 IBM MQ for z/OS 配合使用的两个用例。

本部分介绍了两种可能的场景背后的理论。 有关如何设置方案的更多详细信息，请参阅:

- 第 312 页的『紧急存储器-基本配置』
- 第 318 页的『改进的性能-基本配置』

要点: 将 SCM 与 CF 结构配合使用不依赖于任何特定版本的 IBM MQ。 但是，紧急存储方案仅适用于 IBM WebSphere MQ 7.1 和更高版本，因为它需要 SMDS 和卸载规则。

紧急存储器

SMDS 和消息卸载可与 SCM 配合使用，以降低在扩展中断期间将 MQRC_STORAGE_MEDIUM_FULL 原因码返回到 IBM MQ 应用程序的可能性。

概述

在应用程序结构上配置单个共享队列。 放置应用程序将消息放入共享队列; 获取应用程序从共享队列获取消息。

在正常运行期间，队列深度预期接近零，但业务需求指示系统必须能够容忍获取应用程序中断两小时。 这意味着共享队列必须能够包含来自放置应用程序的两小时消息。

目前，通过使用缺省卸载规则和 SMDS 来实现此过程，从而使结构的大小最小化，同时降低与卸载相关联的性能成本。

预期发送到共享队列的消息速率在短期到中期将增加一倍。 虽然仍存在要求系统能够容忍两小时的中断，但 CF 中没有足够的实际存储空间来使结构大小增加一倍。

由于包含应用程序结构的 CF 位于 zEC12 机器上，因此存在将足够的 SCM 与该结构相关联以存储足够的消息的可能性，以便可以容忍两小时的中断。

考虑一段时间内发生的情况:

1. 最初，该系统处于稳定状态。 放入和获取应用程序都正常运行，并且队列深度接近或处于零。 结果是应用程序结构很大程度上是空的。

2. 在某个时间，获取应用程序发生意外故障并停止。 放置应用程序继续将消息放置到队列中，应用程序结构开始填充。
3. 在结构达到 70% 已满后，将满足第一个卸载规则的条件，并且将大小大于或等于 32 KB 的所有消息卸载到 SMDS。

   请参阅 第 308 页的『SMDS 和卸载规则』 以获取卸载规则的概述。
4. 当消息继续放入共享队列时，结构将继续填充 (因为消息数据存储在结构中，或者由于指向要存储在结构中的已卸载消息的指针)。

   当结构达到 80% 满时，第二个卸载规则将开始应用， 4 KB 或更大的消息将卸载到 SMDS。
5. 当结构超过 90% 已满时，所有消息都将卸载到 SMDS ，并且只有消息指针放置在结构中。

   大约此时，预登台算法开始运行，并开始将数据从结构移动到 SCM 中。 假定队列上的所有消息都具有相同的优先级，那么会预先暂存最新的消息。

   因为现在正在将所有消息卸载到 SMDS ，所以要移动到 SCM 中的数据不是实际的消息数据，而是指向 SMDS 上的消息的指针。

   因此，可以存储在该结构以及与该结构相关联的 SCM 和 SMDS 的组合上的消息数量非常大。

   性能: 在中断的此阶段，由于必须写入 SMDS ，因此放置应用程序可能会遭受一定程度的性能降级。 在这种情况下，在性能方面， SCM 的使用不应成为放置应用程序的限制因素。 SCM 提供了额外的空间来防止结构填充。
6. 最终获取应用程序再次可用，中断已结束。

   但是，该结构仍在使用 SCM。 获取应用程序开始从队列中读取消息，首先获取最旧的最高优先级消息。

   因为这些消息是在结构开始填满之前写出来的，所以它们完全是从结构的实存部分出来的。
7. 随着结构开始清空，它将低于预登台处于活动状态的阈值，因此预登台将停止。
8. 结构使用率低于卸载规则生效的点，因此除非消息超过 63 KB ，否则不再将其卸载到 SMDS。

   大约此时，预取算法开始将数据从 SCM 移动到结构中。 由于获取应用程序按 SCM 算法所期望的顺序从队列中获取消息，因此在获取应用程序需要消息之前会引入消息。

   结果是获取应用程序从不需要等待从 SCM 同步引入消息。
9. 当获取应用程序继续下移队列时，它将开始检索已卸载到 SMDS 的消息。
10. 最后，系统再次处于稳定状态。 SCM 或 SMDS 中未存储任何消息，并且队列深度接近零。

提高了性能

此场景描述使用 SCM 来增加可以存储在共享队列上的消息数，而不会产生使用 SMDS 的性能成本。

说明

对于此场景，放置和获取应用程序通过存储在应用程序结构中的共享队列进行通信。

放置应用程序往往以突发方式运行，当它在短时间内放置大量消息时。 然后，在很长一段时间内，它根本不会生成任何消息。

获取应用程序按顺序处理每条消息，并对每条消息执行复杂处理。 因此，大多数情况下队列深度为零，但放置应用程序开始运行时除外，在此情况下，队列深度开始增加，因为放置消息的速度比获取消息的速度要快。

队列深度会增加，直到放入应用程序停止，并且获取应用程序有足够的时间处理队列上的所有消息。

注意:

1. 在此场景中，关键因素是性能。 发送到队列的消息始终小于 63 KB ，因此从不需要卸载到 SMDS。
2. 已调整应用程序结构的大小，使其足以包含将应用程序放入单个 "分类群发" 中的所有消息。
3. 必须全部禁用卸载规则，以便即使在结构开始填充时，也不会将消息卸载到 SMDS。 这是因为与向 SMDS 写入消息和从 SMDS 读取消息相关联的性能成本被视为不可接受。

随着时间的推移，将应用程序发送到分类群发的消息数必须增加几个数量级。 由于获取应用程序必须按顺序处理每条消息，因此队列上的消息数会增加到结构填充的点。

此时，放置应用程序在放置消息时接收到原因码 (MQRC_STORAGE_MEDIUM_FULL)，而放置操作失败。 放入应用程序只能在无法将消息放入队列时短暂容许时间段。 如果周期太长，那么应用程序将结束。

假定您没有时间或技能来重写放置应用程序或获取应用程序，那么此问题有三种可能的解决方案:

1. 增大应用程序结构的大小。
2. 将卸载规则添加到应用程序结构，以便在队列开始填满时将消息卸载到 SMDS。
3. 使 SCM 与结构相关联。

第一个解决方案可以快速实施，但在 CF 上没有足够的实际存储可用。

第二个解决方案也可以快速实施，但卸载到 SMDS 对性能的影响被认为太大，无法使用此选项。

第三种解决方案 (将 SCM 与结构相关联) 提供了可接受的成本和性能平衡。

将 SCM 与结构相关联会导致 CF 中的实存储器的使用率更高，因为使用了操作的扩充存储器。 但是，实际存储量将小于第一个选项中使用的量。

另一个考虑因素是 SCM 的成本。 但是，这一成本比实际存储要便宜得多。 这些因素结合起来，使得第三种选择比第一种选择更便宜。

虽然第三个选项可能不如第一个选项，但 CF 使用的预取和预登台算法可以组合在一起，使性能方面的差异可以接受，或者在某些情况下可以忽略不计。

当然，性能比使用 SMDS 卸载消息要好得多。

考虑一段时间内发生的情况:

1. 最初，获取应用程序处于活动状态，正在等待消息传递到共享队列。 放置应用程序未处于活动状态，并且共享队列为空。
2. 在一定时间，放置应用程序变为活动状态，并开始将大量消息放入共享队列。 获取应用程序开始获取消息，但队列深度快速开始增加，因为获取应用程序比放置应用程序慢。

   因此，应用程序结构开始填充。
3. 随着时间的增加，放置应用程序仍处于活动状态。 应用程序结构填充了大约 90%。

   这是 SCM 预登台算法开始将消息从结构移动到 SCM 中，从而释放结构中的空间。

   因为获取应用程序首先从队列中获取最旧的，最高优先级的消息，所以它总是从结构中获取消息，而不需要等待消息从 SCM 同步引入到结构中。
4. 放置应用程序仍处于活动状态，并将消息放置到共享队列。 但是，应用程序从不接收 MQRC_STORAGE_MEDIUM_FULL 原因码，因为 SCM 中存在足够的空间来存储结构中不适合的所有消息。
5. 最终，放置应用程序将停止，因为它没有更多要放置的消息。

   预登台算法停止，因为结构在使用中低于 90%，并且获取应用程序继续处理队列中的消息。
6. 当获取应用程序开始释放结构中的空间时，预取算法开始将消息从 SCM 返回到结构中。

   由于获取应用程序按预取算法所期望的顺序处理消息，因此获取应用程序从不阻塞等待消息数据从 SCM 同步引入到结构中。
7. 最后，获取应用程序会处理共享队列上的所有消息，并等到下一条消息可用。 结构和 SCM 不包含消息。

紧急存储器-基本配置
如何在 IBM MQ 上设置紧急存储器的基本方案。

关于此任务

SMDS 和消息卸载可与 SCM 配合使用，以降低在扩展中断期间将 MQRC_STORAGE_MEDIUM_FULL 原因码返回到 IBM MQ 应用程序的可能性。

例如，您的企业有一个将消息放入队列的应用程序和一个从队列中获取消息的应用程序。 在正常运行期间，您期望队列深度接近零，但业务需求指示系统能够容忍获取消息的应用程序中断两小时。

这意味着正在使用的共享队列必须能够包含来自放置应用程序的两小时消息。 目前，您可以使用缺省卸载规则和 SMDS 来实现此目标。

您期望将消息发送到共享队列的速率在短期到中期内增加一倍。 虽然您要求系统能够容忍两小时的中断仍然存在，但 CF 中没有足够的实存储器可用于将结构的大小增加一倍。 由于包含应用程序结构的 CF 位于 zEC12 机器上，因此您可以将足够的 SCM 与该结构相关联，以存储足够的消息，从而可以容忍两个小时的中断

此初始方案使用以下内容:

- 包含单个队列管理器 CSQ3 的队列共享组 IBM1。 除管理结构外，队列共享组还定义了单个应用程序结构 SCEN1。
- 耦合设施 (CF) CF01，其中 SCEN1 应用程序结构存储为 IBM1SCEN1 结构。 此结构的最大大小为 1 GB。
- 应用程序结构使用的单个共享队列 SCEN1.Q 。

此配置在 第 313 页的图 63 中说明。

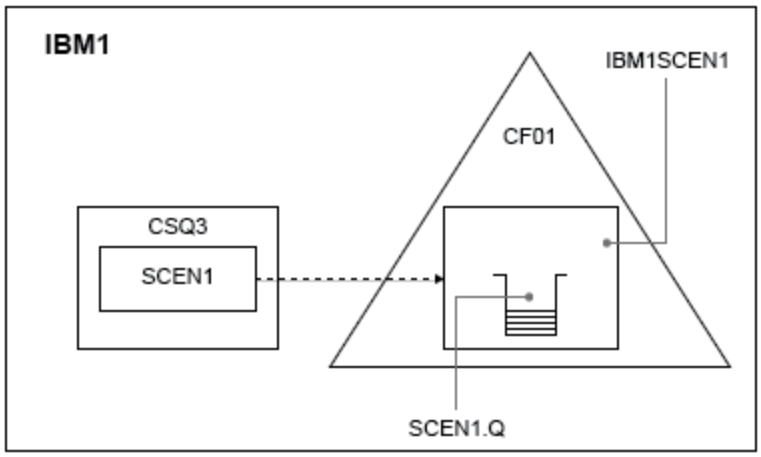


图 *63:* 基本配置

此外，假定队列管理器 CSQ3 已是队列共享组 IBM1 的唯一成员。

必须将结构 IBM1SCEN1 的定义添加到耦合设施资源管理器 (CFRM) 策略。 为了简单起见，定义了结构，以便只能通过指定 PREFLIST(CF01)在单个耦合设施 CF01 中创建该结构。

注意: 要在生产系统中实现高可用性，对于 IBM MQ 所使用的任何结构，应该在 PREFLIST 中至少包含两个 CF。

过程

1. 使用以下命令刷新 CFRM 策略:

```
SETXCF START,POLICY,TYPE=CFRM,POLNAME=IBM1SCEN1
```

结构 IBM1SCEN1 的样本 CFRM 策略:

```
STRUCTURE
NAME(IBM1SCEN1)
SIZE(1024M)
INITSIZE(512M)
ALLOWAUTOALT(YES)
FULLTHRESHOLD(85)
PREFLIST(CF01)
ALLOWREALLOCATE(YES)
DUPLEX(DISABLED)
ENFORCEORDER(NO)
```

2. 使用以下命令验证是否已正确创建结构:

```
D XCF,STR,STRNAME=IBM1SCEN1
```

此时，尚未将结构分配给队列共享组 (由 STATUS 行显示)。

3. 配置 IBM MQ 以使用 CFRM 策略中定义的结构。
   a. 使用具有结构名称 SCEN1 的 DEFINE CFSTRUCT 命令来创建 IBM MQ CFSTRUCT 对象:

```
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. 使用 DISPLAY CFSTRUCT 命令验证结构。
   c. 使用以下 MQSC 命令定义 SCEN1.Q 共享队列以使用 SCEN1 结构:

```
DEFINE QLOCAL(SCEN1.Q) QSGDISP(SHARED) CFSTRUCT(SCEN1) MAXDEPTH(999999999)
```

4. 使用 IBM MQ 资源管理器将单个消息放入队列 SCEN1.Q，然后再次关闭该消息。
5. 发出以下命令以检查现在是否已分配该结构:

```
D XCF,STR,STRNAME=IBM1SCEN1
```

检入命令的输出，即 STATUS 行显示 ALLOCATED。

结果

您已创建基本配置。 现在，您可以使用选择的任何方法来了解配置的基线性能。

下一步做什么

将 SMDS 和 SCM 添加到初始结构

相关概念

第 307 页的『将存储类内存与共享队列配合使用』

当与 IBM MQ for z/OS 共享队列配合使用时，使用存储类内存 (SCM) 可能是有利的。

将 *SMDS* 和 *SCM* 添加到初始结构

如何在 IBM MQ 上为紧急存储器添加 SMDS 和 SCM。

关于此任务

此部分任务使用 第 312 页的『紧急存储器-基本配置』中描述的基本配置。 该方案描述了将共享消息数据集 (SMDS) 以及 SCM 添加到初始结构。

此最终配置在 第 315 页的图 64 中说明。

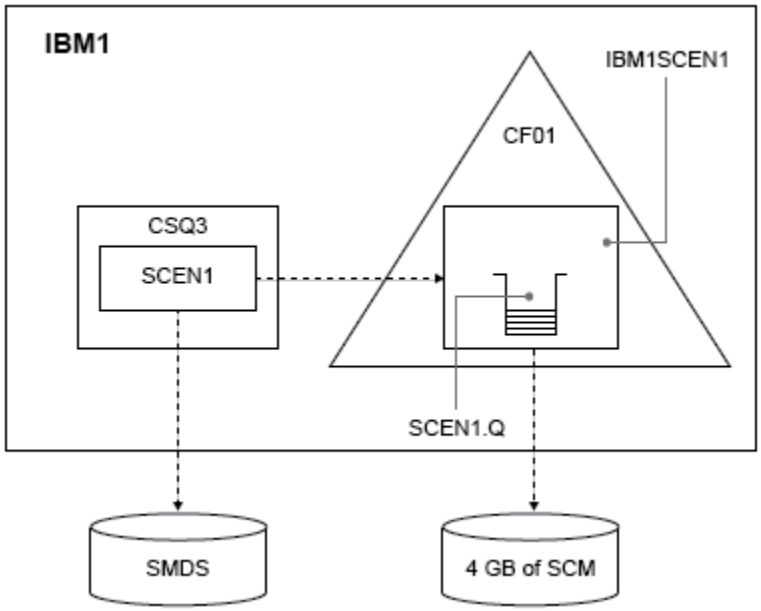


图 *64:* 为紧急存储器添加 *SMDS* 和 *SCM* 的配置

过程

1. 通过编辑 **CSQ4SMDS** 样本 JCL ， 创建 SCEN1 应用程序结构使用的 SMDS 数据集，如下所示:

```
//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. 发出 ALTER CFSTRUCT 命令以更改 SCEN1 应用程序结构，以使用 SMDS 进行卸载，并实现缺省卸载规则:

```
ALTER CFSTRUCT(SCEN1) OFFLOAD(SMDS) OFFLD1SZ(32K) OFFLD2SZ(4K) OFFLD3SZ(0K)
DSGROUP('CSQSMDS.SCEN1.*.SMDS') DSBLOCK(1M)
```

   请注意下列事项:

   - 由于 SCEN1.Q 是 SCEN1 应用程序结构上的唯一共享队列，因此 **DSBLOCK** 值已设置为 1M，这是可能的最大值。 这应该是我们场景最高效的设置。
   - 由于放置应用程序发送的消息为 30 KB ， 因此当结构已满 80% 时，直到满足第二个卸载规则时，才会启动对 SMDS 的卸载。

3. 再次运行测试应用程序。

请注意增加队列上消息的存储量。

4. 通过执行以下过程将 4 GB SCM 添加到结构 IBM1SCEN1 中:
   a) 通过发出以下命令，检查已安装并分配给 CF01 的 SCM 数量:

```
D CF,CFNAME=CF01
```

   b) 检查所显示输出的 STORAGE CONFIGURATION 部分中的 STORAGE-CLASS MEMORY 图以查看可用存储器。
   c) 使用 SCMMAXSIZE 和 SCMALGORITHM 关键字更新 CFRM 策略，如下所示:

```
STRUCTURE
NAME(IBM1SCEN1)
SIZE(1024M)
INITSIZE(512M)
ALLOWAUTOALT(YES)
FULLTHRESHOLD(85)
PREFLIST(CF01)
ALLOWREALLOCATE(YES)
DUPLEX(DISABLED)
ENFORCEORDER(NO)
SCMMAXSIZE(4G)
SCMALGORITHM(KEYPRIORITY1)
```

5. 通过发出以下命令来激活 CFRM 策略:

```
SETXCF START,POLICY,TYPE=CFRM,POLNAME=polname
```

6. 重建 IBM1SCEN1 结构。
   必须执行此过程，因为结构是在进行先前更改时分配的。
   发出以下命令以重建结构:

```
SETXCF START,REBUILD,STRNM=IBM1SCEN1
```

结果

您已成功将 SCM 添加到配置。

下一步做什么

优化系统性能。 请参阅第 316 页的『优化存储类内存使用情况』以获取更多信息。

优化存储类内存使用情况

如何改进存储类内存 (SCM) 的使用。

运行以下命令:

```
D XCF,STR,STRNAME=IBM1SCEN1
```

由于该结构已充满了消息数据，因此由于先前的测试，部分重建涉及将部分消息从该结构预登台到 SCM 中。 此进程是使用先前命令启动的。

此命令的输出生成，例如:

```
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
```

请注意命令输出中的以下内容:

- 该 STORAGE_CLASS MEMORY 确认已将 **MAXIMUM** 的 4096 MB SCM 添加到结构中。
- 用于预登台的 STORAGE-CLASS MEMORY 数量的 ALLOCATED 图。 现在，在添加 SCM 之前，结构中没有可用空间。
- 用于跟踪 SCM 使用情况的 AUGMENTED SPACE 数量。
- 预登台算法开始将数据从结构移动到 SCM 的点是当结构已满 90% 时。 这由不可配置的 **SCMHIGHTHRESHOLD** 属性指示。
- 预取算法开始将数据从 SCM 移动到结构中的以下点是在结构已满 70% 时。 这由不可配置的 **SCMLOWTHRESHOLD** 属性指示。

现在，您可以测试各种方法来优化 SCM 的使用。 请注意下列事项:

- 在使用 SCM 来存储消息之后，只有在从 SCM 中除去所有数据之后，才能更改结构。

  在这种情况下，这意味着输入/元素比率将冻结在首次使用 SCM 时的值。 在预登台算法开始将数据移动到 SCM 中之前，必须仔细确保结构处于所需的状态。
- 在使用 SCM 之前，当前结构大小是否正确?

  例如，您是否已将 **INITSIZE** 从 512 MB 增加到 1 GB 的 SIZE?

  如果不执行此操作，那么虽然您已启用自动更改结构，但预登台算法将在更改有机会启动之前开始将数据移动到 SCM 中。 因此，该结构使用 512 MB 的实存储器进行冻结。
- 在使用 SCM 之前，条目对元素的比率是否正确?

  此场景的目标是增加可作为整体存储在结构和 SCM 中的卸载消息指针数，以及尽可能将尽可能多的消息完全保存在结构存储器中。 访问这些消息比访问 SMDS 上的消息更快。

  因此，您需要具有以有利于存储消息的入口/元素比率开始的结构，然后在预登台算法首次启动之前转换为有利于存储消息指针的比率。 可以部分通过使用 IBM MQ 卸载规则来实现此转换。

通过发出以下命令来更改卸载规则:

```
ALTER CFSTRUCT(SCEN1) OFFLD1SZ(0K)
```

您可能必须执行几次运行以优化条目与元素的比率。

下表显示了在紧急存储方案的不同阶段中放入队列的消息数的可能改进。

表 *23:* 紧急存储方案的结果比较

测试描述	**消息数**	**填充队列的时间 (秒)**
基本配置	27,850	3.2
具有缺省卸载规则的 SMDS	205,000	158
具有缺省卸载规则的 SCM	828,610	469

表 *23:* 紧急存储方案的结果比较 *(*继续*)*		
测试描述	**消息数**	**填充队列的时间 (秒)**
具有调整后的卸载规则的 SCM	1,135,775	679

表中的最后一行显示调整卸载规则具有必需的效果。

您需要检查您的系统，以查看是否可以以任何方式对这些数字进行改进。 例如，您可能会耗尽可用的 SMDS 存储器。 如果可以分配更多 SMDS 存储器，那么您应该能够显着增加队列上的消息数。

改进的性能-基本配置
如何在 IBM MQ 上使用共享队列来设置基本方案以提高性能。

关于此任务

此场景描述了如何使用 SCM 来增加可以存储在共享队列上的消息数，而不会产生使用 SMDS 的性能成本。

此初始场景与用于紧急存储的场景非常相似，并使用:

- 包含单个队列管理器 CSQ3 的队列共享组 IBM1。 除管理结构外，队列共享组还定义了单个应用程序结构 SCEN2。
- 耦合设施 (CF) CF01，其中 SCEN2 应用程序结构存储为 IBM1SCEN2 结构。 此结构的最大大小为 2 GB。
- 配置为使用应用程序结构的单个共享队列 SCEN2.Q。

此配置在 第 318 页的图 65 中说明。

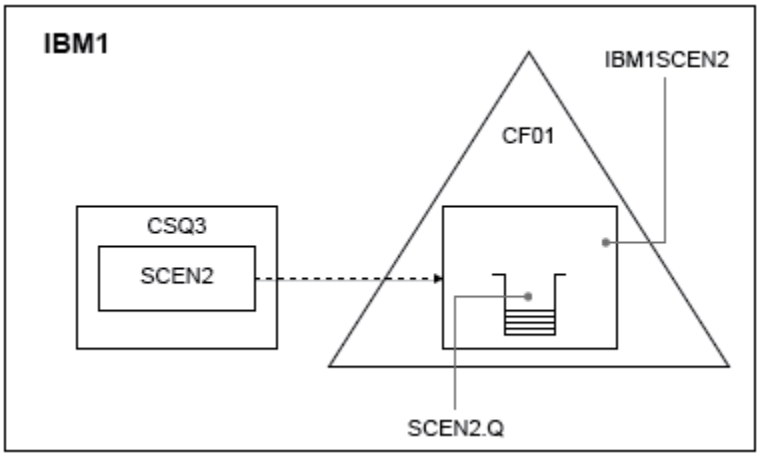


图 *65:* 基本配置

此外，假定队列管理器 CSQ3 已是队列共享组 IBM1 的唯一成员。

必须将结构 IBM1SCEN2 的定义添加到耦合设施资源管理器 (CFRM) 策略。 为了简单起见，定义了结构，以便只能通过指定 PREFLIST(CF01)在单个耦合设施 CF01 中创建该结构。

结构 IBM1SCEN2 的样本 CFRM 策略:

```
STRUCTURE
NAME(IBM1SCEN2)
SIZE(2048M)
INITSIZE(2048M)
ALLOWAUTOALT(YES)
FULLTHRESHOLD(85)
PREFLIST(CF01)
ALLOWREALLOCATE(YES)
DUPLEX(DISABLED)
ENFORCEORDER(NO)
```

INITSIZE 和 **SIZE** 关键字都具有值 2048M，因此结构无法调整大小。

过程

1. 使用以下命令刷新 CFRM 策略:

```
SETXCF START,POLICY,TYPE=CFRM,POLNAME=IBM1SCEN2
```

2. 使用以下命令验证是否已正确创建结构:

```
D XCF,STR,STRNAME=IBM1SCEN2
```

发出上述命令将提供以下输出:

```
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
```

此时，尚未将结构分配给队列共享组 (由 STATUS 行显示)。

3. 配置 IBM MQ 以使用 CFRM 策略中定义的结构。

   a. 使用带有结构名称 SCEN2 的 DEFINE CFSTRUCT 命令来创建 IBM MQ CFSTRUCT 对象。

```
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. 使用 DISPLAY CFSTRUCT 命令检查结构。

   c. 使用以下 MQSC 命令定义 SCEN2.Q 共享队列以使用 SCEN2 结构:

```
DEFINE QLOCAL(SCEN2.Q) QSGDISP(SHARED) CFSTRUCT(SCEN2) MAXDEPTH(999999999)
```

4. 使用 IBM MQ 资源管理器将单个消息放入队列 SCEN2.Q，然后再次关闭该消息。
5. 发出以下命令以检查现在是否已分配该结构:

```
D XCF,STR,STRNAME=IBM1SCEN2
```

查看命令的输出 (显示部分)，并确保 STATUS 行显示 ALLOCATED。

```
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
```

此外，请注意 SPACE USAGE 部分中字段的值:

- 条目
- 元素
- EMCS
- 锁定

以下是值的示例:

```
SPACE USAGE   IN-USE      TOTAL    %
ENTRIES:      344686     345242   99
ELEMENTS:    6548455    6548467   99
EMCS:              2     780318    0
LOCKS:          1024
```

结果

您已创建基本配置。 现在，您可以使用选择的任何方法来了解配置的基线性能。

下一步做什么

您应该测试基本方案。 例如，您可以使用以下三个应用程序，按显示的顺序启动应用程序，并同时运行这些应用程序。

1. 使用 PCF 应用程序来请求当前深度 (**CURDEPTH**) SCEN2.Q 的值 (每 5 秒)。 输出可用于绘制一段时间内队列的深度。
2. 单线程获取应用程序通过使用具有无限等待的 get 从 SCEN2.Q 重复获取消息。 为模拟已除去的消息的处理，获取应用程序将对其除去的每 10 条消息暂停 4 毫秒。
3. 单线程放置应用程序将总共 100 万条 4 KB 非持久消息放入 SCEN2.Q。 此应用程序不会在放入每条消息之间暂停，因此放入消息的速度比获取消息的应用程序获取消息的速度更快 SCEN2.Q 。

   因此，当放置应用程序正在运行时， SCEN2.Q 的深度会增加。

   填充了结构 IBM1SCEN2 ，并且放置应用程序接收到 MQRC_STORAGE_MEDIUM_FULL 原因码时，放置应用程序会在尝试将下一条消息放入队列之前休眠 5 秒。

您可以在一段时间内绘制 CURDEPTH 应用程序的结果。 在放置应用程序暂停时获取某种形式的锯齿波输出，以允许队列部分清空。

转至第 320 页的『将 SCM 添加到初始结构』。

相关概念

第 307 页的『将存储类内存与共享队列配合使用』
当与 IBM MQ for z/OS 共享队列配合使用时，使用存储类内存 (SCM) 可能是有利的。

将 *SCM* 添加到初始结构
如何添加 SCM 以提高 IBM MQ 上的性能。

关于此任务

此部分任务使用 第 318 页的『改进的性能-基本配置』中描述的基本配置。 本方案描述将 SCM 添加到初始结构。

此最终配置在 第 321 页的图 66 中说明。

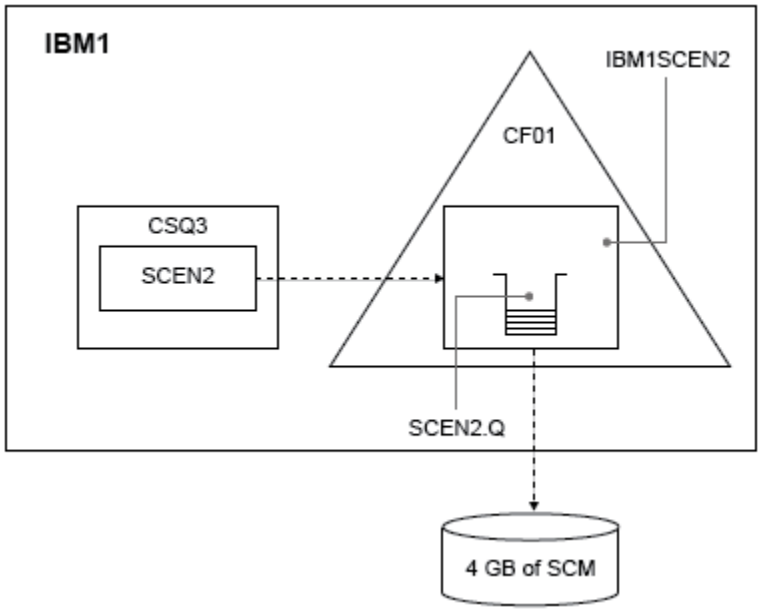


图 *66:* 配置添加 *SCM* 以提高性能

过程

1. 通过执行以下过程将 4 GB SCM 添加到结构 IBM1SCEN2 中:
   a) 通过发出以下命令，检查已安装并分配给 CF01 的 SCM 数量:

```
D CF,CFNAME=CF01
```

   b) 检查所显示输出的 STORAGE CONFIGURATION 部分中的 STORAGE-CLASS MEMORY 图以查看可用存储器。
   c) 使用 SCMMAXSIZE 和 SCMALGORITHM 关键字更新 CFRM 策略，如下所示:

```
STRUCTURE
NAME(IBM1SCEN2)
SIZE(2048M)
INITSIZE(2048M)
ALLOWAUTOALT(YES)
FULLTHRESHOLD(85)
PREFLIST(CF01)
ALLOWREALLOCATE(YES)
DUPLEX(DISABLED)
ENFORCEORDER(NO)
SCMMAXSIZE(4G)
SCMALGORITHM(KEYPRIORITY1)
```

2. 通过发出以下命令来激活 CFRM 策略:

```
SETXCF START,POLICY,TYPE=CFRM,POLNAME=IBM1SCEN2
```

3. 重建 IBM1SCEN2 结构。
   必须执行此过程，因为结构是在进行先前更改时分配的。
   发出以下命令以重建结构:

```
SETXCF START,REBUILD,STRNM=IBM1SCEN2
```

4. 发出以下命令以确认结构的新配置:

```
D XCF,STR,STRNAME=IBM1SCEN2
```

查看命令的输出，其部分如下:

```
SPACE USAGE     IN-USE       TOTAL      %
ENTRIES:          33         342684     0
ELEMENTS:         48        6503697     0
EMCS:              2         575600     0
LOCKS:                         1024
```

结果

通过增加使用 SCM 所需的控制存储器，计算实存储器使用方面的变化。

- 在将 SCM 添加到该结构之前，该结构具有以下总计，如 第 318 页的『改进的性能-基本配置』中所示:
  - 345,242 个条目
  - 6 548 467 个要素
  - 780,318 EMCS
- 将 SCM 添加到该结构后，该结构具有以下总计:
  - 342,684 个条目
  - 6 503 697 个要素
  - 575,600 EMCS

使用这些数字，在添加 SCM 后，将通过以下方法减小结构的大小:

- 2558 个条目
- 44 770 个元素
- 204,718 EMCS

对于已分配 4 GB SCM 的 2 GB 结构，用于管理 SCM 的结构存储量如下所示:

```
(2558 + 44,770 + 204,718) * 256 = 61.5 MB
```

请注意，添加更多 SCM 可能仅实现结构大小的边际缩减，因为用于跟踪 SCM 的控制存储量会随着结构大小和已分配 SCM 的数量增加而增加。

下一步做什么

重复 第 318 页的『改进的性能-基本配置』的最后部分中描述的测试。

您可以在一段时间内绘制修订后的应用程序的结果。 将图与先前获得的图进行比较，您现在获得一个没有锯齿波的输出，因为放置应用程序不再需要等待队列部分清空。

有关更多信息，请参阅 MP16: WebSphere MQ for z/OS -容量规划和调整。

分布式排队和队列共享组

分布式排队和队列共享组是可用于提高应用程序系统可用性的两种方法。 使用本主题来查找有关这些方法的更多信息。

为了补充共享队列上消息的高可用性， IBM MQ 的分布式排队组件具有其他功能来提供以下功能:

- 更高的网络可用性。

- 增加了与队列共享组的入站网络连接的容量。

第 323 页的图 67 说明了分布式排队和队列共享组。 它显示了综合系统中的两个队列管理器，这两个队列管理器都属于同一个队列共享组。 它们都可以访问共享队列 SQ1。 网络中的队列管理器 (例如，在 AIX 和 Windows 上) 可以通过任一队列管理器的通道启动程序将消息放入此队列。 两个队列管理器上的克隆应用程序都为队列提供服务。

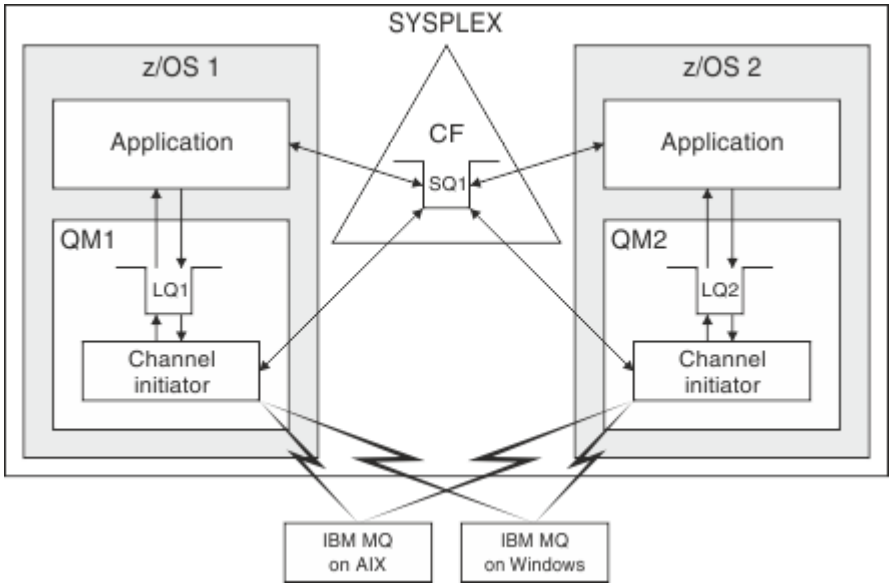


图 *67:* 分布式排队和队列共享组

相关概念

第 323 页的『共享通道』
使用本主题来了解共享通道的概念及其与 IBM MQ 8.0 for z/OS 配合使用的概念。

第 325 页的『组内排队』
组内排队允许在队列共享组中的队列管理器之间传输消息。

第 325 页的『集群和队列共享组』
使用本主题来了解如何将队列共享组与集群配合使用。

共享通道

使用本主题来了解共享通道的概念及其与 IBM MQ 8.0 for z/OS 配合使用的概念。

许多网络产品提供了一种机制来隐藏网络中的服务器故障，或者在一组符合条件的服务器之间平衡入站网络请求。 网络产品使 通用端口 可用于入站网络连接请求，并且可以通过连接到其中一个合格服务器来满足入站请求。

这些联网产品包括:

- VTAM 通用资源
- SYSPLEX 分销商

通道启动程序利用这些产品来使用共享队列的功能

有两种类型的共享通道: 共享入站通道和 共享出站通道。

- 共享入站通道
- 共享出站通道

有关渠道的更多信息，请参阅

- 共享通道摘要
- 共享通道状态

共享入站通道

队列共享组中的每个通道启动程序都会启动一个额外的侦听器任务，以在 通用端口上进行侦听。 此通用端口由其中一种支持技术 (VTAM 和 TCP/IP) 提供给网络。 对通用端口的入站网络连接请求由网络技术分派给在通用端口上侦听的队列共享组 (QSG) 中的任何一个侦听器。

如果通道启动程序有权访问具有该名称的通道的通道定义，那么可以在入站连接定向到的通道启动程序上启动通道。 您可以将通道定义定义为队列管理器专用或存储在共享存储库中，以便在任何位置提供 (全局定义)。 这意味着您可以通过将通道定义定义为全局定义，使其在队列共享组中的任何通道启动程序上可用。

通过通用端口启动通道时存在其他差异; 通道同步与队列共享组而不是与单个队列管理器同步。 例如，考虑远程队列管理器通过通用端口启动通道。 当通道首次启动时，它可能在队列管理器 QM1 和消息流上启动。 如果通道停止并在队列管理器 QM2 上重新启动，那么有关已流动的消息数的信息仍然正确，因为与队列共享组同步。

您可以使用通过通用端口启动的入站通道将消息放入任何队列。 远程队列管理器不知道目标队列是否共享。 如果目标队列是共享队列，那么远程队列管理器将以负载均衡的方式通过任何可用通道启动程序进行连接，并将消息放入共享队列。

如果目标队列是专用队列，那么会将消息放入通道的当前实例所连接到的队列管理器所拥有的专用队列。 在此环境 (称为 复制的本地队列) 中，每个队列管理器必须定义相同的专用队列集。

为队列共享组配置 SVRCONN 通道

队列共享组中 SVRCONN 通道的最佳配置是在使用从点到点通道的不同端口号的每个 CHINIT 中设置专用侦听器。 然后，这些侦听器端口将用作新的工作负载分发机制 (例如，使用虚拟 IP 地址 (VIPA) 的综合系统分发器) 的 "后端" 资源。 然后，外部 VIPA 地址将用作网络中 CLNTCONN 定义的目标地址。 可以使用 QSGDISP (GROUP) 定义 SVRCONN 通道，因此 QSG 中的所有队列管理器都可以使用相同的定义。 此配置避免使用共享侦听器，因此降低了 QSG 维护共享通道状态 (客户机/服务器通道不需要此状态) 的性能效果。

共享出站通道

如果出站通道从共享传输队列中获取消息，那么该出站通道被视为共享通道。 如果它是共享的，那么它将在队列共享组级别保存同步信息。 这意味着如果通信子系统，通道启动程序或队列管理器发生故障，那么可以在队列共享组中的其他队列管理器和通道启动程序实例上重新启动通道。 以这种方式重新启动失败的通道是称为 对等通道恢复的共享通道的功能。

共享出站通道的工作负载均衡

如果未指定要在特定通道启动程序上启动出站共享通道，那么该出站共享通道可以在队列共享组中的任何通道启动程序上启动。 IBM MQ 选择的通道启动程序是使用以下条件确定的:

- 当前需要的通信子系统对通道启动程序可用吗?
- Db2 连接是否可用于通道启动程序?
- 哪个通道启动程序的当前工作负载最低? 工作负载包括处于活动状态并正在重试的通道。

共享通道摘要

共享通道与专用通道在以下方面有所不同:

专用通道
与单个通道启动程序绑定。

- 出站通道使用本地传输队列。
- 已通过本地端口启动入站通道。
- 同步信息保存在 SYSTEM.CHANNEL.SYNCQ 队列。

共享通道
工作负载与高可用性平衡。

- 出站通道使用共享传输队列。
- 通过通用端口启动入站通道。
- 同步信息保存在 SYSTEM.QSG.CHANNEL.SYNCQ 队列。

通过将 CHLDISP 选项与 START CHANNEL 命令配合使用来启动通道时，可以指定通道是专用通道还是共享通道。 可以通过触发与专用通道相同的方式来启动共享通道。 但是，启动共享通道时， IBM MQ 会执行工作负载均衡，并在队列共享组中最合适的通道启动程序上启动该通道。 (如果需要，可以指定将在特定通道启动程序上启动共享通道。)

共享通道状态

队列共享组中的通道启动程序在 Db2 中维护共享通道状态表。 这将记录哪些通道在哪些通道启动器上处于活动状态。 如果存在通道启动程序或通信系统故障，那么将使用共享通道状态表。 它指示需要在队列共享组中的其他通道启动程序上重新启动哪些通道。

组内排队

组内排队允许在队列共享组中的队列管理器之间传输消息。

您可以在队列共享组中的队列管理器之间执行快速消息传输，而无需定义通道。 这将使用称为 SYSTEM.QSG.TRANSMIT.QUEUE，这是共享传输队列。 队列共享组中的每个队列管理器都将启动一个称为组内排队代理的任务，该任务将等待消息到达此队列中的目标队列管理器。 检测到此类消息时，会将其从队列中除去并放置在正确的目标队列上。

将使用标准名称解析规则，但如果启用了组内排队 (IGQ) ，并且目标队列管理器在队列共享组 SYSTEM.QSG.TRANSMIT.QUEUE 用于将消息传输到正确的目标队列管理器，而不是使用传输队列和通道。

通过队列管理器属性启用组内排队。 组内排队将非持久消息移动到同步点之外，并将持久消息移动到同步点内。 如果发现将消息传递到目标队列时发生问题，那么组内排队会尝试将这些消息放入死信队列。 如果死信队列已满或未定义，那么将废弃非持久消息，但持久消息将回退并返回到 SYSTEM.QSG.TRANSMIT.QUEUE 和 IGQ 代理程序尝试传递消息，直到成功。

在队列共享组中的另一个队列管理器上接收以队列为目标的消息的入站共享通道可以使用组内排队将消息 中继 到正确的目标。

有时，如果目标队列是共享队列，而不是首先将消息传输到目标队列管理器，那么您可能希望本地队列管理器将消息直接放入目标队列。 您可以使用队列管理器属性 SQQMNAME 来控制此操作。 如果将 SQQMNAME 的值设置为 USE ，那么将在 ObjectQMgr 名称指定的队列管理器上执行 MQOPEN 命令。 但是，如果目标队列是共享队列，并且您将 SQQMNAME 的值设置为 IGNORE ，并且 ObjectQMgr 名称是队列共享组中另一个队列管理器的名称，那么将在本地队列管理器上打开共享队列。 如果本地队列管理器无法打开目标队列或将消息放入队列，那么将通过 IGQ 或 MQ 通道将消息传输到指定的 ObjectQMgr 名称。

组内排队 (IGQ) 支持大型消息，最大为 100 MB 减去 传输队列头的长度。

如果使用此功能，那么用户必须对队列共享组中每个队列管理器上的队列具有相同的访问权。

集群和队列共享组

使用本主题来了解如何将队列共享组与集群配合使用。

您可以使共享队列可用于单个定义中的集群。 要执行此操作，请在定义共享队列时指定集群的名称。

网络中的用户将共享队列视为由队列共享组中的每个队列管理器托管 (不会将共享队列通告为由队列共享组托管)。 客户机可以启动与队列共享组的任何成员的会话，以将消息放入同一共享队列。

第 326 页的图 68 显示了集群成员如何通过队列共享组的任何成员访问共享队列。

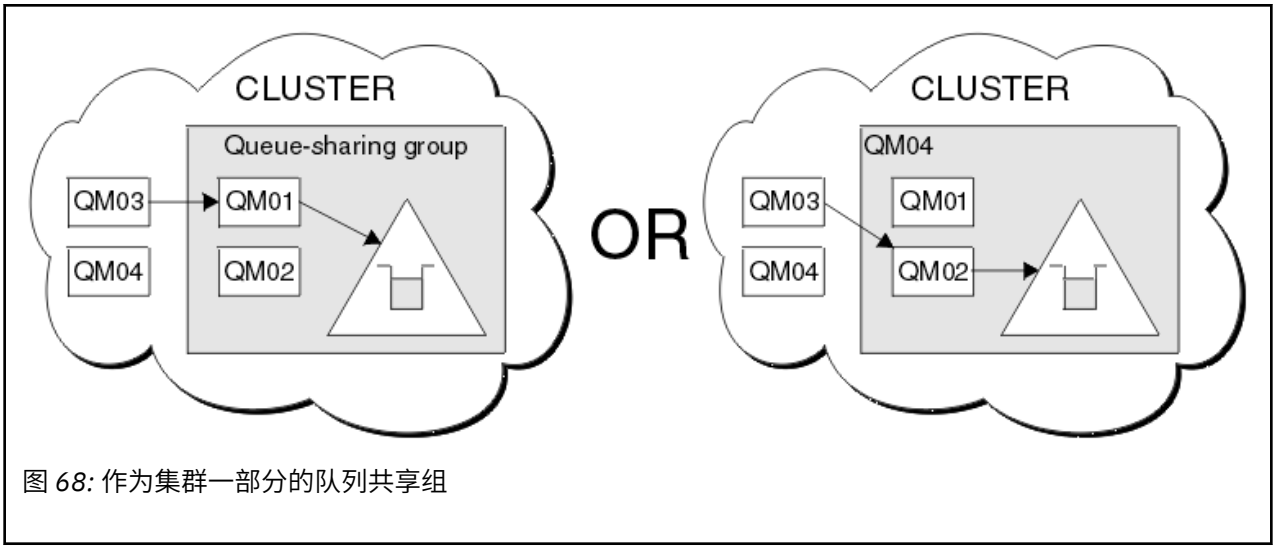


图 *68:* 作为集群一部分的队列共享组

使用共享队列影响工作负载分布

使用本主题来了解影响队列共享组中共享队列的工作负载分布的因素。

IBM MQ 未提供共享队列的工作负载均衡。 但是，队列共享组 (QSG) 中的工作负载分布可能会受到 基于拉取的方式的影响。 队列管理器服务 (接收写入共享队列的消息) 的选择受队列共享组中每个队列管理器的可用处理容量以及在综合系统中定义的工作负载管理目标的影响。

但是，很重要的一点是，执行消息的 MQPUT 的队列管理器在决定哪个队列管理器获取消息时也会产生很大影响。

本地队列管理器更有可能执行 MQGET

对于执行 MQPUT 的应用程序，本地队列管理器被认为是应用程序所连接的队列管理器。

以下注意事项会影响哪个队列管理器通过代表获取应用程序执行 MQGET 来为消息的 MQPUT 提供服务。

将消息放入空共享队列时，通常会先发布本地队列管理器，然后再通知 QSG 中的任何其他队列管理器。 如果本地队列管理器能够处理消息，那么它会在 QSG 中的任何其他队列管理器之前接收来自耦合设施 (CF) 的列表转换通知。 (列表转换通知是共享队列已将状态从 "空" 更改为 "非空" 的通知。)

在这种情况下，可能的情况如下:

1. 非持久消息的 MQPUT 不同步点， 快速放入正在等待的 *getter*。

   如果在队列的本地队列管理器上存在具有 *MQGET with wait* 的应用程序，那么消息的 MQPUT 将直接传递到获取应用程序的缓冲区，而不会写入队列。 这适用于共享队列和非共享队列。 此功能通常称为 快速放入等待的 *getter* 机制。 对于共享队列，不会通知 QSG 中的其他队列管理器，因为没有从空到非空的队列转换。 例如，这意味着如果此队列管理器可以处理来自此应用程序的所有放入，并且假定没有其他应用程序将消息放入队列，那么 QSG 中的任何其他队列管理器都不会帮助排出此队列。 但是，如果本地队列管理器上没有等待的 MQGET ，并且将消息放入共享队列，那么 CF 将根据其列表转换通知规则通知 QSG 中的其他队列管理器。
2. 持久或同步点消息的 MQPUT。

   在这种情况下，如果本地队列管理器上存在具有 *MQGET with wait* 的应用程序，那么会将消息放入共享队列中，并且 CF 会根据其通知列表转换的规则来通知 QSG 中的其他队列管理器。 但是，本地队列管理器不会等待来自 CF 的转换通知，而是首先采用任何本地 *MQGET with wait* ，并且通常代表应用程序执行此消息的获取，然后 QSG 中的任何其他队列管理器才能响应 CF 通知。 这取决于本地队列管理器的繁忙程度。 否则， CF 由于消息到达空队列而通知的任何队列管理器都将尝试先处理 get。 响应的第一个队列管理器将处理新消息。
3. 最后，如果队列未耗尽消息，其中 CF 已针对队列发送状态从 "空" 更改为 "非空" 的通知，那么所有已连接的队列管理器都将有机会帮助处理队列。 在此情况下，工作负载据说是 基于拉取的。

此设计允许在纯基于拉取的工作负载分布上提高性能。 目的是利用 CF 中保留的队列提供的高可用性，同时允许队列管理器在可能的情况下执行 MQGET ， 而不需要引用 CF ， 从而尽可能高效地处理消息工作负载。

在强调工作负载平衡比先前描述的性能增强更重要的情况下，可以采用替代方法。 例如，确保没有任何获取应用程序连接到放置应用程序所连接到的同一队列管理器。 使用此设计时，将所有消息放入队列中，当队列从空移至非空时，将根据用于处理此类转换的 CF 算法通知 QSG 中的所有队列管理器。 此外， 快速放入等待 *getter* 机制不适用。

在何处查找有关这些概念的更多信息

使用本主题中的表来查找有关本信息中心中其他主题的更多信息。

您可以从以下来源找到有关本产品文档中包含的主题的更多信息:

表 *24:* 在何处查找有关共享队列和队列共享组的更多信息

Topic	**在哪里查找信息**
队列共享组恢复	第 361 页的『恢复和重新启动』
队列共享组安全性	第 374 页的『z/OS 上的安全概念』
专用和全局对象定义 将命令定向到不同的队列 :NONE.	发出命令
规划耦合设施 环境	定义耦合设施资源
规划 SMDS 环境	规划共享消息数据集 (SMDS) 环境
规划您的 Db2 环境	规划 Db2 环境
设置共享队列 系统参数	第 291 页的『共享队列和队列共享组』
实用程序 迁移队列	使用 IBM MQ 实用程序
控制台消息	IBM MQ for z/OS 的消息
MQSC 命令	MQSC 命令
IBM MQ 集群	配置队列管理器集群
IBM MQ 分布式排队 (distributed queuing) 通道名称	分布式队列管理简介
编写应用程序	应用程序设计概述
MQCONNX 调用	MQCONNX

组内排队

本部分描述了组内排队，这是 z/OS 平台独有的 IBM MQ for z/OS 函数。 此功能仅对定义到队列共享组的队列管理器可用。

有关队列共享组的信息，请参阅 第 291 页的『共享队列和队列共享组』。

相关概念

第 328 页的『概念』
组内排队可用于更有效地将小消息传递到队列共享组中远程队列管理器上的队列。

第 330 页的『优势』
组内排队的好处包括: 减少了系统定义，减少了系统管理，提高了性能，支持在队列共享组中的队列管理器之间进行多跳时迁移和传递消息。

第 331 页的『限制』
组内排队的限制包括: 有资格使用组内排队进行传输的消息，每个队列管理器的组内排队代理程序数，以及启动和停止组内排队代理程序。

第 331 页的『入门』
您可以启用，禁用和使用组内排队，如本主题中所述。

第 332 页的『配置』
除了典型的组内排队配置外，还可以进行其他配置。

第 336 页的『组内排队消息』
本部分描述了放入 SYSTEM.QSG.TRANSMIT.QUEUE。

第 338 页的『安全性』
本主题描述组内排队的安全安排。

第 338 页的『特定属性』
本部分描述了组内排队的特定属性。

相关参考

第 330 页的『组内排队的术语』
术语解释: 组内排队，供组内排队使用的共享传输队列，组内排队代理。

概念

组内排队可用于更有效地将小消息传递到队列共享组中远程队列管理器上的队列。

组内排队 (IGQ) 可以在队列共享组 (QSG) 中的队列管理器之间进行潜在的快速且成本较低的小型消息传输，而无需在队列管理器之间定义通道。

下图显示了组内排队的典型示例。

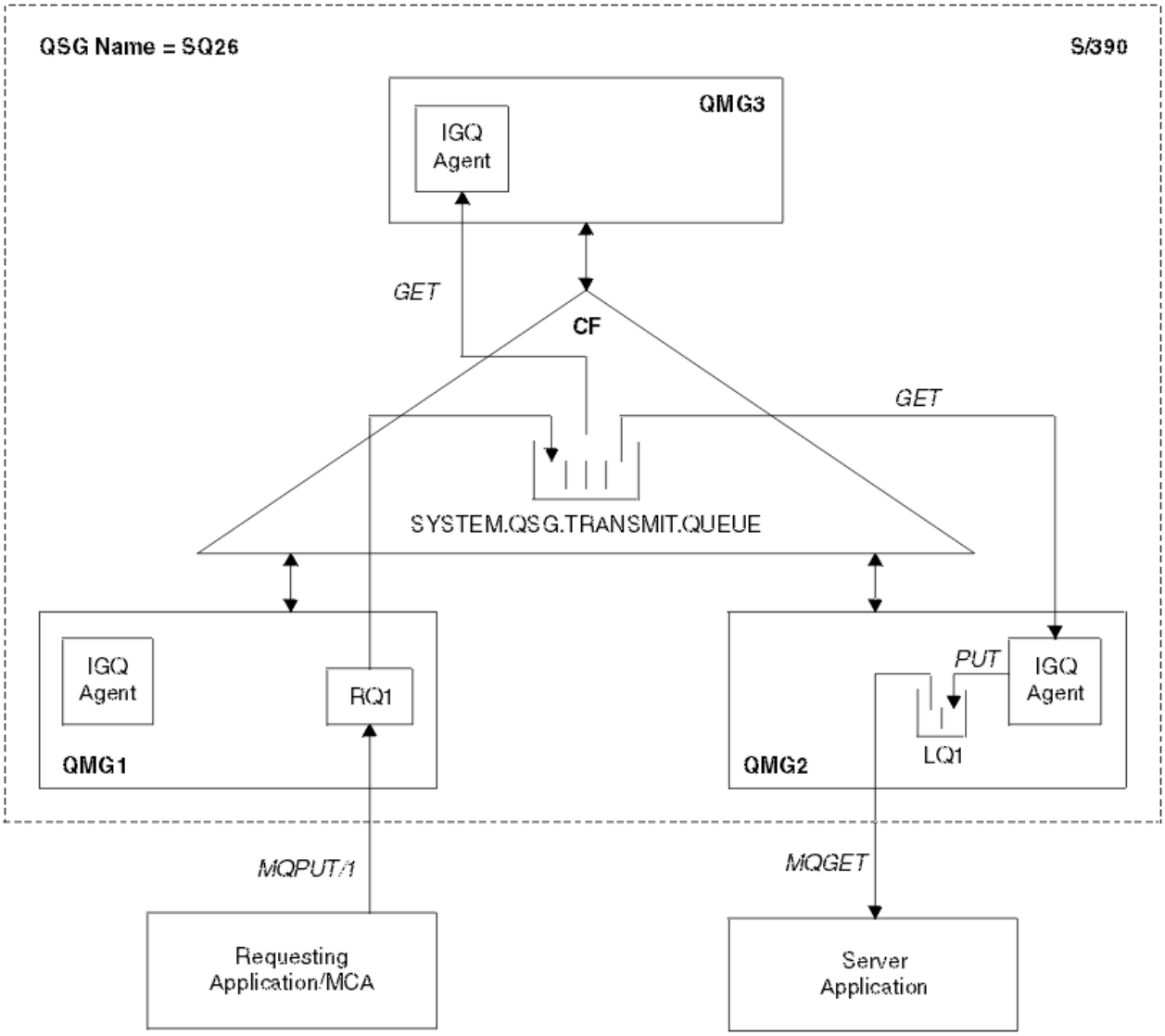


图 *69:* 组内排队的示例

图显示:

- 在定义到名为 SQ26 的队列共享组的三个队列管理器 (QMG1， QMG2 和 QMG3) 上运行的 IGQ 代理程序。
- 共享传输队列 SYSTEM.QSG.TRANSMIT.QUEUE 。
- 在队列管理器 QMG1 中定义的远程队列定义。
- 在队列管理器 QMG2 中定义的本地队列。
- 请求应用程序 (此应用程序可能是连接到队列管理器 QMG1 的消息通道代理程序 (MCA))。
- 连接到队列管理器 QMG2 的服务器应用程序。
- 放置到 SYSTEM.QSG.TRANSMIT.QUEUE。

组内排队和组内排队代理

在队列管理器初始化期间启动 IGQ 代理程序。 当应用程序打开消息并将其放入远程队列时，本地队列管理器确定是否将组内排队用于消息传输。 如果要使用组内排队，那么本地队列管理器会将消息放入 SYSTEM.QSG.TRANSMIT.QUEUE。 目标远程队列管理器上的 IGQ 代理程序检索消息并将其放入目标队列。

组内排队的术语

术语解释: 组内排队，供组内排队使用的共享传输队列，组内排队代理。

组内排队

组内排队可以在队列共享组中的队列管理器之间进行潜在的快速且成本较低的消息传输，而无需定义通道。

供组内排队使用的共享传输队列

每个队列共享组都有一个名为 SYSTEM.QSG.TRANSMIT.QUEUE ，供组内排队使用。 如果启用组内排队，那么 SYSTEM.QSG.TRANSMIT.QUEUE 将显示在名称解析路径中。 当应用程序 (包括消息通道代理程序 (MCA)) 将消息放入远程队列时，本地队列管理器将确定消息的快速传输资格，并将其放在 SYSTEM.QSG.TRANSMIT.QUEUE。

组内排队代理程序

IGQ 代理程序是在队列管理器初始化时启动的任务，用于等待合适的消息到达 SYSTEM.QSG.TRANSMIT.QUEUE。 IGQ 代理程序从此队列中检索适当的消息，并将它们传递到目标队列。

每个队列管理器的 IGQ 代理程序始终启动，因为组内排队由队列管理器本身用于其自己的内部处理。

优势

组内排队的好处包括: 减少了系统定义，减少了系统管理，提高了性能，支持在队列共享组中的队列管理器之间进行多跳时迁移和传递消息。

组内排队的好处有:

减少的系统定义
组内排队将不需要在队列共享组中的队列管理器之间定义通道。

减少系统管理
由于队列共享组中的队列管理器之间未定义任何通道，因此不需要进行通道管理。

提高了性能
因为只有一个 IGQ 代理需要将消息传递到目标队列 (而不是两个中间发送方和接收方代理) ，所以使用组内排队传递消息的成本可能低于使用通道传递消息的成本。 在组内排队中，只有一个接收组件，因为已除去对发送组件的需求。 此保存是因为消息可供目标队列管理器上的 IGQ 代理在本地队列管理器上的放置操作完成后传递到目标队列，并且在消息放入同步点作用域的情况下，已落实。

支持迁移

队列共享组外部的应用程序可以将消息传递到驻留在队列共享组中的任何队列管理器上的队列，而仅连接到队列共享组中的特定队列管理器。 这是因为到达接收方通道 (针对远程队列管理器上的队列) 的消息可以使用组内排队以透明方式发送到目标队列。 此工具允许在队列共享组中部署应用程序，而无需更改队列共享组外部的任何系统。

下图说明了典型配置，其中:

- 连接到队列管理器 QMG1 的请求应用程序需要将消息发送到队列管理器 QMG3 上的本地队列。
- 队列管理器 QMG1 仅连接到队列管理器 QMG2。
- 先前使用通道连接的队列管理器 QMG2 和 QMG3 现在是队列共享组 SQ26 的成员。

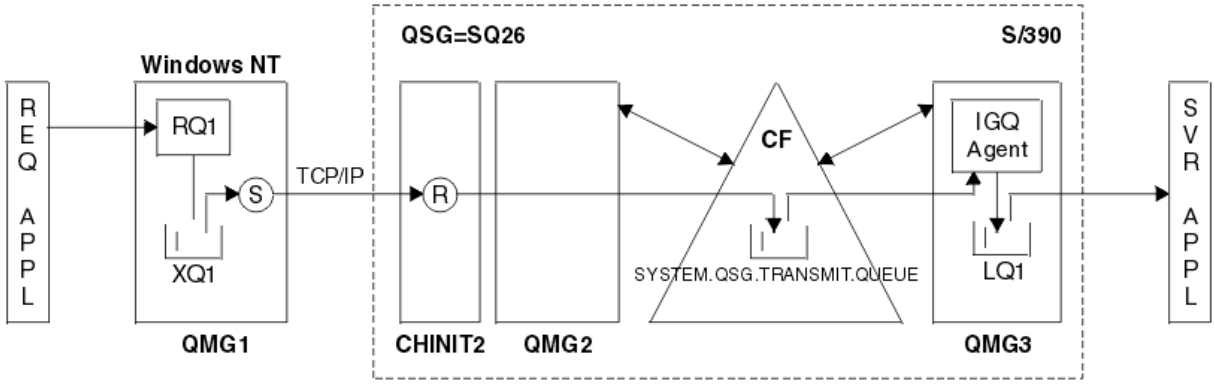


图 *70:* 迁移支持示例

操作流程如下所示:

1. 请求应用程序将以远程队列管理器 QMG3 上的本地队列 LQ1 为目标的消息放入远程队列定义 RQ1。
2. 在 Windows NT 工作站上运行的队列管理器 QMG1 将消息放入传输队列 XQ1。
3. QM1 上的发送方 MCA (S) 使用 TCP/IP 将消息传输到通道启动程序 CHINIT2 上的接收方 MCA (R)。
4. 通道启动程序 CHINIT2 上的接收方 MCA (R) 将消息放入共享传输队列 SYSTEM.QSG.TRANSMIT.QUEUE。
5. 队列管理器 QMG3 上的 IGQ 代理程序从 SYSTEM.QSG.TRANSMIT.QUEUE 并将其放入目标本地队列 LQ1。
6. 服务器应用程序从目标本地队列检索消息并对其进行处理。

在队列共享组中的队列管理器之间进行多跳时传递消息

支持迁移 中的上图还说明了在队列共享组中的队列管理器之间进行多跳跃时传递消息的情况。 可以使用组内排队将到达队列共享组中的队列管理器上的消息，但这些消息的目标是队列共享组中的另一个队列管理器上的队列，可以轻松地将这些消息传输到目标队列管理器上的目标队列。

限制

组内排队的限制包括: 有资格使用组内排队进行传输的消息，每个队列管理器的组内排队代理程序数，以及启动和停止组内排队代理程序。

本主题描述组内排队的限制。

有资格使用组内排队传输的消息

由于组内排队使用耦合设施 (CF) 中定义的共享传输队列，因此组内排队仅限于传递共享队列的最大受支持消息长度减去传输队列头 (MQXQH) 长度的消息。

每个队列管理器的组内排队代理程序数

队列共享组中的每个队列管理器仅启动一个 IGQ 代理程序。

启动和停止组内排队代理程序

IGQ 代理程序在队列管理器初始化期间启动，并在队列管理器关闭期间终止。 它被设计为长时间运行，自我恢复 (在异常终止的情况下)，任务。 如果 SYSTEM.QSG.TRANSMIT.QUEUE (例如，如果此队列为 "禁止获取")，那么 IGQ 代理程序将继续重试。 如果 IGQ 代理程序迁到导致在队列管理器仍处于活动状态时代理程序正常终止的错误，那么可以通过发出 ALTER QMGR IGQ (ENABLED) 命令来重新启动该代理程序。 此命令可避免需要重新启动队列管理器。

入门

您可以启用，禁用和使用组内排队，如本主题中所述。

启用组内排队

要在队列管理器上启用组内排队，需要执行以下操作:

- 定义名为 SYSTEM.QSG.TRANSMIT.QUEUE。 此队列的定义可以在 thlqual.SCSQPROCS(CSQ4INSS) 中找到，即用于队列共享组的 SYSTEM 对象的 CSQINP2 样本。 必须使用正确的属性定义此队列，如 thlqual.SCSQPROCS(CSQ4INSS) 中所述，才能使组内排队正常工作。
- 由于 IGQ 代理程序始终在队列管理器初始化时启动，因此组内排队始终可用于入站消息处理。 IGQ 代理程序处理放在 SYSTEM.QSG.TRANSMIT.QUEUE。 但是，要对出站处理启用组内排队，必须将队列管理器属性 IGQ 设置为 ENABLED。

要点: 虽然您可以动态启用已连接的应用程序，但在执行此操作时，您会收到 MQRC 2041。 您应该有一个机制来正确处理此情况。 请参阅第 338 页的『对象句柄失效 (MQRC_OBJECT_CHANGED)』以获取更多信息。

此外，由于 IGQ 设计为长时间运行的自恢复任务，在初始化期间启动并在关闭时终止，请参阅第 331 页的『限制』以获取更多信息。

禁用组内排队

要对出站消息传输禁用组内排队，请将队列管理器属性 IGQ 设置为 DISABLED。 如果对特定队列管理器禁用组内排队，那么该队列管理器上的 IGQ 代理程序仍可以处理已放置在 SYSTEM.QSG.TRANSMIT.QUEUE ，该队列管理器已启用组内排队以进行出站传输。

要点: 虽然您可以动态禁用已连接的应用程序，但在执行此操作时，您会收到 MQRC 2041。 您应该有一个机制来正确处理此情况。 请参阅第 338 页的『对象句柄失效 (MQRC_OBJECT_CHANGED)』以获取更多信息。

此外，由于 IGQ 设计为长时间运行的自恢复任务，在初始化期间启动并在关闭时终止，请参阅第 331 页的『限制』以获取更多信息。

使用组内排队

一旦启用了组内排队，就可以使用该队列，并且队列管理器将尽可能使用该队列。 即，当应用程序将消息放入远程队列定义，放入标准远程队列或放入集群队列时，队列管理器将确定消息是否符合使用组内排队传递的条件，如果符合，那么将消息放入 SYSTEM.QSG.TRANSMIT.QUEUE。 不需要更改用户应用程序或应用程序队列，因为对于符合条件的消息，队列管理器使用 SYSTEM.QSG.TRANSMIT.QUEUE，优先于任何其他传输队列。

配置

除了典型的组内排队配置外，还可以进行其他配置。

第 329 页的图 69 描述了典型配置。

相关概念

第 332 页的『分布式排队与组内排队 (多个交付路径)』
对于处理短消息的应用程序，配置组内排队 (仅用于在队列共享组中的队列管理器之间传递消息) 可能是可行的。

第 334 页的『集群与组内排队 (多个交付路径)』
可以配置队列管理器，以便它们既位于集群中，也位于队列共享组中。

第 336 页的『集群，组内排队和分布式排队』
可以配置作为集群成员以及队列共享组的队列管理器，并使用发送方/接收方通道对连接到分布式队列管理器。

分布式排队与组内排队 (多个交付路径)

对于处理短消息的应用程序，配置组内排队 (仅用于在队列共享组中的队列管理器之间传递消息) 可能是可行的。

通过通道通信进行组内排队的选择可由 CFSTRUCT 类型级别控制。 (3 而不是 4 或 5)。 在 SYSTEM.QSQ.TRANSMIT.QUEUE。

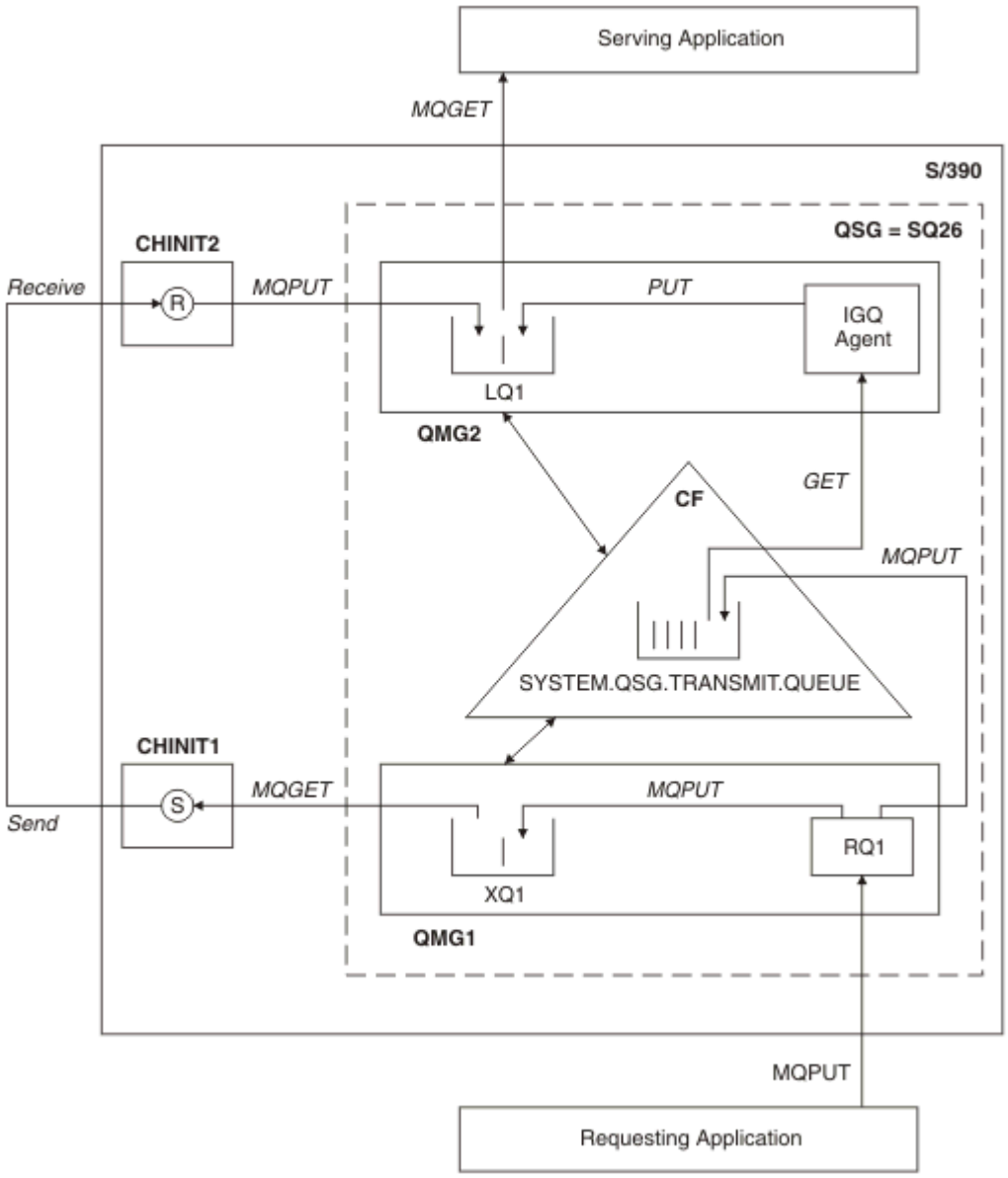


图 *71:* 示例配置

打开/放入处理

1. 请务必注意，当发出请求的应用程序打开远程队列 RQ1 时，将对非共享传输队列 XQ1 和共享传输队列 SYSTEM.QSG.TRANSMIT.QUEUE。
2. 当请求应用程序将消息放入远程队列时，根据是否对队列管理器上的出站传输启用组内排队以及消息特征，将消息放入传输队列 XQ1 或传输队列 SYSTEM.QSG.TRANSMIT.QUEUE。 队列管理器将所有大型消息放在传输队列 XQ1 上，并将所有小型消息放在传输队列 SYSTEM.QSG.TRANSMIT.QUEUE。
3. 如果传输队列 XQ1 已满或不可用，那么针对大型消息的放入请求将同步失败，并返回适当的返回码和原因码。 但是，针对小消息的放入请求将继续成功，并将其放入传输队列 SYSTEM.QSG.TRANSMIT.QUEUE。
4. 如果传输队列为 SYSTEM.QSG.TRANSMIT.QUEUE 已满，或者无法放入，针对小型消息的放入请求将同步失败，并返回适当的返回码和原因码。 但是，针对大型消息的放入请求将继续成功，并将其放入传输队列 XQ1。 在这种情况下，不会尝试将小消息放入传输队列。

大消息流

1. 请求应用程序将大型消息放入远程队列 RQ1。
2. 队列管理器 QMG1 将消息放入传输队列 XQ1。
3. 队列管理器 QMG1 上的发送方 MCA (S) 从传输队列 XQ1 检索消息，并将它们发送到队列管理器 QMG2。
4. 队列管理器 QMG2 上的接收方 MCA (R) 接收消息并将其放入目标队列 LQ1。
5. 服务应用程序从队列 LQ1 检索消息，然后进行处理。

小消息流

1. 请求应用程序将小消息放入远程队列 RQ1。
2. 队列管理器 QMG1 将消息放入传输队列 SYSTEM.QSG.TRANSMIT.QUEUE。
3. 队列管理器 QMG2 上的 IGQ 检索消息并将其放入目标队列 LQ1。
4. 服务应用程序从队列 LQ1 检索消息。

要考虑的要点

1. 请求应用程序不需要知道用于传递消息的底层机制。
2. 可以为小消息实现潜在的更快的消息传递机制。
3. 多个路径可用于消息传递 (即，正常通道路由和组内排队路由)。
4. 选择组内排队路径 (可能更快) 优先于正常通道路径。 根据消息特征，消息传递可能分为两条路径。 因此，消息可能会按顺序传递 (尽管如果仅使用正常通道路由传递消息，那么也可以进行此传递)。
5. 当选择了路由并且已将消息放入传输队列时，仅将所选路由用于消息传递。 SYSTEM.QSG.TRANSMIT.QUEUE 未转至传输队列 XQ1。

集群与组内排队 (多个交付路径)

可以配置队列管理器，以便它们既位于集群中，也位于队列共享组中。

当消息发送到集群队列并且本地和远程目标队列管理器位于同一个队列共享组中时，组内排队用于传递小消息 (使用 SYSTEM.QSG.TRANSMIT.QUEUE) ，以及在组内排队支持消息大小的情况下传递大型消息。 另外，SYSTEM.CLUSTER.TRANSMIT.QUEUE 用于将消息传递到集群中但在队列共享组外部的任何队列管理器。 下图说明了此配置 (未显示通道启动程序)。

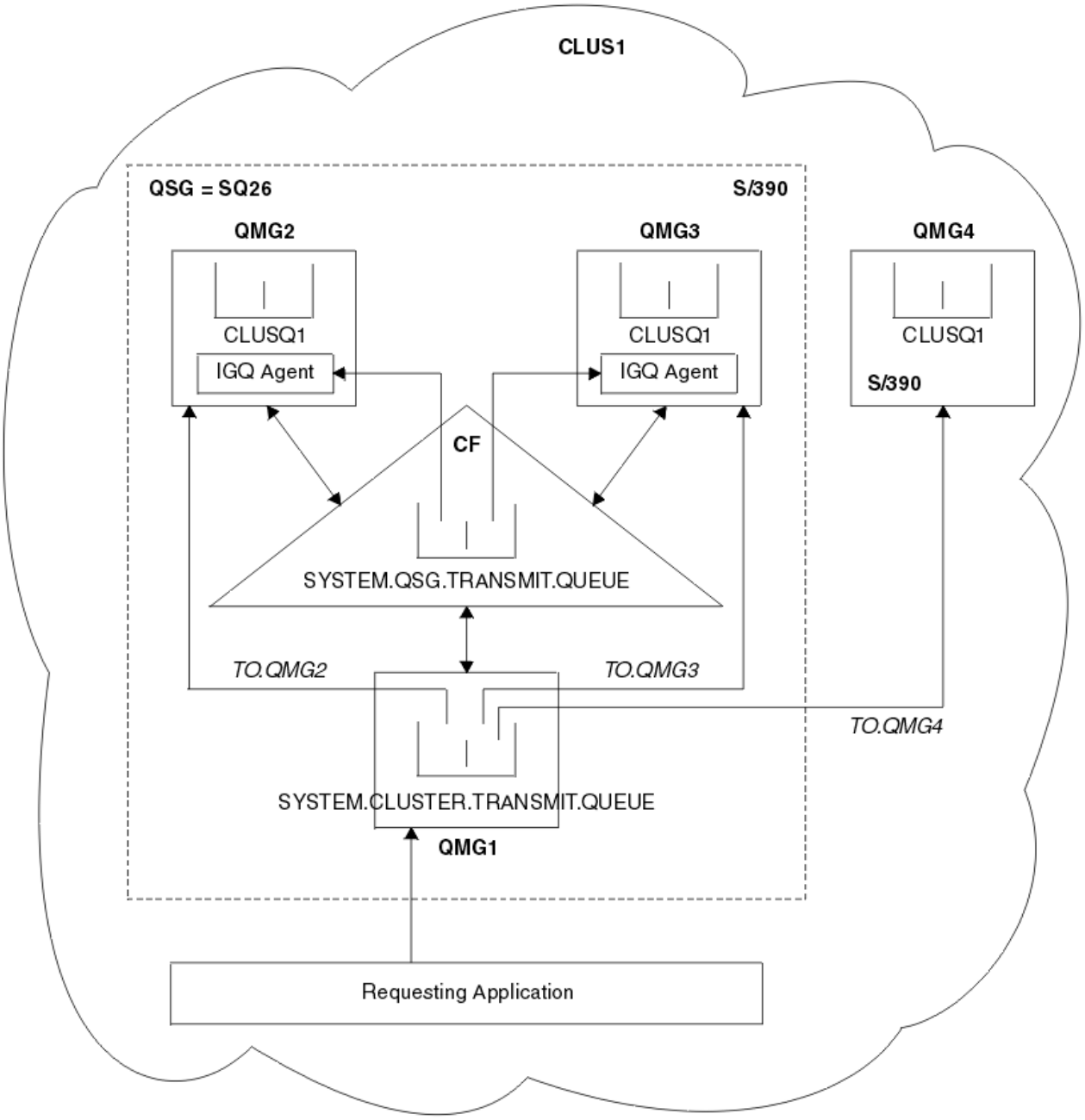


图 *72:* 使用组内排队的集群示例

图显示:

- 集群 CLUS1 中配置了四个 z/OS 队列管理器 QMG1， QMG2， QMG3 和 QMG4 。
- 队列共享组 SQ26 中配置的队列管理器 QMG1， QMG2 和 QMG3 。
- 在队列管理器 QMG2 和 QMG3 上运行的 IGQ 代理程序。
- 在 QMG1 中定义的本地 SYSTEM.CLUSTER.TRANSMIT.QUEUE 。

  注: 为了清晰起见，请参阅 SYSTEM.CLUSTER.TRANSMIT.QUEUE 。
- 共享 SYSTEM.QSG.TRANSMIT.QUEUE ， 它位于使用 CFLEVEL ($TAG3) RECOVER (YES) 属性配置的 IBM MQ 结构中。
- 集群通道 TO.QMG2 (将 QMG1 连接到 QMG2) ， TO.QMG3 (将 QMG1 连接到 QMG3) 和 TO.QMG4 (将 QMG1 连接到 QMG4)。

- 正在队列管理器 QMG2， QMG3 和 QMG4 上托管集群队列 CLUSQ1 。

假定发出请求的应用程序使用 MQOO_BIND_NOT_FIXED 选项打开集群队列，以便在放置时选择集群队列的目标队列管理器。

如果所选目标队列管理器为 QMG2:

- 请求应用程序放入的所有大型消息为:
  - 放入 SYSTEM.CLUSTER.TRANSMIT.QUEUE on QMG1，因为 SYSTEM.QSG.TRANSMIT.QUEUE 处于 CFLEVEL (3) 结构中; 因此仅支持最大大小为 63 KB 的消息。
  - 已使用集群通道 TO.QMG2 传输到 QMG2 上的集群队列 CLUSQ1
- 请求应用程序放入的所有小消息都是
  - 放入共享传输队列 SYSTEM.QSG.TRANSMIT.QUEUE。 此队列采用配置了 RECOVER (YES) 属性的结构，因此用于持久和非持久的小消息。
  - 由 QMG2 上的 IGQ 代理程序检索
  - 放入 QMG2 上的集群队列 CLUSQ1

如果所选目标队列管理器为 QMG4:

- 由于 QMG4 不是队列共享组 SQ26 的成员，因此请求应用程序放入的所有消息都是
  - 放入 QMG1 上的 SYSTEM.CLUSTER.TRANSMIT.QUEUE
  - 已使用集群通道 TO.QMG4 传输到 QMG4 上的集群队列 CLUSQ1

要考虑的要点

- 请求应用程序不需要知道用于传递消息的底层机制。
- 对于在队列共享组中的队列管理器之间传输小型非持久消息 (即使同一队列管理器位于集群中) ，可能会实现更快的传递机制。
- 多个路径可用于消息传递 (即，集群路由和组内排队路由)。
- 选择组内排队路径 (可能更快) 优先于集群路径。 根据消息特征，消息传递可能分为两条路径。 因此，可能会按顺序传递消息。 请务必注意，此传递是可能的，而不考虑应用程序指定的 MQOO_BIND_ * 选项。 根据是在打开时指定 MQOO_BIND_NOT_FIXED ， MQOO_BIND_ON_OPEN ， MQOO_BIND_ON_GROUP 还是 MQOO_BIND_AS_Q_DEF ，组内排队以集群相同的方式分发消息。
- 当选择了路由并且已将消息放入传输队列时，仅将所选路由用于消息传递。 SYSTEM.QSG.TRANSMIT.QUEUE 不会转移到 SYSTEM.CLUSTER.TRANSMIT.QUEUE。

集群，组内排队和分布式排队

可以配置作为集群成员以及队列共享组的队列管理器，并使用发送方/接收方通道对连接到分布式队列管理器。

此配置是分布式排队与组内排队以及集群与组内排队的组合。

第 332 页的『分布式排队与组内排队 (多个交付路径)』中描述了组内排队。

第 334 页的『集群与组内排队 (多个交付路径)』中描述了使用组内排队的集群。

组内排队消息

本部分描述了放入 SYSTEM.QSG.TRANSMIT.QUEUE。

消息结构

与放入传输队列的所有其他消息一样，放入 SYSTEM.QSG.TRANSMIT.QUEUE 以传输队列头 (MQXQH) 作为前缀。

消息持久性

在 IBM MQ V 5 发行版 3 及更高版本中，共享队列同时支持持久消息和非持久消息。

如果队列管理器在 IGQ 代理程序处理非持久消息时终止，或者如果 IGQ 代理程序在处理消息期间异常终止，那么正在处理的非持久消息可能会丢失。 如果需要恢复非持久消息，应用程序必须作出恢复安排。

如果 IGQ 代理程序发出的非持久消息的放置请求意外失败，那么正在处理的消息将丢失。

传递消息
IGQ 代理程序检索并传递同步点作用域外的所有非持久消息以及同步点作用域内的所有持久消息。 在这种情况下， IGQ 代理程序充当同步点协调程序。 因此， IGQ 代理程序会处理非持久消息，例如在消息通道上快速处理非持久消息的方式。 请参阅 快速非持久消息 (Fast ， nonpersistent messages)。

消息批处理
IGQ 代理程序使用 50 条消息的固定批处理大小。 在批处理中检索到的任何持久消息将按 50 条消息的时间间隔落实。 当 SYSTEM.QSG.TRANSMIT.QUEUE。

消息大小
可以放入 SYSTEM.QSG.TRANSMIT.QUEUE 是共享队列支持的最大消息长度减去传输队列头 (MQXQH) 的长度。

缺省消息持久性和缺省消息优先级
如果是 SYSTEM.QSG.TRANSMIT.QUEUE 位于在打开时建立的队列名称解析路径中，然后对于具有缺省持久性和缺省优先级 (或者具有缺省持久性或缺省优先级) 的消息，将在选择具有所使用的缺省优先级和持久性值的队列时应用常规规则。 (请参阅 IBM MQ 消息 部分，以获取有关队列选择规则的更多信息)。

相关概念
第 337 页的『未传递/未处理的消息』
本主题描述在 SYSTEM.QSG.TRANSMIT.QUEUE。

第 337 页的『报告消息-组内排队』
本主题描述报告消息: 到达确认， 交付确认，到期报告和异常报告。

未传递/未处理的消息
本主题描述在 SYSTEM.QSG.TRANSMIT.QUEUE。

如果 IGQ 代理程序无法将消息传递到目标队列，那么 IGQ 代理程序:

- 采用 MQRO_DISCARD_MSG 报告选项 (如果未传递的消息的 MQMD 的 "报告选项" 字段指示它必须) 并废弃未传递的消息。
- 尝试将未传递的消息放在目标队列管理器的死信队列上 (如果尚未废弃该消息)。 IGQ 代理程序以死信队列头 (MQDLH) 作为消息的前缀。

如果未定义死信队列，或者如果无法将未传递的消息放入死信队列，并且未传递的消息为:

- 持久， IGQ 代理程序将其正在处理的当前持久消息批处理回退，并进入重试状态。 有关更多信息，请参阅第 339 页的『组内排队代理程序的重试功能』。
- 非持久， IGQ 代理程序将废弃该消息并继续处理下一条消息。

如果队列共享组中的队列管理器在其关联的 IGQ 代理程序有时间处理其所有消息之前终止，那么未处理的消息将保留在 SYSTEM.QSG.TRANSMIT.QUEUE ， 直到下次启动队列管理器为止。 然后， IGQ 代理程序会检索消息并将其传递到目标队列。

如果耦合设施在 SYSTEM.QSG.TRANSMIT.QUEUE ， 任何未处理的非持久消息都将丢失。

IBM 建议应用程序不要将消息直接放入传输队列。 如果应用程序将消息直接放入 SYSTEM.QSG.TRANSMIT.QUEUE， IGQ 代理程序可能无法处理这些消息，它们将保留在 SYSTEM.QSG.TRANSMIT.QUEUE。 然后，用户必须使用自己的方法来处理这些未处理的消息。

报告消息-组内排队
本主题描述报告消息: 到达确认， 交付确认，到期报告和异常报告。

确认到达 (COA)/确认交付 (COD) 报告消息
使用组内排队时，队列管理器会生成 COA 和 COD 消息。

到期报告消息
到期报告消息由队列管理器生成。

异常报告消息

根据未传递消息的消息描述符的 报告选项 字段中指定的 MQRO_EXCEPTION_ * 报告选项， IGQ 代理程序将生成所需的异常报告并将其放置在指定的应答队列上。 组内排队可用于将异常报告传递到目标应答队列。

报告消息的持久性与未传递消息的持久性相同。 如果 IGQ 代理未能解析目标应答队列的名称，或者如果它未能将应答消息放入传输队列 (用于后续传输至目标应答队列) ，那么它会尝试将异常报告放入生成报告消息的队列管理器的死信队列。 如果不可能，那么如果未传递的消息为:

- 持久， IGQ 代理程序会废弃异常报告，回退当前消息批次，并进入重试状态。 有关更多信息，请参阅第 339 页的『组内排队代理程序的重试功能』。
- 非持久性， IGQ 代理程序将废弃异常报告并继续处理 SYSTEM.QSG.TRANSMIT.QUEUE。

安全性

本主题描述组内排队的安全安排。

可以设置队列管理器属性 IGQAUT (IGQ 权限) 和 IGQUSER (IGQ 代理程序用户标识) ， 以控制当 IGQ 代理程序打开目标队列时执行的安全性检查级别。

组内排队权限 (IGQAUT)

可以设置 IGQAUT 属性以指示要执行的安全性检查的类型，从而确定 IGQ 代理程序在建立将消息放入目标队列的权限时要使用的用户标识。

IGQAUT 属性类似于通道定义上可用的 PUTAUT 属性。

组内排队用户标识 (IGQUSER)

IGQUSER 属性可用于指定 IGQ 代理程序在建立将消息放入目标队列的权限时要使用的用户标识。

IGQUSER 属性类似于通道定义上可用的 MCAUSER 属性。

特定属性

本部分描述了组内排队的特定属性。

相关概念

第 338 页的『对象句柄失效 (MQRC_OBJECT_CHANGED)』
如果在打开对象后发现对象的属性已更改，那么队列管理器将在下次使用 MQRC_OBJECT_CHANGED 时使对象句柄失效。

第 339 页的『组内排队代理的自我恢复』
如果 IGQ 代理程序异常终止，那么将发出消息 CSQM067E ，并且 IGQ 代理程序将再次启动。

第 339 页的『组内排队代理程序的重试功能』
如果 IGQ 代理程序在访问 SYSTEM.QSG.TRANSMIT.QUEUE (例如，由于未定义此代理程序，或者未使用不正确的属性定义此代理程序，或者由于某些其他原因而禁止此代理程序) ，因此 IGQ 代理程序将进入重试状态。

第 339 页的『组内排队代理程序和序列化』
当对等恢复仍在进行时， IGQ 代理程序尝试序列化对共享队列的访问可能会失败。

对象句柄失效 *(MQRC_OBJECT_CHANGED)*

如果在打开对象后发现对象的属性已更改，那么队列管理器将在下次使用 MQRC_OBJECT_CHANGED 时使对象句柄失效。

组内排队引入了对象句柄失效的以下新规则:

- 如果是 SYSTEM.QSG.TRANSMIT.QUEUE 包含在名称解析路径中，因为组内排队在打开时为 ENABLED ，但在放置时发现组内排队为 DISABLED ，那么队列管理器会使对象句柄失效，并使使用 MQRC_OBJECT_CHANGED 的放置请求失败。
- 如果是 SYSTEM.QSG.TRANSMIT.QUEUE 包含在名称解析路径中，因为组内排队在打开时已禁用，但发现组内排队在放置时已启用，那么队列管理器将使对象句柄失效，并使带有 MQRC_OBJECT_CHANGED 的放置请求失败。

- 如果是 SYSTEM.QSG.TRANSMIT.QUEUE 在打开处理期间包含在名称解析路径中，因为组内排队是在打开时启用的，但 SYSTEM.QSG.TRANSMIT.QUEUE 定义已按放置时间进行更改，那么队列管理器将使对象句柄失效，并使带有 MQRC_OBJECT_CHANGED 的放置请求失败。

组内排队代理的自我恢复

如果 IGQ 代理程序异常终止，那么将发出消息 CSQM067E ，并且 IGQ 代理程序将再次启动。

组内排队代理程序的重试功能

如果 IGQ 代理程序在访问 SYSTEM.QSG.TRANSMIT.QUEUE (例如，由于未定义此代理程序，或者未使用不正确的属性定义此代理程序，或者由于某些其他原因而禁止此代理程序) ，因此 IGQ 代理程序将进入重试状态。

IGQ 代理程序会观察短期和长期重试计数和时间间隔。 这些计数和时间间隔的值 (不可更改) 如下所示:

constant	值
短重试次数	10
短重试时间间隔	60 秒 = 1 分钟
长重试次数	999,999,999
长重试时间间隔	1200 秒 = 20 分钟

组内排队代理程序和序列化

当对等恢复仍在进行时， IGQ 代理程序尝试序列化对共享队列的访问可能会失败。

如果当 IGQ 代理程序正在处理一个或多个共享队列上未落实的消息时，队列共享组中的队列管理器发生故障，那么 IGQ 代理程序将结束，并且将对发生故障的队列管理器进行共享队列对等恢复。 由于共享队列对等恢复是一项异步活动，因此在共享队列对等恢复完成之前，可能会使失败的队列管理器以及该队列管理器的 IGQ 代理程序重新启动。 这将使任何已落实的消息都有可能在仍在恢复的情况下提前和不按顺序进行处理。 为了确保不按顺序处理消息， IGQ 代理程序通过发出 MQCONNX API 调用来序列化对共享队列的访问。

当对等恢复仍在进行时， IGQ 代理程序尝试序列化对共享队列的访问可能会失败。 发出错误消息，并将 IGQ 代理程序置于重试状态。 当队列管理器对等恢复完成时 (例如，在下次重试时) ， IGQ 代理程序可以启动。

存储管理

IBM MQ for z/OS 需要永久和临时数据结构，并使用页集和内存缓冲区来存储此数据。 这些主题提供了有关 IBM MQ 如何利用这些页集和缓冲区的更多详细信息。

相关概念

第 339 页的『页面集合』
使用本主题来了解 IBM MQ for z/OS 如何使用页面集来存储消息。

第 340 页的『存储类』
存储类 是一个 IBM MQ for z/OS 概念，允许队列管理器将队列映射到页集。 您可以使用存储类来控制哪些数据集由哪些队列使用。

第 341 页的『缓冲区和缓冲池』
IBM MQ for z/OS 使用缓冲区和缓冲池来临时高速缓存数据。 使用本主题来进一步了解如何组织和使用缓冲区。

相关参考

第 342 页的『在何处查找有关存储管理的更多信息』
使用本主题作为参考，以查找有关 IBM MQ for z/OS 的存储管理的更多信息。

页面集合

使用本主题来了解 IBM MQ for z/OS 如何使用页面集来存储消息。

页集 是经过特殊格式化以供 IBM MQ 使用的 VSAM 线性数据集。 页集用于存储大多数消息和对象定义。

这方面的例外是全局定义，这些定义存储在 Db2 上的共享存储库中，以及共享队列上的消息。 这些未存储在队列管理器页集上。 有关共享队列的更多信息，请参阅 第 291 页的『共享队列和队列共享组』，有关全局定义的更多信息，请参阅 专用和全局定义。

IBM MQ 页集的大小可达 64 GB。 每个页集都由页集标识 (PSID) 标识，该标识是 00 到 99 范围内的整数。每个队列管理器都必须具有自己的页集。

IBM MQ 使用页集零 (PSID=00) 来存储与队列管理器相关的对象定义和其他重要信息。 对于 IBM MQ 的正常操作，至关重要的是页集零不会变满，因此请勿将其用于存储消息。

为了提高系统的性能，您还应该通过将短命消息与长命消息放在不同的页集中来将它们分开。

您必须格式化页集， IBM MQ 为此提供了 FORMAT 实用程序; 请参阅 格式化页集 (FORMAT)。 还必须向 IBM MQ 子系统定义页集。

可以将 IBM MQ for z/OS 配置为在页集变满时动态展开页集。 如果需要， IBM MQ 将继续扩展页集，直到存在 123 个逻辑扩展数据块 (如果有足够的可用磁盘存储空间) 为止。 如果以这种方式定义线性数据集，那么扩展数据块可以跨越卷，但是 IBM MQ 无法将页集扩展至超过 64 GB。

不能使用来自另一个 IBM MQ 队列管理器上的一个 IBM MQ 队列管理器的页集，也不能更改队列管理器名称。 如果要将数据从一个队列管理器传输到另一个队列管理器，那么必须从第一个队列管理器卸载所有对象和消息，然后将它们重新装入到另一个队列管理器。

在运行低于 V6 的发行版的队列管理器中，无法使用大于 4 GB 的页集。 在迁移期间，当您可能需要回退到先前发行版的代码时:

- 请勿将页集 0 更改为大于 4 GB。
- 在使用先前发行版重新启动队列管理器时，大于 4 GB 的其他页集将保持脱机状态。

有关迁移能够扩展至 4 GB 以外的现有页集的更多信息，请参阅 将页集定义为大于 4 GB。

管理员可以动态地将页集添加到正在运行的队列管理器，或者从正在运行的队列管理器中除去页集 (页集为零除外)。 仅当 DEFINE PSID 命令包含 DSN 关键字时，该命令才能在队列管理器重新启动完成后运行。

存储类

存储类 是一个 IBM MQ for z/OS 概念，允许队列管理器将队列映射到页集。 您可以使用存储类来控制哪些数据集由哪些队列使用。

存储类简介

存储类 将一个或多个队列映射到页集。 这意味着该队列的消息存储在该页集上。

存储类允许您控制存储非共享消息数据的位置，以用于管理，数据集空间和负载管理或应用程序隔离目的。如果您正在使用 IMS 网桥 (如 第 386 页的『IBM MQ 和 IMS』 中所述)，那么还可以使用存储类来定义 IMS 区域的 XCF 组和成员名。

共享队列不使用存储类来获取页集映射，因为它们上的消息未存储在页集上。

存储类的工作方式

- 使用 DEFINE STGCLASS 命令定义存储类，并指定页集标识 (PSID)。
- 定义队列时，请在 STGCLASS 属性中指定存储类。

在以下示例中，本地队列 QE5 通过存储类 ARC2 映射到页集 21。

```
DEFINE STGCLASS(ARC2) PSID(21)
DEFINE QLOCAL(QE5) STGCLASS(ARC2)
```

这意味着放置在队列 QE5 上的消息将存储在页集 21 上 (如果它们在队列上的长度足以写入 DASD)。

多个队列可以使用相同的存储类，您可以根据需要定义任意数量的存储类。 例如，您可以扩展先前示例以包含更多存储类和队列定义，如下所示:

```
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) ...
```

在 第 341 页的图 73 中，两个存储类 ARC1 和 MAXI 都与页集 05 相关联。 因此，队列 QE1， QE2 和 QE3 映射到页集 05。 同样，存储类 ARC2 将队列 QE4 和 QE5 与页集 21 相关联。

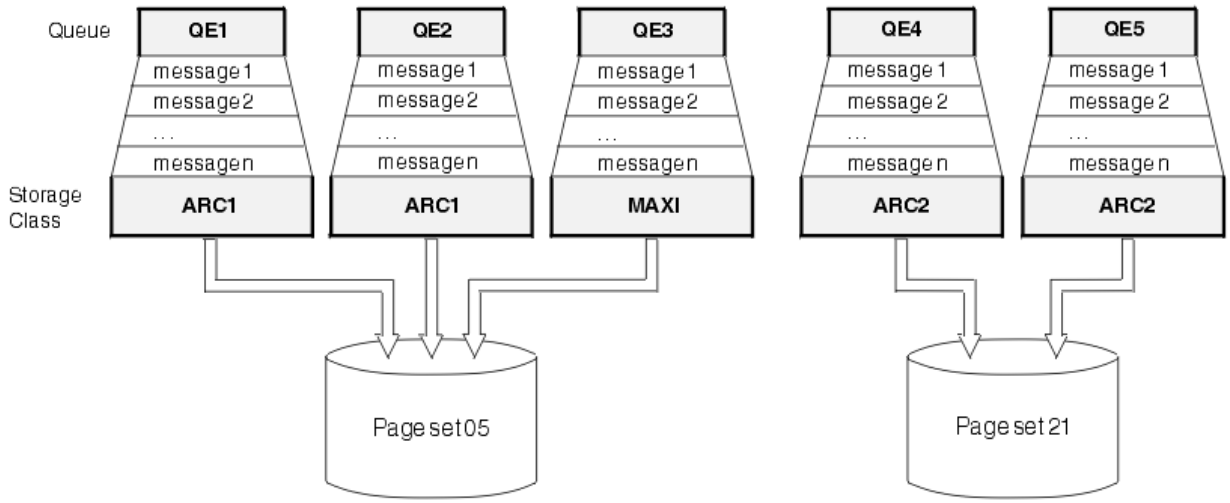


图 *73:* 通过存储类将队列映射到页集

如果在未指定存储类的情况下定义队列，那么 IBM MQ 将使用缺省存储类。

如果将消息放在指定了不存在的存储类的队列上，那么应用程序将接收到错误。 必须更改队列定义以向其提供现有存储类名，或者创建由队列命名的存储类。

仅当下列情况下，才能更改存储类:

- 使用此存储类的所有队列都为空，并且没有未落实的活动。
- 将关闭所有使用此存储类的队列。

缓冲区和缓冲池

IBM MQ for z/OS 使用缓冲区和缓冲池来临时高速缓存数据。 使用本主题来进一步了解如何组织和使用缓冲区。

为了提高效率， IBM MQ 使用了一种高速缓存形式，即消息 (和对象定义) 临时存储在缓冲区中，然后存储在 DASD 上的页集中。 短命消息 (即，在接收消息后不久从队列中检索的消息) 可能只会存储在缓冲区中。 此高速缓存活动由作为 IBM MQ 组件的缓冲区管理器控制。

这些缓冲区组织成 缓冲池。 最多可以为每个队列管理器定义 16 个缓冲池 (0 到 15)。

如果 OPMODE 设置为 OPMODE = (NEWFUNC， 800) ，那么最多可以为每个队列管理器定义 100 个缓冲池 (0 到 99)。

建议您使用与 第 342 页的图 74 中概述的对象和消息类型隔离一致的最小缓冲池数，以及应用程序可能具有的任何数据隔离需求。 每个缓冲区的长度为 4 KB。 缺省情况下，缓冲池使用 31 位存储器，在此方式下，最大缓冲区数由队列管理器地址空间中可用的 31 位存储器量确定; 对于缓冲区，请勿使用超过约 70% 的存

储空间。 或者，可以从 64 位存储器进行缓冲池存储器分配 (使用 **DEFINE BUFFPOOL** 命令的 LOCATION 属性)。 使用 LOCATION (上面) 以便使用 64 位存储器有两个好处。 首先，有更多的 64 位存储器可用，因此缓冲池可以大得多，其次， 31 位存储器可供其他功能使用。 通常，您拥有的缓冲区越多，缓冲效率越高，IBM MQ 的性能越好。

第 342 页的图 74 显示了消息，缓冲区，缓冲池和页集之间的关系。 缓冲池与一个或多个页集相关联; 每个页集与单个缓冲池相关联。

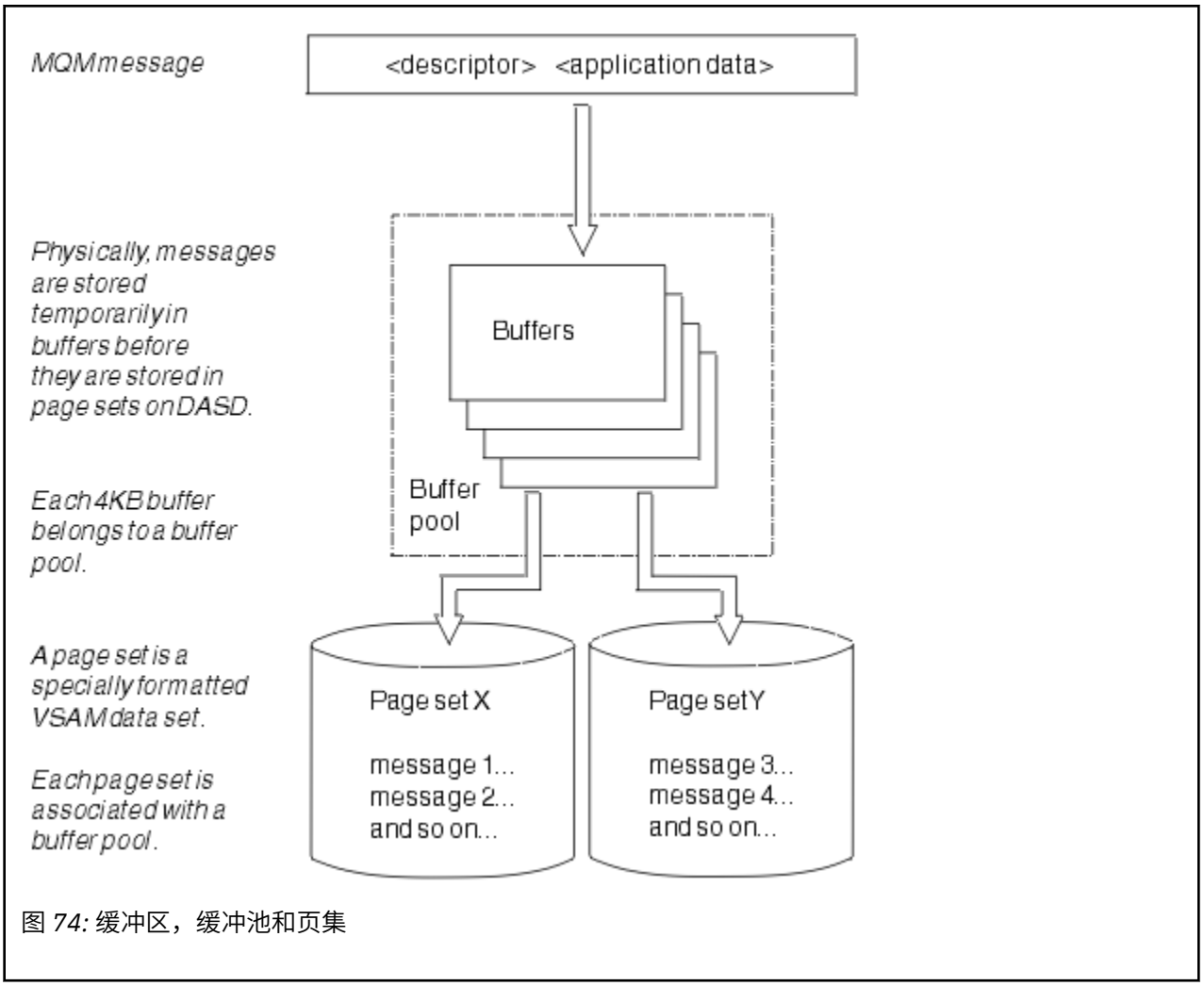


图 *74:* 缓冲区，缓冲池和页集

您可以使用 **ALTER BUFFPOOL** 命令动态发出命令以修改缓冲池大小 和位置。 可以使用 **DEFINE PSID** 命令动态添加页集，也可以使用 **DELETE PSID** 命令删除页集。

如果缓冲池太小，那么 IBM MQ 会发出消息 CSQP020E。 然后，可以动态地向受影响的缓冲池添加更多缓冲区 (请注意，您可能必须从其他缓冲池中除去缓冲区才能执行此操作)。

您可以使用 **DEFINE BUFFPOOL** 命令指定池中的缓冲区数，并且可以使用 **ALTER BUFFPOOL** 命令动态调整缓冲池的大小。 通过使用 **DISPLAY USAGE** 命令显示使用缓冲池的页集，可动态确定池中的当前缓冲区数。

出于性能原因，请勿将消息和对象定义放在同一缓冲池中。 将一个缓冲池 (例如，数字零) 专门用于保留对象定义的页集零。 同样，将短命消息和长命消息保留在不同的缓冲池中，因此保留在不同的页集和不同的队列中。

重新启动后无法使用 **DEFINE BUFFPOOL** 命令来创建新的缓冲池。 相反，如果 **DEFINE PSID** 命令使用 DSN 关键字，那么它可以显式标识当前未定义的缓冲池。 然后将创建新的缓冲池。

在何处查找有关存储管理的更多信息

使用本主题作为参考，以查找有关 IBM MQ for z/OS 的存储管理的更多信息。

您可以从以下源中找到有关此部分中主题的更多信息:

表 *25:* 在何处查找有关存储管理的更多信息	
Topic	**在哪里查找信息**
您需要多少存储空间	在 z/OS 上规划存储和性能需求
创建页面集的大小 和缓冲池	规划页集和缓冲池
管理页面集	管理页集
MQSC 命令	MQSC 命令

日志记录

IBM MQ 维护发生数据更改和重要事件时的 日志 。 如果需要，可以使用这些日志将数据恢复到先前状态。

引导数据集 (BSDS) 存储有关包含日志的数据集的信息。

日志不包含有关统计信息，跟踪或性能评估的信息。 有关 IBM MQ 收集的统计信息和监视信息的更多详细信息，请参阅 监视和统计信息。

有关日志记录的更多信息，请参阅以下主题:

- 第 343 页的『日志文件』
- 第 346 页的『日志的构造方式』
- 第 347 页的『日志的写入方式』
- 第 351 页的『引导程序数据集』

相关信息
规划日志记录环境
使用系统参数模块设置日志
管理 z/OS
IBM MQ for z/OS 的消息
MQSC 引用

日志文件

日志文件包含事务恢复所需的信息。 可以归档活动日志文件，以便您可以长时间保留日志数据。

什么是日志文件

IBM MQ 记录在 活动日志中发生的所有重要事件。 日志包含恢复所需的信息:

- 持久消息
- IBM MQ 对象，例如队列
- IBM MQ 队列管理器

活动日志包含循环使用的数据集 (最多 310 个) 的集合。

您可以启用日志归档，以便在活动日志填满时在归档数据集中生成副本。 使用归档允许您在较长时间内保留日志数据。 如果不使用归档，那么将覆盖日志合并和较早的数据。 要恢复页集或恢复 CF 结构中的数据，您需要从生成页集或结构的备份时获取日志数据。 可以在磁盘或磁带上创建归档日志。

归档

由于活动日志具有固定大小，因此 IBM MQ 会定期将每个日志数据集的内容复制到 归档日志，该日志通常是直接访问存储设备 (DASD) 或磁带上的数据集。 如果存在子系统或事务故障，那么 IBM MQ 将使用活动日志并在必要时使用归档日志进行恢复。

归档日志最多可包含 1000 个连续数据集。 您可以使用 z/OS 集成目录工具 (ICF) 对每个数据集进行编目。

归档是 IBM MQ 恢复的基本组件。 如果恢复单元是长时间运行的恢复单元，那么可能会在归档日志中找到该恢复单元中的日志记录。 在这种情况下，恢复需要来自归档日志的数据。 但是，如果关闭归档，那么具有新日志记录的活动日志会合并，覆盖先前的日志记录。 这意味着 IBM MQ 可能无法回退恢复单元，并且消息可能丢失。 然后，队列管理器异常终止。

因此，在生产环境中，**从不关闭归档**。 如果这样做，那么在系统或事务发生故障后，您将面临丢失数据的风险。 只有在测试环境中运行时，才能考虑关闭归档。 如果需要执行此操作，请使用 CSQ6LOGP 宏，如 使用 CSQ6LOGP 中所述。

为了帮助防止意外长时间运行的工作单元出现问题， IBM MQ 会发出一条消息 (CSQJ160I 或 CSQJ161I) 如果在活动日志卸载处理期间检测到长时间运行的工作单元。

双重日志记录

在双日志记录中，每个日志记录将写入两个不同的活动日志数据集，以最大程度地降低在重新启动期间发生数据丢失问题的可能性。

您可以将 IBM MQ 配置为与 单日志记录 或 双日志记录一起运行。 通过单一日志记录，日志记录将写入一次活动日志数据集。 每个活动日志数据集都是一个单扩展数据块 VSAM 线性数据集 (LDS)。 通过双重日志记录，每个日志记录将写入两个不同的活动日志数据集。 双日志记录可最大限度降低重新启动期间发生数据丢失问题的可能性。

日志分流

日志分流会导致某些工作单元的日志记录进一步向下写入日志。 这将减少在队列管理器重新启动或回退时，对于长时间运行或长期不确定的工作单元，必须读取的日志数据量。

当工作单元被视为长时，每个日志记录的表示将进一步写到日志下。 此方法称为 分流。 处理整个工作单元后，该工作单元处于 *shunted* 状态。 与已中断的工作单元相关的任何回退或重新启动活动都可以使用已中断的日志记录，而不是使用原始工作单元日志记录。

检测长时间运行的工作单元是检查点进程的功能。 在检查点时间，检查每个活动工作单元以确定是否需要对其进行 shunted。 如果工作单元自创建以来已通过两个先前的检查点，或者自上次被回避以来，那么该工作单元适合被回避。 这意味着单个工作单元可能被多次回避。 这称为 多舍入 工作单元。

每三个检查点都有一个工作单元。 但是，检查点以异步方式执行到日志开关 (或写入导致超过 LOGLOAD 的日志记录)。

一次只发生一个检查点，因此在检查点完成之前可能有多个日志开关。

这意味着，如果没有足够的活动日志，或者这些日志太小，那么在填充所有日志之前，可能无法完成大型工作单元的分流。

如果无法完成分流，那么会产生消息 CSQR027I 。

如果已关闭日志归档，那么当尝试回退其分流失败的工作单元时，将发生 ABEND 5C6 ，原因为 00D1032A 。 要避免此问题，应使用 OFFLOAD=YES。

日志分流始终处于活动状态，无论是否启用日志归档都将运行。

注: 虽然对工作单元的所有日志记录都进行了舍入，但不会对每条记录的整个内容进行舍入，而只会对回退所需的部分进行舍入。 这意味着写入的日志数据量将保持在最小范围内，如果发生页集故障，那么无法使用已中断的记录。 长时间运行的工作单元是已运行超过三个队列管理器检查点的工作单元。

有关日志分流的更多信息，请参阅 管理日志。

日志压缩

您可以配置 IBM MQ for z/OS 以在从日志数据集写入和读取日志记录时对其进行压缩和解压缩。

日志压缩可用于减少针对专用队列上的持久消息写入日志的数据量。 实现的压缩量取决于消息中包含的数据类型。 例如， "运行长度编码" (RLE) 通过压缩字节的重复实例来工作，这可以为结构化数据或面向记录的数据提供有效的结果。

注意: 要放入共享队列的持久消息不受日志压缩限制。

您可以使用系统管理设施 115 (SMF) 记录的 "日志管理器" 部分中的字段来监视实现的数据压缩量。 有关 SMF 的更多信息，请参阅 使用系统管理设施 和 记帐和统计信息消息。

日志压缩会提高系统的处理器利用率。 仅当队列管理器的吞吐量受写入日志数据集的 IO 带宽限制时，或者受保存日志数据集所需的磁盘存储器限制时，才应考虑使用压缩。 如果您正在使用共享队列，那么可以通过向队列共享组添加其他队列管理器并在更多队列管理器之间分配工作负载来缓解 IO 带宽约束。

可以根据需要启用和禁用日志压缩选项，而无需停止和重新启动队列管理器。 无论当前日志压缩设置如何，队列管理器都可以读取任何压缩日志记录。

队列管理器支持日志压缩的 3 设置。

无
不使用日志数据压缩。 这是缺省值。

RLE
使用运行长度编码 (RLE) 执行日志数据压缩。

ANY
启用队列管理器以选择用于提供最大程度的日志记录压缩的压缩算法。 此选项将导致 RLE 压缩。

您可以使用下列其中一项来控制日志记录的压缩:

- MQSC 中的 SET 和 DISPLAY LOG 命令; 请参阅 SET LOG 和 DISPLAY LOG
- PCF 接口中的 "设置日志" 和 "查询日志" 功能; 请参阅 设置日志 和 查询日志
- 系统参数模块中的 CSQ6LOGP 宏; 请参阅 使用 CSQ6LOGP

此外， "日志打印" 实用程序 CSQ1LOGP 支持扩展任何压缩日志记录。

日志数据

日志最多可包含 1800 万 (1.8* 10^{19}) 字节。 每个字节可由其从日志开头的偏移量寻址，该偏移量称为其 相对字节地址 (RBA)。

RBA 由 6 字节或 8 字节字段引用，该字段提供 2^{48} 字节或 2^{64} 字节的总可寻址范围，具体取决于是否正在使用 6 字节或 8 字节的日志 RBA。

但是，当 IBM MQ 检测到所使用的范围超出 F00000000000 (如果正在使用 6 字节的 RBA) 或 FFFF800000000000 (如果正在使用 8 字节的日志 RBA) 时，会发出消息 CSQI045， CSQI046， CSQI047 和 CSQJ032 ，警告您重置日志 RBA。

如果 RBA 值达到 FFF800000000 (如果正在使用 6 字节日志 RBA) 或 FFFFFFC000000000 (如果正在使用 8 字节日志 RBA) ，那么队列管理器将终止，原因码为 00D10257。

发出有关已用日志范围的警告消息后，应规划队列管理器中断，在此期间可以将队列管理器转换为使用 8 字节的日志 RBA ，或者可以重置日志。 重置队列管理器的日志中记录了重置日志的过程。

如果您的队列管理器正在使用 6 字节的日志 RBA ，请考虑将队列管理器转换为使用 8 字节的日志 RBA ，而不是按照 实现较大的日志相对字节地址中记录的过程来重置队列管理器的日志。

日志由 日志记录组成，每个日志记录都是作为单个单元处理的一组日志数据。 日志记录由其头的第一个字节的 RBA 或其日志记录序号 (LRSN) 标识。 RBA 或 LRSN 唯一地标识从日志中的特定点开始的记录。

是否使用 RBA 或 LRSN 来标识日志点取决于是否使用队列共享组。 在队列共享环境中，不能使用相对字节地址来唯一标识日志点，因为多个队列管理器可以同时更新同一队列，并且每个队列都有自己的日志。 要解决此问题，日志记录序号派生自时间戳记值，并且不一定表示日志记录在日志中的物理位移。

每个日志记录都有一个标头，该标头提供其类型，创建记录的 IBM MQ 子组件以及恢复单元记录的恢复单元标识。

有四种类型的日志记录，在以下标题下描述:

- 恢复单元日志记录
- 检查点记录
- 页集控制记录
- CF 结构备份记录

恢复单元日志记录

大多数日志记录描述了对 IBM MQ 队列的更改。 所有此类更改都在恢复单元内进行。

IBM MQ 使用涉及 撤销/重做 和 补偿日志记录 的特殊日志记录方法来减少重新启动时间并提高系统可用性。

这样做的一个效果是，重启时间是有界的。 如果在重新启动期间发生故障，因此必须再次重新启动队列管理器，那么在第一次重新启动期间完成到故障点的所有恢复活动都不需要在第二次重新启动期间重新应用。 这意味着连续重新启动不会花费越来越长的时间来完成。

检查点记录

为缩短重新启动时间， IBM MQ 会在正常操作期间采用定期检查点。 这些情况如下所示:

- 写入预定义数量的日志记录时。 此数字由系统参数宏 CSQ6SYSP 的称为 LOGLOAD 的检查点频率操作数定义，如 使用 CSQ6SYSP 中所述。
- 在成功重新启动结束时。
- 正常终止。
- 每当 IBM MQ 切换到循环中的下一个活动日志数据集时。

在执行检查点时， IBM MQ 发出 DISPLAY CONN 命令 (如 DISPLAY CONN 中所述) 以便将当前处于不确定状态的连接的列表写入 z/OS 控制台日志。

页集控制记录

这些记录在每个检查点注册 IBM MQ 队列管理器已知的页集和缓冲池，并记录有关在检查点时执行页集的介质恢复所需的日志范围的信息。

对页集和缓冲池的某些动态更改也会作为页集控制记录写入，以便可以恢复这些更改并在下次队列管理器重新启动时自动恢复这些更改。

CF 结构备份记录

这些记录保存从耦合设施列表结构读取的数据，以响应 BACKUP CFSTRUCT 命令。 在不太可能发生耦合设施结构故障的情况下， RECOVER CFSTRUCT 命令使用这些记录以及恢复单元记录来执行耦合设施结构到故障点的介质恢复。

相关信息

实现较大的日志相对字节地址

日志的构造方式

使用本主题来了解用于描述日志记录的术语。

每个活动日志数据集都必须是 VSAM 线性数据集 (LDS)。 写入活动日志数据集的物理输出单元是 4 KB 控制区间 (CI)。 每个 CI 都包含一个 VSAM 记录。

物理和逻辑日志记录

一个 VSAM CI 是 物理 记录。 在特定时间记录的信息构成 逻辑 记录，其长度因 CI 中的可用空间而异。 因此，一条物理记录可能包含:

- 多条逻辑记录
- 一个或多个逻辑记录以及另一个逻辑记录的一部分
- 仅一个逻辑记录的一部分

术语 日志记录 指的是 逻辑 记录，而不考虑需要多少 物理 记录来存储该记录。

日志的写入方式

使用本主题来了解 IBM MQ 如何处理日志文件记录。

IBM MQ 将每个日志记录写入称为 活动日志的 DASD 数据集。 当活动日志已满时， IBM MQ 会将其内容复制到称为 归档日志的 DASD 或磁带数据集。 此过程称为 卸载。

第 348 页的图 75 说明了日志记录过程。 日志记录通常经历以下循环:

1. IBM MQ 注意到对恢复日志记录中的数据和重要事件的更改。
2. 如果需要， IBM MQ 将处理恢复日志记录并将其分为多个段。
3. 日志记录按顺序放置在 输出日志缓冲区中，这些缓冲区格式化为 VSAM 控制区间 (CI)。 每个日志记录都由零到 $2^{64}-1$ 范围内的相对字节地址标识。
4. CI 将写入一组预定义的 DASD 活动日志数据集，这些数据集按顺序使用并循环使用。
5. 如果归档处于活动状态，那么随着每个活动日志数据集变满，其内容将自动卸载到新的归档日志数据集。

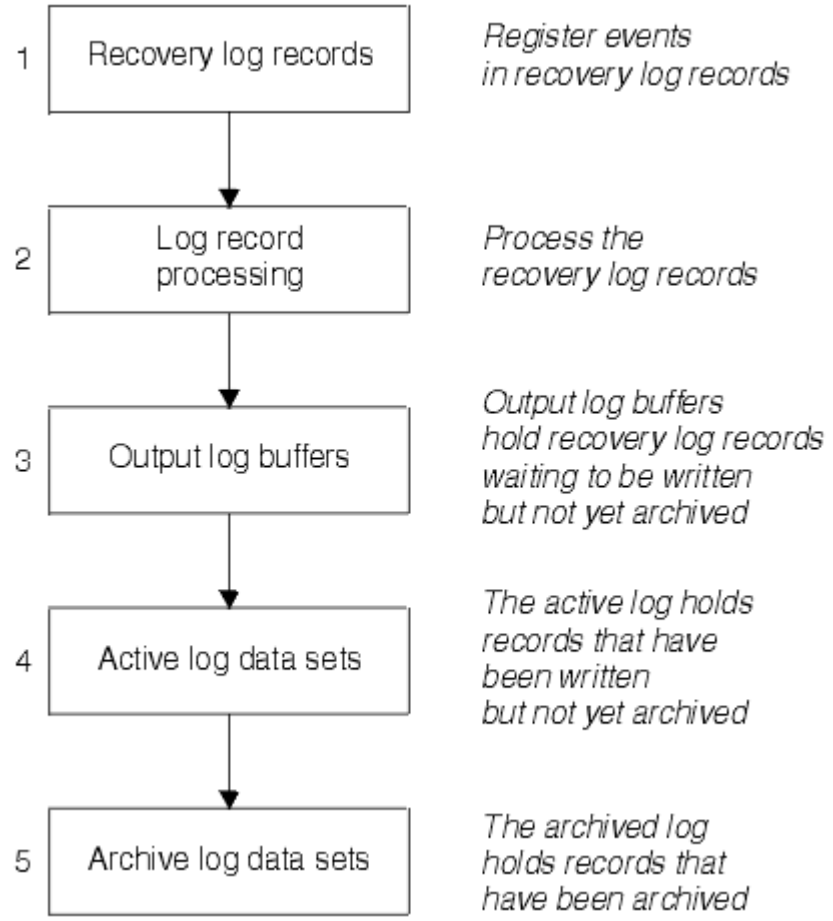


图 *75:* 日志记录过程

写入活动日志时

每当发生以下任何情况时，都会将存储中日志缓冲区写入活动日志数据集:

- 日志缓冲区变满。
- 达到写入阈值 (如 CSQ6LOGP 宏中所指定)。
- 发生某些重要事件，例如落实点，或者发出 IBM MQ BACKUP CFSTRUCT 命令时。

初始化队列管理器时，将动态分配 BSDS 中指定的活动日志数据集，以供队列管理器独占使用，并保持专门分配给 IBM MQ ，直到队列管理器终止为止。

动态添加日志数据集

可以在队列管理器运行时动态定义新的活动日志数据集。 此功能可减轻由于瞬态问题而导致归档无法卸载活动日志时队列管理器挂起的问题。 请参阅 DEFINE LOG 命令以获取更多信息。

注: 要重新定义或除去活动日志，必须终止并重新启动队列管理器。

IBM MQ 和存储管理子系统

IBM MQ 参数使您能够在动态分配 IBM MQ 归档日志数据集时指定存储管理子系统 (MVS/DFP SMS) 存储类。 IBM MQ 启动日志数据集的归档，但您可以使用 SMS 来执行归档数据集的分配。

相关参考

第 349 页的『写入归档日志时』

使用本主题来了解将活动日志复制到归档日志的过程以及该过程发生的时间。

写入归档日志时

使用本主题来了解将活动日志复制到归档日志的过程以及该过程发生的时间。

将活动日志复制到归档日志的过程称为 卸载。 在 第 349 页的图 76 中以示意图方式显示了卸载到其他日志记录事件的关系。

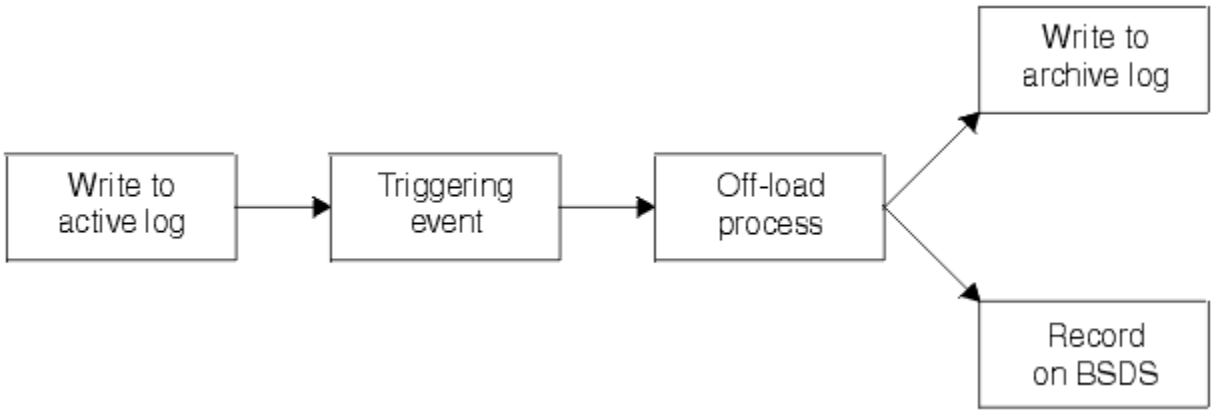


图 *76:* 卸载过程

触发卸载过程

活动日志到归档日志的卸载过程可由多个事件触发。 例如:

- 填充活动日志数据集。
- 使用 MQSC ARCHIVE LOG 命令。
- 写入活动日志数据集时发生错误。

数据集在故障点之前被截断，未写入的记录将成为新数据集的第一条记录。 对于截断的数据集，将触发卸载，如同对于正常的完整日志数据集一样。 如果存在双活动日志，那么将截断这两个副本，以使这两个副本保持同步。

消息 CSQJ110E 在最后一个可用活动日志为 5% 已满时发出，此后将以 5% 为增量发出，并说明正在使用的日志容量的百分比。 如果所有活动日志已满，那么 IBM MQ 将停止处理，直到发生卸载，并发出以下消息:

```
CSQJ111A +CSQ1 OUT OF SPACE IN ACTIVE LOG DATA SETS
```

卸载过程

当所有活动日志已满时， IBM MQ 将运行卸载过程并停止处理，直到卸载过程完成为止。 如果在活动日志已满时卸载处理失败，那么 IBM MQ 将异常终止。

当活动日志准备好卸载时，将向 z/OS 控制台操作员发送请求以安装磁带或准备 DASD 单元。 ARCWTOR 日志记录选项的值 (有关更多信息，请参阅 使用 CSQ6ARVP) 确定是否接收到请求。 如果使用磁带进行卸载，请指定 ARCWTOR=YES。 如果值为 YES ，那么请求前面有 WTOR (消息号 CSQJ008E) 指示操作员准备要分配的归档日志数据集。

操作员无需立即响应此消息。 但是，延迟响应会延迟卸载过程。 它不会影响 IBM MQ 性能，除非操作员将响应延迟很长时间以使 IBM MQ 耗尽活动日志。

操作员可以通过取消卸载过程进行响应。 在这种情况下，如果分配用于双归档数据集的第一个副本，那么卸载过程仅会延迟，直到下一个活动日志数据集变满为止。 如果分配给第二个副本，那么归档进程将切换到单副本方式，但仅针对此数据集。

卸载时的中断和错误

在卸载处理完成之前，停止队列管理器的请求不会生效。 如果 IBM MQ 在卸载过程中失败，那么卸载将在重新启动队列管理器时再次开始。

卸载处理期间的消息

卸载消息由 IBM MQ 和卸载过程发送到 z/OS 控制台。 您可以使用这些消息在各种日志数据集中查找 RBA 范围。

z/OS 较大的日志相对字节地址

此功能通过增加必须重置日志之前的时间段来提高队列管理器的可用性。

恢复数据将写入日志，以便在重新启动队列管理器时持久消息可用。 术语 "日志相对字节地址" (log RBA) 用于引用数据位置作为日志开头的偏移量。

在 IBM MQ 8.0 之前， 6 字节日志 RBA 最多可以处理 256 太字节的数据。 在写入此数量的日志记录之前，必须遵循 重置队列管理器的日志中记录的过程来重置队列管理器的日志。

重置队列管理器的日志不是一个快速过程，可能需要长时间中断，因为需要在此过程中重置页集。 对于高使用率队列管理器，此操作通常可能每年执行一次。

在 IBM MQ 8.0 中，日志 RBA 可以是 8 字节长的，现在队列管理器可以在需要重置日志 RBA 之前处理 64,000 倍以上的数据 (16 EB)。 使用更大的日志 RBA 的影响是写入的日志数据大小增加了几个字节。

何时启用此功能?

您可以随时启用此功能，但理想情况下，您应该规划更改并一次显式启用此功能以适合您的企业。 请参阅 规划以增加最大可寻址日志范围 ，以获取有关规划迁移到 8 字节日志 RBA 的指导信息。

要点: 在队列共享组中使用此功能之前，该组中的所有队列管理器都需要运行 IBM MQ 8.0 并启用 版本 8.0 新功能。

如果当前日志 RBA 接近日志 RBA 范围的末尾，请考虑将队列管理器转换为使用 8 字节日志 RBA ，而不是重置队列管理器的日志。 转换队列管理器以使用 8 字节日志 RBA 需要比重置日志更短的中断时间，并显着增加必须重置日志之前的时间段。

在队列管理器初始化期间发出的消息 CSQJ034I 指示队列管理器的日志 RBA 范围结束 (已配置) ，并且可用于确定是否正在使用 6 字节或 8 字节日志 RBA。

如何启用此功能?

通过启动版本为 2 格式的 BSDS 的队列管理器来启用 8 字节日志 RBA。 总之，实现这一目标的途径是:

1. 启用 V 8 新功能
2. 完全关闭队列管理器
3. 运行 BSDS 转换实用程序 以创建版本为 2 的 BSDS 副本。
4. 使用转换后的 BSDS 重新启动队列管理器。

一旦将队列管理器转换为使用 8 字节日志 RBA ，它就无法返回到使用 6 字节日志 RBA。

有关如何启用 8 字节日志 RBA 的详细过程，请参阅 实现较大的日志相对字节地址 。

相关信息

BSDS 转换实用程序 (CSQJUCNV)
计划增大最大可寻址日志范围

z/OS 引导程序数据集

IBM MQ 需要引导数据集作为引用日志数据集和日志记录的机制。 在正常处理和重新启动恢复期间需要此信息。

引导程序数据集用于的内容

引导数据集 (BSDS) 是 VSAM 键序列数据集 (KSDS)，用于保存 IBM MQ 所需的信息。 它包含以下内容:

- IBM MQ 已知的所有活动日志数据集和归档日志数据集的清单。 IBM MQ 使用此库存来执行以下操作:
  - 跟踪活动日志数据集和归档日志数据集
  - 找到日志记录，以便它可以满足正常处理期间的日志读取请求
  - 找到日志记录，以便它可以处理重新启动处理

  每次定义归档日志数据集或复用活动日志数据集时， IBM MQ 都会将信息存储在库存中。 对于活动日志，清单会显示哪些已满，哪些可供复用。 库存保存该数据集中保存的日志的每个部分的相对字节地址 (RBA)。
- 所有最近 IBM MQ 活动的 回绕 清单。 如果必须重新启动队列管理器，那么需要执行此操作。

如果队列管理器有错误并且您必须将其重新启动，那么 BSDS 是必需的。 IBM MQ **必须** 具有 BSDS。 为了最大限度降低重新启动期间发生问题的可能性，您可以使用双 BSD 配置 IBM MQ，每个 BSD 都记录相同的信息。 使用双 BSD 称为以 双方式运行。 如果可能，请将副本放在不同的卷上。 这将降低在卷损坏或损坏时两者都丢失的风险。 使用双 BSD，而不是对 DASD 进行双写。

BSDS 是在定制 IBM MQ 时设置的，您可以使用更改日志库存实用程序 (CSQJU003) 来管理库存。 有关此实用程序的更多信息，请参阅管理 IBM MQ for z/OS。 它由队列管理器启动过程中的 DD 语句引用。

通常， IBM MQ 会保留 BSDS 的重复副本。 如果发生 I/O 错误，那么它将取消分配失败的副本并继续使用单个 BSDS。 您可以复原双模操作，如 管理 IBM MQ for z/OS 中所述。

安装 IBM MQ 时，将首先在 BSDS 中注册活动日志。 如果不终止并重新启动队列管理器，那么无法替换活动日志。

将动态分配归档日志数据集。 分配一个数据集时，将在 BSDS 中注册数据集名称。 当添加归档时，归档日志数据集的列表将展开，并在达到用户确定的条目数时回绕。 对于单个归档日志记录，最大条目数为 1000，对于双日志记录，最大条目数为 2000。

您可以使用磁带管理系统来删除归档日志数据集 (IBM MQ 没有自动化方法)。 因此，在系统管理员删除归档日志数据集之后，有关归档日志数据集的信息可以在 BSDS 中。

相反，可能已超过最大归档日志数据集数，并且 BSDS 中的数据在数据集到达到期日期之前很久就已删除。

您可以使用以下 MQSC 命令来确定日志的范围，以及保存各种类型的介质或队列管理器恢复所需的最早日志 RBA 的活动或归档日志数据集的名称:

```
DISPLAY USAGE TYPE(DATASET)
```

如果系统参数模块指定在分配时对归档日志数据集进行编目，那么 BSDS 将指向集成目录设施 (ICF) 目录以获取后续分配所需的信息。 否则，每个卷的 BSDS 条目将注册以后分配所需的卷序列号和单元信息。

BSDS 版本

BSDS 的格式因其版本而异。 增加 BSDS 的版本允许使用新功能。 IBM MQ 支持以下 BSDS 版本:

V1
受 IBM MQ 的所有发行版支持。 版本 1 BSDS 支持 6 字节日志 RBA 值。

V2
仅受 IBM MQ 8.0 和更高版本支持。 版本 2 BSDS 支持 8 字节日志 RBA 值，每个活动日志副本中最多 310 个数据集。

V3
仅受 IBM MQ 8.0 和更高版本支持。 当将超过 31 个数据集添加到任一活动日志副本时， BSDS 将自动转换为 V 3。

您可以通过运行打印日志映射实用程序 (CSQJU004)来确定 BSDS 的版本。 要将 BSDS 从 V 1 转换为 V 2 ，请运行 BSDS 转换实用程序 (CSQJUCNV)。

请参阅 第 350 页的『较大的日志相对字节地址』 ，以获取有关 6 字节和 8 字节日志 RBA 的更多信息。

归档日志数据集和 BSDS 副本

每次创建新的归档日志数据集时，还会创建 BSDS 的副本。 如果归档日志在磁带上，那么 BSDS 是第一个输出卷上的第一个数据集。 如果归档日志位于 DASD 上，那么 BSDS 是单独的数据集。

归档日志和 BSDS 副本的数据集名称相同，但归档日志名称的最低级别限定符以 A 开头， BSDS 副本以 B 开头，例如:

归档日志名称
CSQ.ARCHLOG1.E00186.T2336229. *A* 0000001

BSDS 副本名称
CSQ.ARCHLOG1.E00186.T2336229. *B* 0000001

如果在复制 BSDS 时发生读错误，那么不会创建副本，将发出消息 CSQJ125E ，并且将在没有 BSDS 副本的情况下继续卸载到新的归档日志数据集。

定义系统

IBM MQ for z/OS 使用许多缺省对象定义，并提供样本 JCL 来创建这些缺省对象。 使用本主题来了解这些缺省对象以及样本 JCL。

设置系统参数

在 IBM MQ for z/OS 中，系统参数模块控制 IBM MQ 在其操作中使用的日志记录，归档，跟踪和连接环境。 系统参数由三个汇编程序宏指定，如下所示:

CSQ6SYSP
系统参数，包括设置连接和跟踪环境。

CSQ6LOGP
日志记录参数。

CSQ6ARVP
日志归档参数。

IBM MQ for z/OS 随附了缺省参数模块。 如果这些值不包含要使用的值，那么您可以使用随 IBM MQ 提供的样本来创建自己的参数模块。 样本为 `thlqual.SCSQPROC(CSQ4ZPRM)`。

您可以在队列管理器运行时更改某些系统参数。 请参阅 MQSC 命令中的 SET SYSTEM ， SET LOG 和 SET ARCHIVE 命令。

有关定义的更多信息，请参阅以下主题:

- 第 353 页的『定义系统对象』
- 第 356 页的『调整队列管理器』
- 第 357 页的『随 IBM MQ 提供的样本定义』

相关信息
定制样本初始化输入数据集
管理 z/OS
MQSC 引用
配置集群
监视 IBM MQ

定义系统对象

IBM MQ for z/OS 需要其他预定义对象以用于发布/预订应用程序，集群和通道控制以及其他系统管理功能。

IBM MQ for z/OS 所需的系统对象可分为以下类别:

- 发布/预订对象
- 系统缺省对象
- 系统命令对象
- 系统管理对象
- 通道队列
- 集群队列
- 队列共享组队列
- 存储类
- 定义系统对象死信队列
- 缺省传输队列
- 内部队列
- 第 356 页的『通道认证队列』

发布/预订对象

在将发布/预订应用程序与 IBM MQ for z/OS 配合使用之前，需要定义多个系统对象。 样本定义随 IBM MQ 一起提供，以帮助您定义这些对象。 CSQ4INSG 中描述了这些样本。

要使用发布/预订，您需要定义以下对象:

- 名为 SYSTEM.RETAINED.PUB.QUEUE，用于保存队列管理器中每个保留发布的副本。 每个完整主题名称最多可以有一个保留发布存储在此队列中。 如果应用程序将在许多不同的主题上使用保留发布，或者如果保留发布消息是大型消息，那么应仔细规划此队列的存储需求，包括将其分配给其自己的页集 (如果其存储需求较大)。 为了提高性能，您应该使用索引类型 MSGID 来定义此队列 (如提供的样本队列定义中所示)。
- 名为 SYSTEM.DURABLE.SUBSCRIBER.QUEUE，用于保存队列管理器中持久预订的持久副本。 为了提高性能，您应该使用索引类型 CORRELID 来定义此队列 (如提供的样本队列定义中所示)。
- 名为 SYSTEM.DURABLE.MODEL.QUEUE，用作受管持久预订的模型。
- 名为 SYSTEM.NDURABLE.MODEL.QUEUE，用作受管非持久预订的模型。
- 名为 SYSTEM.QPUBSUB.QUEUE.NAMELIST，其中包含由排队的发布/预订接口监视的队列名称的列表。
- 名为 SYSTEM.QPUBSUB.SUBPOINT.NAMELIST，其中包含由排队的发布/预订接口用于将主题对象与预订点匹配的主题对象的列表。
- 名为 SYSTEM.BASE.TOPIC，用作解析属性的基本主题。
- 名为 SYSTEM.BROKER.DEFAULT.STREAM，这是已排队的发布/预订接口所使用的缺省流。
- 名为 SYSTEM.BROKER.DEFAULT.SUBPOINT，这是已排队的发布/预订接口使用的缺省 RFH2 预订点。
- 名为 SYSTEM.BROKER.ADMIN.STREAM，这是已排队的发布/预订接口所使用的管理流。
- 名为 SYSTEM.DEFAULT.SUB，这是用于在 DEFINE SUB 命令上提供缺省值的缺省预订对象。

系统缺省对象

系统缺省对象用于在定义对象时提供缺省属性，并且不指定定义所基于的其他对象的名称。

缺省系统对象定义的名称以字符 "SYSTEM.DEFAULT"或"SYSTEM.DEF。" 例如，系统缺省本地队列名为 SYSTEM.DEFAULT.LOCAL.QUEUE。

这些对象定义这些 IBM MQ 对象的属性的系统缺省值:

- 本地队列
- 模型队列
- 别名队列
- 远程队列
- 进程
- 名称列表
- 通道
- 存储类
- 认证信息

共享队列是一种特殊类型的本地队列，因此当您定义共享队列时，定义基于 SYSTEM.DEFAULT.LOCAL.QUEUE。 您需要记住为耦合设施结构名称提供一个值，因为缺省定义中未指定一个值。 或者，您可以定义自己的缺省共享队列定义，以用作共享队列的基础，以便它们都继承必需的属性。 请记住，仅需要在队列共享组中的一个队列管理器上定义共享队列。

系统命令对象

系统命令对象的名称以字符 SYSTEM.COMMAND。 必须先定义这些对象，然后才能使用 IBM MQ 操作和控制面板向 IBM MQ 子系统发出命令。

有两个系统命令对象:

1. 系统命令输入队列是一个本地队列，在 IBM MQ 命令处理器处理命令之前，会将这些命令放在该队列上。 它必须称为 SYSTEM.COMMAND.INPUT，但别名为 SYSTEM.ADMIN.COMMAND.QUEUE ，以便与非 z/OS IBM MQ 兼容。
2. SYSTEM.COMMAND.REPLY.MODEL 是用于定义系统命令应答队列的模型队列。

有两个额外的对象可供 MQ Explorer 使用:

- SYSTEM.MQEXPLORER.REPLY.MODEL 队列
- SYSTEM.ADMIN.SVRCONN 通道

通常使用非持久消息发送命令，因此这两个系统命令对象都应该具有 DEFPSIST (NO) 属性，以便使用它们的应用程序 (包括提供的应用程序，例如实用程序以及操作和控制面板) 在缺省情况下获取非持久消息。 如果应用程序将持久消息用于命令，请为应答队列设置 DEFTYPE (PERMDYN) 属性，因为此类命令的应答消息是持久消息。

系统管理对象

系统管理对象的名称以字符 SYSTEM.ADMIN。

有七个系统管理对象:

- SYSTEM.ADMIN.CHANNEL.EVENT 队列
- SYSTEM.ADMIN.COMMAND.EVENT 队列
- SYSTEM.ADMIN.CONFIG.EVENT 队列
- SYSTEM.ADMIN.PERFM.EVENT 队列
- SYSTEM.ADMIN.QMGR.EVENT 队列
- SYSTEM.ADMIN.TRACE.ROUTE.QUEUE 队列
- SYSTEM.ADMIN.ACTIVITY.QUEUE 队列

通道队列

要使用分布式排队，您需要定义以下对象:

- 名为 SYSTEM.CHANNEL.SYNCQ，用于维护通道的序号和逻辑工作单元标识 (LUWID)。 要提高通道性能，您应该使用索引类型 MSGID 来定义此队列 (如提供的样本队列定义中所示)。
- 名为 SYSTEM.CHANNEL.INITQ，用于通道命令。

不能将这些队列定义为共享队列。

集群队列

要使用 IBM MQ 集群，您需要定义以下对象:

- 称为 SYSTEM.CLUSTER.COMMAND.QUEUE，用于在队列管理器之间通信存储库更改。 写入此队列的消息包含对要应用于存储库本地副本的存储库数据的更新或对存储库数据的请求。
- 名为 SYSTEM.CLUSTER.REPOSITORY.QUEUE，用于保存存储库的持久副本。
- 名为 SYSTEM.CLUSTER.TRANSMIT.QUEUE，这是集群中所有目标的传输队列。 出于性能原因，您应该使用索引类型 CORRELID 来定义此队列 (如样本队列定义中所示)。

这些队列通常包含大量消息。

不能将这些队列定义为共享队列。

队列共享组队列

要使用共享通道和组内排队，需要定义以下对象:

- 名为 SYSTEM.QSG.CHANNEL.SYNCQ，用于保存共享通道的同步信息。
- 名为 SYSTEM.QSG.TRANSMIT.QUEUE，用作组内排队的传输队列。 如果您正在队列共享组中运行，那么必须定义此队列，即使您未使用组内排队也是如此。

存储类

建议您定义以下六个存储类。 您必须定义其中四个值，因为 IBM MQ 需要这些值。 建议使用其他存储类定义，因为它们在样本队列定义中使用。

DEFAULT (必需)
此存储类用于所有性能不重要且不适合任何其他存储类的消息队列。 如果在定义队列时未指定缺省存储类，那么它也是提供的缺省存储类。

NODEFINE (必需)
如果未定义定义队列时指定的存储类，那么将使用此存储类。

REMOTE (必需)
此存储类主要用于传输队列，即具有短期性能关键消息的系统相关队列。

SYSLNGLV
此存储类用于长时间运行的性能关键型消息。

SYSTEM (必需)
此存储类用于性能关键型系统相关消息队列，例如 SYSTEM.CHANNEL.SYNQ 和 SYSTEM.CLUSTER.* 队列。

SYSVOLAT
此存储类用于短期的性能关键型消息。

您可以根据需要修改其属性并添加其他存储类定义。

定义系统对象死信队列

如果消息目标无效，那么将使用死信队列。 IBM MQ 将此类消息放在称为死信队列的本地队列上。 虽然具有死信队列不是必需的，但您应该将其视为必需的，尤其是在使用分布式排队或其中一个 IBM MQ 网桥时。

请 **不要** 将死信队列定义为共享队列。 放入一个队列管理器上的本地队列可能会放入死信队列。 如果死信队列是共享队列，那么另一个系统上的死信队列处理程序可以处理消息并将其放在同名的队列上，但因为这是在另一个队列管理器上，所以它将是错误的队列，或者具有不同的安全概要文件。 如果该队列不存在，那么将无法对其进行重新处理。

如果您决定定义死信队列，那么还必须将其名称告知队列管理器。 要执行此操作，请使用 ALTER QMGR DEADQ (*queue-name*) 命令。 有关更多信息，请参阅 更改队列管理器属性。

缺省传输队列

当没有其他合适的传输队列可用于将消息发送到另一个队列管理器时，将使用缺省传输队列。 如果您定义了缺省传输队列，那么还必须定义通道，以便为该队列提供服务。 如果不执行此操作，那么放入缺省传输队列的消息不会传输至远程队列管理器，它们将保留在该队列上。

如果您决定定义缺省传输队列，那么还必须将其名称告知队列管理器。 要执行此操作，请使用 ALTER QMGR 命令。

内部队列

- **暂挂数据队列**
  - 为内部使用而定义的队列 SYSTEM.PENDING.DATA.QUEUE，支持在 JMS 发布/预订环境中使用持久预订。
- **JMS 2.0 交付延迟登台队列**
  - 如果使用 JMS 2.0 提供的交付延迟功能，那么将使用内部登台队列 SYSTEM.DDELAY.LOCAL.QUEUE。 队列管理器使用此队列来临时存储以非零交付延迟发送的消息，直到交付延迟完成，并将消息放入其目标目的地。 在 CSQ4INSG 中提供并注释掉了此队列的样本定义。
  - 定义 SYSTEM.DDELAY.LOCAL.QUEUE 队列，您必须设置 STGCLASS， MAXMSGL 和 MAXDEPTH 属性，以获取将以传递延迟方式发送的预期消息数。 此外，在定义 SYSTEM.DDELAY.LOCAL.QUEUE 队列确保只有队列管理器可以将消息放入此队列。 应注意确保没有用户标识具有将消息放入此队列的权限。

通道认证队列

要在内部使用通道认证，请使用 SYSTEM.CHLAUTH.DATA.QUEUE 队列是必需的。 样本定义随 IBM MQ 一起提供，以帮助您定义这些对象。 此样本在 CSQ4INSA 中进行了描述，其中还定义了一些缺省规则。

调整队列管理器

您可以执行几个简单步骤来确保调整队列管理器以避免基本性能问题。

您可以通过多种方法来提高队列管理器的性能，这些性能由 ALTER QMGR 命令设置的队列管理器属性控制。 此部分包含有关如何通过设置队列管理器上允许的最大消息数或通过对队列管理器执行 "清理" 来执行此操作的信息。 IBM MQ SupportPac MP16 - WebSphere MQ，用于 z/OS 容量规划和调整 提供了有关性能和调整的更多信息。

同步点

队列管理器的其中一个角色是应用程序中的同步点控制。 应用程序会构造一个工作单元，其中包含任意数量的 MQPUT 或 MQGET 调用，这些调用通过 MQCMIT 调用终止。

随着一个 MQCMIT 作用域内 MQPUT 或 MQGET 调用的数量增加，落实的性能成本显着增加。 通常，应用程序应设计为在单个同步点中不使用 MQPUT/MQGET 大量消息。

您可以使用 MAXUMSGS 队列管理器属性以管理方式限制任何单个同步点中的消息数。 如果应用程序超过此限制，那么它会在 MQPUT,MQPUT1 或 MQGET 调用上接收到 MQRC_SYNCPOINT_LIMIT_达到的 MQRC_SYNCPOINT_LIMIT_超出此限制。 然后，应用程序应根据需要发出 MQCMIT 或 MQBACK。

MAXUMSGS 的缺省值为 10000。 如果要实施下限，那么可以降低此值，这也可以帮助防止循环应用程序。 在减少 MAXUMSGS 之前，请确保您了解现有应用程序以确保它们不会超过限制，或者可以容许 MQRC_SYNCPOINT_LIMIT_已达到返回码

到期消息数

已到期的消息将由下一个相应的 MQGET 调用废弃。 但是，如果未发生此类调用，那么不会废弃已到期的消息，对于某些队列，尤其是那些通过 MessageId， CorrelId 或 GroupId 进行消息检索并且为队列建立索引以实现性能的队列，可能会累积许多已到期的消息。 队列管理器可以定期扫描任何队列以查找到期的消息，然后删除这些消息。 您可以选择执行此扫描的频率 (如果有的话)。 有两种方法可以执行此操作:

显式请求
您可以控制扫描哪些队列以及时间。 发出 REFRESH QMGR TYPE (到期) 命令，指定要扫描的一个或多个队列。

定期扫描
您可以使用 EXPRYINT 属性在队列管理器对象中指定到期时间间隔。 队列管理器在每个队列上维护有关到期消息的信息，并知道何时值得扫描到期消息。 每次达到 EXPRYINT 时间间隔时，队列管理器都会查找值得扫描到期消息的候选队列，并仅扫描它认为值得扫描的那些队列。 它不会扫描所有队列。 这可避免在不必要的扫描上浪费任何处理器时间。

共享队列仅由队列共享组中的一个队列管理器扫描。 通常，要重新启动的第一个队列管理器或要设置 EXPRYINT 的第一个队列管理器将执行扫描。

注: 必须为队列共享组中的所有队列管理器设置相同的 EXPRYINT 值。

随 IBM MQ 提供的样本定义

使用本主题作为样本 JCL 的参考，以及 IBM MQ for z/OS 随附的代码。

IBM MQ 在 thlqual.SCSQPROC 库中提供了以下样本定义。 您可以使用它们来定义系统对象以及定制您自己的对象。 您可以将其中部分内容包含在初始化输入数据集中 (如 初始化命令 中所述)。

表 *26:* 系统对象的 *IBM MQ* 样本定义

初始化输入数据集 (initialization input data set)	**样本名称**
CSQINP1	CSQ4INP1 CSQ4INPR
CSQINP2	CSQ4INSA CSQ4INYS [1] CSQ4INSX CSQ4INSG CSQ4INSR CSQ4INSS CSQ4INSJ CSQ4INSM CSQ4INYG CSQ4INYR CSQ4INYC CSQ4INYD
CSQINPT	CSQ4INST CSQ4INYT

表 *26:* 系统对象的 *IBM MQ* 样本定义 *(*继续*)*	
初始化输入数据集 (initialization input data set)	**样本名称**
其他	CSQ4DISP CSQ4INPX CSQ4IVPQ CSQ4IVPG

注:

1. 这些样本定义的顺序很重要: 如果 INYS， INSX 和 INSG 顺序不正确，那么会发生错误。

CSQINP1 样本

当您对每个消息类使用一个页集时，请使用样本 CSQINP1 数据集 thlqual.SCSQPROC(CSQ4INP1)，或者在对主要消息类使用多个页集时使用 thlqual.SCSQPROC(CSQ4INPR)。 它包含缓冲池的定义，与缓冲池关联的页集以及 ALTER SECURITY 命令。 将样本包括在队列管理器启动式任务过程的 CSQINP1 并置中。

CSQINP2 样本

CSQ4INSG 系统对象样本

样本 CSQINP2 数据集 thlqual.SCSQPROC(CSQ4INSG) 包含以下系统对象的定义以供一般使用:

- 系统缺省对象
- 系统命令对象
- 系统管理对象
- 供系统使用的其他对象

必须定义此样本中的对象，但在子系统首次启动时只需执行一次。 在 CSQINP2 数据集中包含定义是执行此操作的最佳方法。 它们在队列管理器关闭和重新启动之间进行维护。 您不得更改对象名，但可以根据需要更改其属性。

满足以下条件时，将向 SYSTEM.DURABLE.SUBSCRIBER.QUEUE 队列 (即使发布预订未处于活动状态):

- IBM MQ 安装为 版本 7.0.0 或更高版本
- QMGR 属性 PSMODE 设置为 DISABLED
- 存在样本对象 CSQ4INST 语句 `DEFINE SUB('SYSTEM.DEFAULT.SUB')`。

要避免此情况，请删除或注释掉 `DEFINE SUB('SYSTEM.DEFAULT.SUB')` 语句。

仅当使用 JMS 2.0 交付延迟时，才需要定义 JMS 2.0 交付延迟登台队列 SYSTEM.DDELAY.LOCAL.QUEUE。 缺省情况下，会将队列定义注释掉，如果需要，可以将其取消注释。

CSQ4INSA 系统对象和认证样本

样本 CSQINP2 数据集 thlqual.SCSQPROC(CSQ4INSA) 包含通道认证系统队列定义。 此队列保存通道认证记录。 它还包含缺省通道认证规则。

如果 CHLAUTH 在队列管理器上已启用并且要运行通道，或者要 SET 或 DISPLAY CHLAUTH 记录，那么必须定义此样本中的对象。 您只需在子系统首次启动时定义一次它们。 在 CSQINP2 数据集中包含定义是执行此操作的最佳方法。 它们在队列管理器关闭并重新启动时进行维护，您不得更改队列名称。

CSQ4INSS 系统对象样本

如果您正在使用队列共享组，那么可以定义其他系统对象。

样本数据集 thlqual.SCSQPROC(CSQ4INSS) 包含用于 CF 结构的样本命令以及共享通道和组内排队所需的系统对象的一组定义。

不能按现在的方式使用此样本; 必须先对其进行定制，然后才能使用此样本。 然后，可以将此成员包括在队列管理器启动过程的 CSQINP2 DD 并置中，也可以将其用作 CSQUTIL 实用程序的 COMMAND 函数的输入以发出所需的命令。

在定义组或共享对象时，仅需要将它们包括在队列共享组中的一个队列管理器的 CSQINP2 DD 并置中。

CSQ4INSX 系统对象样本

如果您正在使用分布式排队和集群，那么必须定义其他系统对象。

样本数据集 thlqual.SCSQPROC(CSQ4INSX) 包含所需的队列定义。 可以将此成员包括在队列管理器启动过程的 CSQINP2 DD 并置中，也可以将其用作 CSQUTIL 实用程序中的 COMMAND 函数的输入，以发出必需的 DEFINE 命令。

有两种类型的对象定义:

- SYSTEM.CHANNEL.xx，任何分布式排队都需要
- SYSTEM.CLUSTER.xx，集群所需的

CSQ4INSJ 系统 JMS 对象样本

定义在 JMS 发布/预订域中使用的队列。

CSQ4INSM 系统对象样本

如果您正在使用高级消息安全性，那么必须定义其他系统对象。 样本数据集 thlqual.SCSQPROC(CSQ4INSM) 包含所需的队列定义。

CSQ4INSR 对象样本

定义 WebSphere Application Server 和代理使用的队列。

CSQ4INYD 对象样本

如果您正在使用分布式排队，并且需要设置自己的队列，进程和通道。

样本数据集 thlqual.SCSQPROC(CSQ4INYD) 包含可用于定制分布式排队对象的样本定义。 它包括:

- 发送端的一组定义
- 接收端的一组定义
- 一组用于使用客户机的定义

不能按样使用此样本-在使用前必须对其进行定制。 然后，可以将此成员包含在队列管理器启动过程的 CSQINP2 DD 并置中，也可以将其用作 CSQUTIL 实用程序的 COMMAND 函数的输入以发出必需的 DEFINE 命令。 (这更可取，因为这意味着您不必在每次重新启动队列管理器时重新定义这些对象)。

CSQ4INYC 对象样本

如果您正在使用集群，那么将在需要时自动创建等同于分布式排队的通道定义和远程队列定义的定义。 但是，需要一些手动通道定义-集群的集群接收方通道以及至少一个集群存储库队列管理器的集群发送方定义。

样本数据集: thlqual.SCSQPROC(CSQ4INYC) 包含可用于定制集群对象的以下样本定义:

- 队列管理器的定义
- 接收通道的定义
- 发送通道的定义
- 集群队列的定义
- 集群列表的定义

不能按样使用此样本-在使用前必须对其进行定制。 然后，可以将此成员包含在队列管理器启动过程的 CSQINP2 DD 并置中，也可以将其用作 CSQUTIL 实用程序的 COMMAND 函数的输入以发出必需的 DEFINE 命令。 这是首选项，因为这意味着您不必在每次重新启动 IBM MQ 时重新定义这些对象。

CSQ4INYG 对象样本

样本数据集: thlqual.SCSQPROC(CSQ4INYG) 包含可用于定制您自己的对象以供常规使用的以下样本定义:

- 死信队列
- 缺省传输队列
- CICS 适配器对象

不能按样使用此样本-在使用前必须对其进行定制。 然后，可以将此成员包含在队列管理器启动过程的 CSQINP2 DD 并置中，也可以将其用作 CSQUTIL 实用程序的 COMMAND 函数的输入以发出必需的 DEFINE 命令。 这是首选项，因为这意味着您不必在每次重新启动 IBM MQ 时重新定义这些对象。

除了此处的样本定义外，您还可以使用系统对象定义作为自己的资源定义的基础。 例如，您可以制作 SYSTEM.DEFAULT.LOCAL.QUEUE ，并将其命名为 MY.DEFAULT.LOCAL.QUEUE。 然后，您可以根据需要更改此副本中的任何参数。 然后，可以使用您选择的任何方法发出 DEFINE 命令，前提是您有权创建该类型的资源。

缺省传输队列
在确定是否要定义缺省传输队列之前，请先阅读 缺省传输队列 描述。

- 如果您决定要定义缺省传输队列，请记住还必须定义通道以提供服务。
- 如果您决定不想定义一个，请记住从样本中的 ALTER QMGR 命令中除去 DEFXMITQ 语句。

CICS 适配器对象
此样本定义名为 CICS01.INITQ。 此队列由 IBM MQ 提供的 CKTI 事务使用。 您可以更改此队列的名称; 但是，它必须与在 INITPARM 语句中的 CICS 系统初始化表 (SIT) 或 SYSIN 覆盖中指定的名称相匹配。

CSQ4INYS/CSQ4INYR 对象样本

用于使用的存储类定义:

- 为每类消息设置一个页面
- 主要消息类的多页集

例如， SYSTEM.COMMAND.INPUT 使用 STGCLASS ('SYSVOLAT') 和 SYSTEM.CLUSTER.TRANSMIT.QUEUE 使用 STGCLASS ('REMOTE')。 在 CSQ4INYS 中，这两个存储类都使用相同的页集。 在 CSQ4INYR 中，这些存储类使用不同的页集以减少传输队列填充的影响。

CSQINPT 样本

CSQ4INST
样本数据集 thlqual.SCSQPROC(CSQ4INST) 包含系统缺省预订的定义。

缺省系统预订 SYSTEM.DEFAULT.SUB 已从 IBM WebSphere MQ 7.1 中的 CSQ4INSG 移至 CSQ4INST 。

必须在此样本中定义对象，但只需在首次启动发布/预订引擎时执行一次。 在 CSQINPT 数据集中包含定义是执行此操作的最佳方法。 它在队列管理器关闭和重新启动之间进行维护。 您不得更改对象名，但可以根据需要更改其属性。

CSQ4INYT
样本数据集 thlqual.SCSQPROC(CSQ4INYT) 包含一组在启动发布/预订引擎时可能要运行的命令。 此样本显示主题和预订信息。

其他

CSQ4DISP 显示样本

样本数据集: thlqual.SCSQPROC(CSQ4DISP) 包含一组通用 DISPLAY 命令，这些命令显示队列管理器上的所有已定义资源。 这包括所有 IBM MQ 对象的定义以及诸如存储类和跟踪之类的定义。 这些命令可以生成大量输出。 可以在 CSQINP2 数据集中使用此样本，也可以将其用作 CSQUTIL 实用程序的 COMMAND 函数的输入。

CSQ4INPX 样本

样本数据集: thlqual.SCSQPROC(CSQ4INPX) 包含您每次启动通道启动程序时可能要执行的一组命令。 您必须先定制此样本，然后才能将其包含在通道启动程序的 CSQINPX 数据集中。

CSQ4IVPQ 和 CSQ4IVPG 样本

样本数据集 thlqual.SCSQPROC(CSQ4IVPQ) 和 thlqual.SCSQPROC(CSQ4IVPG) 包含运行安装验证程序 (IVP) 所需的 DEFINE 命令集。

可以在 CSQINP2 数据集中包含这些样本。 成功运行 IVP 后，每次重新启动队列管理器时都不需要再次运行这些 IVP。 因此，您不需要将这些样本永久保留在 CSQINP2 并置中。

恢复和重新启动

使用本主题中的链接可了解 IBM MQ for z/OS 用于重新启动和恢复的功能。

IBM MQ for z/OS 具有用于重新启动和恢复的强大功能。 请参阅以下链接，以获取有关队列管理器在停止后如何恢复以及重新启动时会发生什么情况的信息:

- 第 361 页的『如何对数据进行更改』
- 第 363 页的『如何保持一致性』
- 第 365 页的『终止期间发生的情况』
- 第 366 页的『重新启动和恢复期间发生的情况』
- 第 367 页的『如何解析不确定的恢复单元』
- 第 369 页的『共享队列恢复』

相关信息

规划备份和恢复
IBM MQ for z/OS 恢复操作
管理 z/OS
IBM MQ for z/OS 的消息
MQSC 引用

如何对数据进行更改

IBM MQ 必须与其他子系统交互以保持所有数据一致。 本主题包含有关 恢复单元的信息，它们是什么以及如何在 回退中使用它们。

恢复单元

恢复单元 是由应用程序的单个队列管理器完成的处理，用于将 IBM MQ 数据从一个一致性点更改为另一个一致性点。 一致性点 (也称为 同步点 或 落实点) 是应用程序访问的所有可恢复数据一致的时间点。

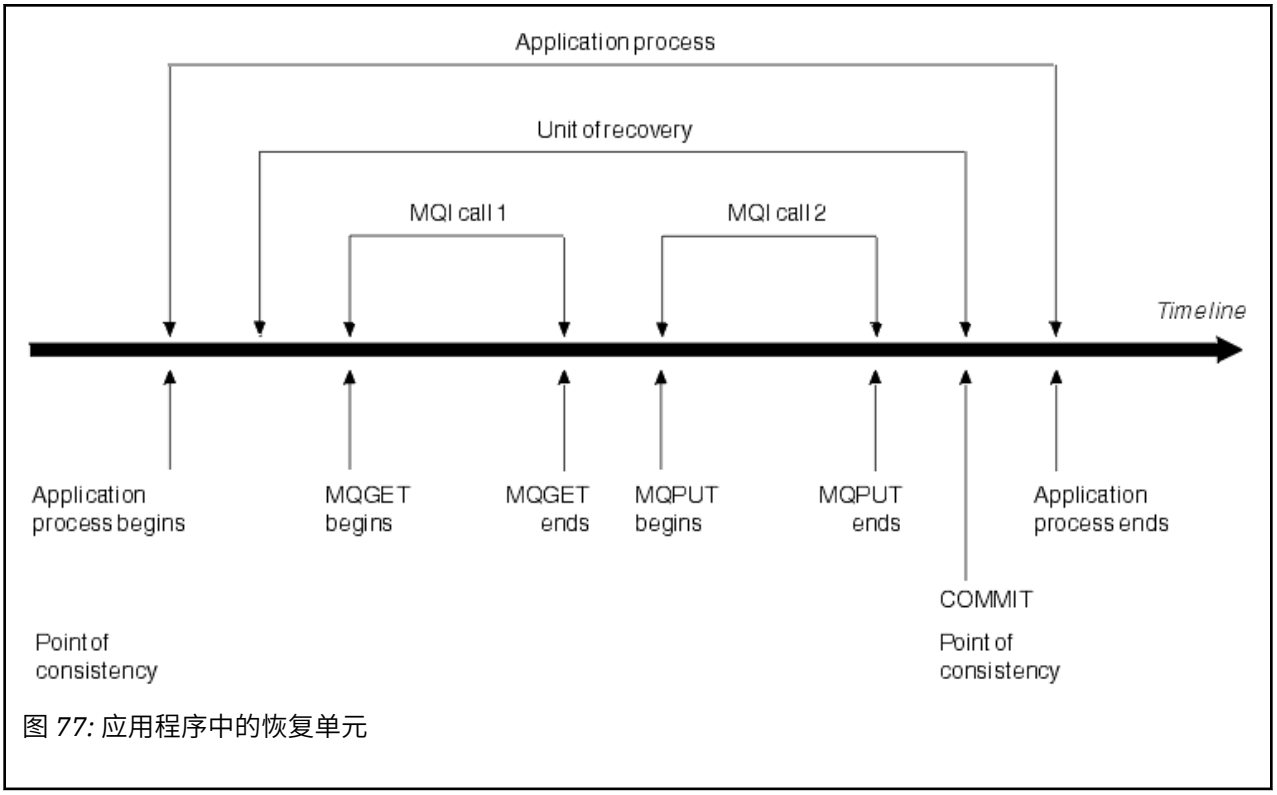


图 77: 应用程序中的恢复单元

恢复单元从程序开始后对数据的第一个更改开始，或者在前一个一致性点之后; 它以后一个一致性点结束。 第 362 页的图 77 显示了恢复单元，一致性点和应用程序之间的关系。 在此示例中，应用程序通过 MQI 调用 1 和 2 对队列进行更改。 应用程序可以包含多个恢复单元，也可以仅包含一个恢复单元。 但是，任何完整的恢复单元都将在落实点中结束。

例如，银行交易将资金从一个帐户转移到另一个帐户。 首先，程序从第一个科目 A 中减去金额。 然后，将金额添加到第二个帐户 B。 从 A 中减去金额后，这两个帐户不一致， IBM MQ 无法落实。 当将金额添加到科目 B 时，它们变为一致。 当这两个步骤都完成时，程序可以通过落实来声明一致性点，从而使更改对其他应用程序可见。

应用程序的正常终止会自动导致一致性点。 CICS 和 IMS 程序中的某些程序请求也会导致一致性点，例如 EXEC CICS SYNCPOINT。

回退工作

如果在恢复单元中发生错误，那么 IBM MQ 将除去对数据的任何更改，将数据返回到恢复单元启动时的状态; 即， IBM MQ 将回退工作。 这些事件显示在 第 363 页的图 78 中。

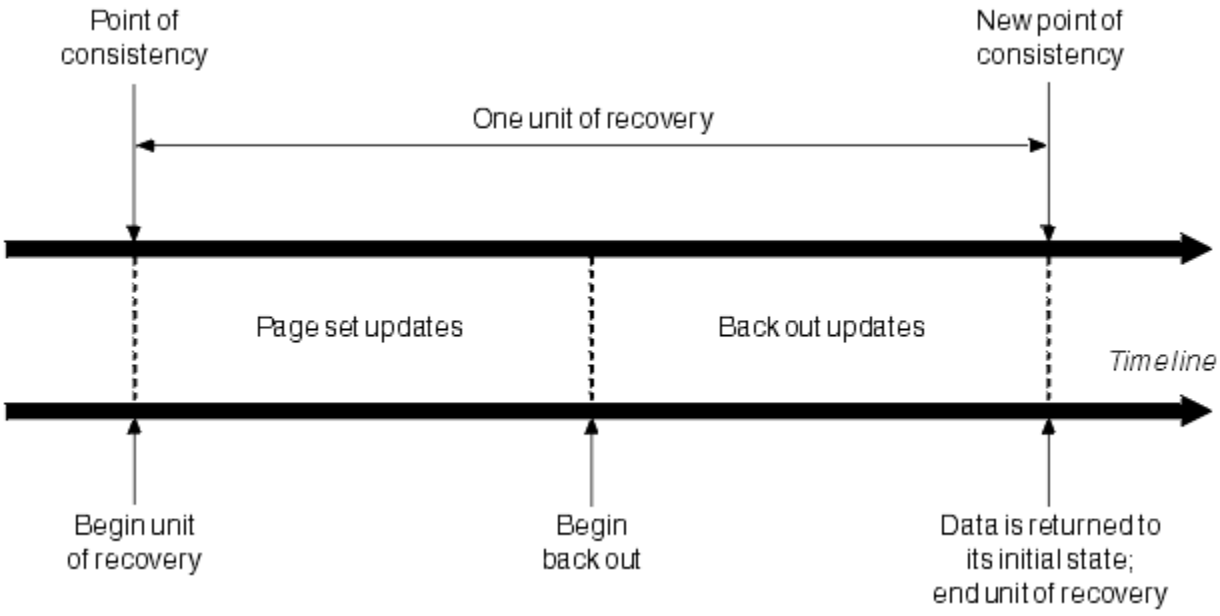


图 *78:* 显示回退的恢复单元

如何保持一致性

IBM MQ 中的数据必须与批处理， CICS， IMS 或 TSO 一致。 在一个数据中更改的任何数据都必须与另一个数据中的更改相匹配。

在一个系统落实已更改的数据之前，它必须知道另一个系统可以进行相应的更改。 因此，系统必须进行通信。

在 两阶段落实 期间 (例如，在 CICS 下) ，一个子系统协调进程。 该子系统称为 协调程序; 另一个是 参与者。 CICS 或 IMS 始终是与 IBM MQ 交互的协调程序，而 IBM MQ 始终是参与者。 在批处理或 TSO 环境中， IBM MQ 可以参与由 z/OS RRS 协调的两阶段落实协议。

在 单阶段落实 期间 (例如，在 TSO 或批处理下) ， IBM MQ 始终是交互中的协调程序，并完全控制落实过程。

在 WebSphere Application Server 环境中， JMS 会话对象的语义确定是使用单阶段还是两阶段落实协调。

与 CICS 或 IMS 的一致性

IBM MQ 与 CICS 或 IMS 之间的连接支持以下同步点协议:

- 两阶段落实-用于更新由多个资源管理器拥有的资源的事务。

  这是标准的分布式同步点协议。 它涉及比单阶段落实更多的日志记录和消息流。
- 单阶段落实-用于更新单个资源管理器 (IBM MQ) 所拥有的资源的事务。

  此协议针对日志记录和消息流进行了优化。
- 绕过同步点-对于涉及 IBM MQ 的事务，但在需要同步点的队列管理器中不执行任何操作 (例如，浏览队列)。

在每种情况下， CICS 或 IMS 都充当同步点管理器。

IBM MQ 用于与 CICS 或 IMS 通信的两阶段落实的阶段如下所示:

1. 在阶段 1 中，每个系统独立确定其日志中是否记录了足够的恢复信息，并且可以落实其工作。

   在阶段结束时，系统进行通信。 如果他们同意，每个人都开始下一阶段。

2. 在阶段 2 中，更改是永久性的。 如果其中一个系统在阶段 2 期间异常终止，那么该操作由恢复过程在重新启动期间完成。

两阶段落实过程的说明

第 364 页的图 79 说明了两阶段落实过程。 CICS 或 IMS 协调程序中的事件显示在上行， IBM MQ 中的事件显示在下行。

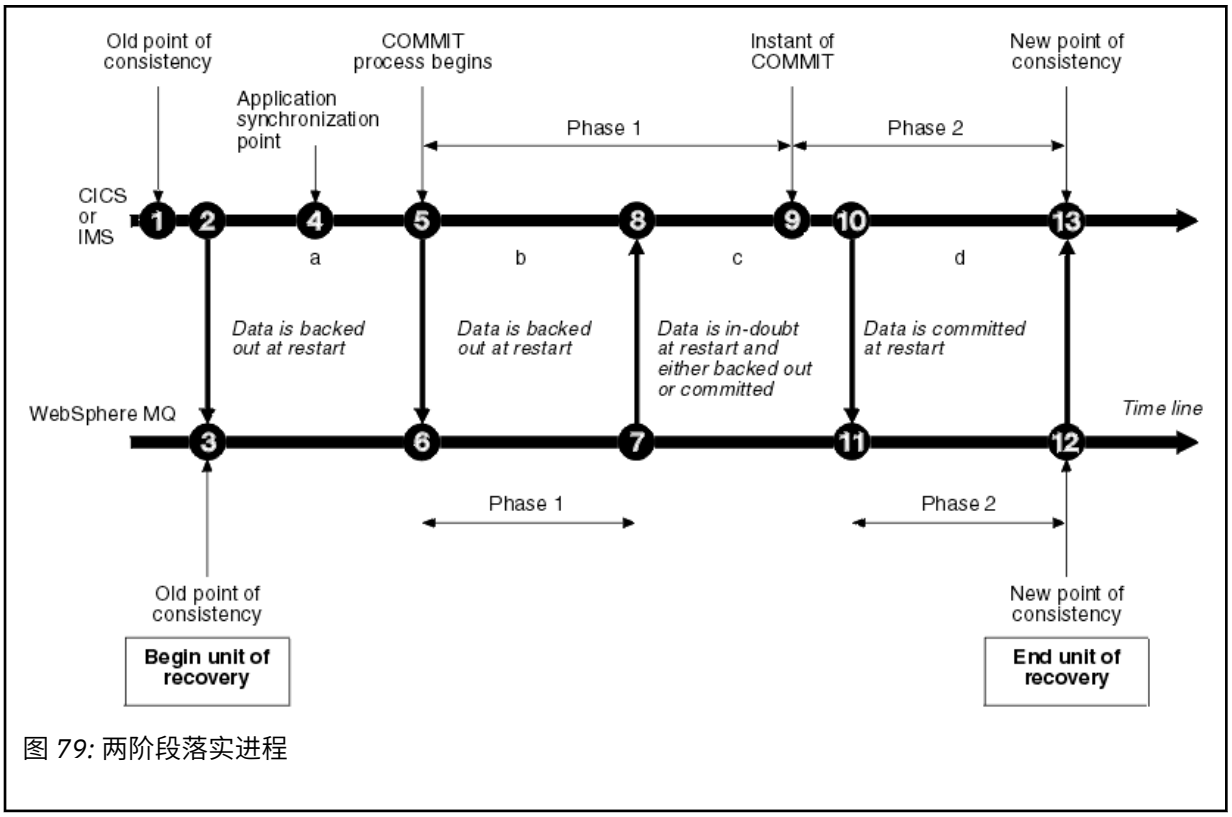


图 *79:* 两阶段落实进程

以下部分中的数字与图中显示的数字相链接。

1. 协调程序中的数据是一致的。
2. 协调程序中的应用程序通过添加消息来调用 IBM MQ 以更新队列。
3. 这将在 IBM MQ 中启动恢复单元。
4. 在协调程序中继续处理，直到到达应用程序同步点为止。
5. 然后，协调程序启动落实处理。 CICS 程序使用 SYNCPOINT 命令或正常应用程序终止来启动落实。 IMS 程序可以使用 CHKP 调用， SYNC 调用，对 IOPCB 的 GET UNIQUE 调用或正常应用程序终止来启动落实。 落实处理的阶段 1 开始。
6. 当协调程序开始阶段 1 处理时， IBM MQ 也是如此。
7. IBM MQ 成功完成阶段 1 ，将此事实写入其日志，并通知协调程序。
8. 协调程序接收通知。
9. 协调程序已成功完成其阶段 1 处理。 现在，两个子系统都同意落实数据更改，因为这两个子系统都已完成阶段 1 ，并且可以从任何错误中恢复。 协调程序在其日志中记录落实的瞬间-两个子系统进行更改的不可撤销决策。

   协调程序现在开始处理的阶段 2-实际落实。
10. 协调程序通知 IBM MQ 开始其阶段 2。
11. IBM MQ 记录阶段 2 的开始。
12. 阶段 2 已成功完成，现在这是 IBM MQ 的新一致性点。 IBM MQ 然后通知协调程序它已完成其阶段 2 处理。
13. 协调程序完成其阶段 2 处理。 由两个子系统控制的数据现在是一致的，可供其他应用程序使用。

异常终止后如何保持一致性

在异常终止后重新启动队列管理器时，它必须确定是落实还是回退在终止时处于活动状态的恢复单元。 对于某些恢复单元， IBM MQ 具有足够的信息来做出决策。 对于其他用户，它不会执行此操作，并且必须在重新建立连接时从协调程序获取信息。

第 364 页的图 79 显示两个阶段中的四个周期 :a， b， c 和 d。 恢复单元的状态取决于发生终止的时间段。 状态可以是以下项之一：

在飞行中
队列管理器在完成阶段 1 (时间段 a 或 b) 之前结束; 在重新启动期间， IBM MQ 会回退更新。

不确定
队列管理器在完成阶段 1 之后以及在开始阶段 2 之前结束 (周期 c); 只有协调程序知道是在落实之前还是之后发生了错误 (点 9)。 如果是之前发生的，那么 IBM MQ 必须回退其更改; 如果是之后发生的，那么 IBM MQ 必须进行其更改并落实这些更改。 重新启动时， IBM MQ 将在处理此恢复单元之前等待来自协调程序的信息。

在落实中
队列管理器在开始其自己的阶段 2 处理 (周期 d) 后结束; 它进行已落实的更改。

在回退中
队列管理器在恢复单元开始回退后结束，但在重新启动期间进程完成 (图中未显示) 之前， IBM MQ 继续回退更改。

终止期间发生的情况

队列管理器正常终止以响应 STOP QMGR 命令。 如果队列管理器由于任何其他原因而停止，那么终止会异常。

请注意，在队列管理器终止期间， IBM MQ 在内部发出命令

```
DISPLAY CONN(*) TYPE(CONN) ALL WHERE (APPLTYPE NE SYSTEMAL)
```

以便您了解哪些线程可能会阻止队列管理器完成关闭。

SYSTEMAL 与 SYSTEM 或 CHINIT 的 APPLTYPES 相匹配，因此 DISPLAY CONN 命令将过滤与 SYSTEMAL 不匹配的应用程序类型，返回到可能阻止正常关闭的线程的作业记录信息。

正常终止

在正常终止中， IBM MQ 以有序方式停止所有活动。 您可以使用停顿，强制或重新启动方式来停止 IBM MQ 。 效果在 第 365 页的表 27 中给出。

表 *27:* 使用 *QUIESCE* ， *FORCE* 和 *RESTART* 终止

线程类型	**QUIESCE**	**FORCE**	**重新启动**
活动线程数	运行到完成	回退	回退
新线程数	可以启动	不允许	不允许
新连接	不允许	不允许	不允许

如果在应用程序仍处于连接状态时发生终止，那么将通知批处理应用程序。

使用 CICS 时，当前线程仅运行到恢复单元的末尾。 对于 CICS，以停顿方式停止队列管理器将停止 CICS 适配器，因此如果活动任务包含多个恢复单元，那么该任务不一定运行到完成。

如果以强制或重新启动方式停止队列管理器，那么不会分配任何新线程，并且将回滚已连接的线程上的工作。 使用这些方式可以为处于落实处理阶段之间的线程创建不确定的恢复单元。 当 IBM MQ 与控制 CICS， IMS 或 RRS 子系统重新连接时，将解决这些问题。

在任何方式下停止队列管理器时，步骤如下：

1. 连接已结束。
2. IBM MQ 停止接受命令。
3. IBM MQ 确保完成对页集的任何未完成的更新。
4. IBM MQ 在内部发出 DISPLAY USAGE 命令，以便在 z/OS 控制台日志上记录重新启动 RBA。
5. 将执行关闭检查点并更新 BSDS。

指定停顿方式的终止不会影响不确定的恢复单元。 任何有疑问的单位都是有疑问的。

异常终止

异常终止可能会使数据处于不一致状态，例如:

- 恢复单元在到达一致性点之前已中断。
- 尚未将已落实的数据写入页集。
- 未落实的数据已写入页集。
- 应用程序已在落实进程的阶段 1 和阶段 2 之间中断，使恢复单元处于不确定状态。

IBM MQ 解决了在重新启动和恢复期间由于异常终止而产生的任何数据不一致。

重新启动和恢复期间发生的情况

IBM MQ 使用其恢复日志和引导数据集 (BSDS) 来确定重新启动时要恢复的内容。 BSDS 标识活动日志数据集和归档日志数据集，以及日志中最新 IBM MQ 检查点的位置。

重新启动和恢复简介

初始化 IBM MQ 后，将按如下所示执行队列管理器重新启动过程:

- 日志初始化
- 当前状态重建
- 转发日志恢复
- 反向日志恢复
- 队列索引重建

恢复完成后:

- 已落实的更改将反映在数据中。
- 不确定活动反映在数据中。 但是，在 IBM MQ 识别并处理不确定决策之前，数据已锁定并且无法使用。
- 已从队列中除去中断的正在进行的更改和正在中止的更改。 这些消息是一致的，可以使用。
- 已采用新的检查点。
- 已为包含持久消息的已建立索引的队列构建新索引 (如 第 367 页的『重建队列索引』 中所述)。

如果正在使用双 BSD ，那么 IBM MQ 会检查 BSDS 中时间戳记的一致性:

- 如果 BSDS 的两个副本都是最新的，那么 IBM MQ 会测试两个时间戳记是否相等。 如果不是， IBM MQ 将发出消息 CSQJ120E 并终止。 当 BSDS 的两个副本在单独的 DASD 卷上维护，并且在队列管理器停止时复原了其中一个卷时，可能会发生此情况。 IBM MQ 在重新启动时检测到情境。
- 如果取消分配 BSDS 的一个副本，并且继续使用单个 BSDS 进行日志记录，那么可能会出现问题。 如果两个 BSDS 副本都在单个卷上维护，并且卷已复原，或者如果两个 BSDS 副本都单独复原，那么 IBM MQ 可能不会检测到复原。 在这种情况下，系统将不知道 BSDS 中未记录的日志记录。

在应用程序请求连接 之后 发生重新启动时，不会通知批处理应用程序。

了解恢复所需的日志范围

在重新启动期间，必须读取的日志数据范围取决于许多因素:

- 在异常终止时，系统中通常有许多不完整的工作单元。 如前所述，重新启动处理将使系统处于一致性状态，这可能涉及回退正在运行的工作单元，或恢复对不确定工作单元的锁定。 工作单元恢复要求所有工作单元日志记录都可用于进行中工作单元，回退工作单元和不确定工作单元。 IBM MQ 将 "分流" 旧的工作单元，以便可以使用更小范围的日志数据来执行工作单元恢复。
- 在异常终止时，通常有许多持久更新仅保存在缓冲池高速缓存中。 它们尚未写入磁盘。 这些更改必须从日志中读取，并重新应用于页集中保存的数据。 检查点中的页集恢复 RBA 描述将页集更新为一致状态所需的最低日志 RBA。
- 如果已将旧页集引入到系统中，例如，已引入页集备份以从介质故障中恢复，那么必须从执行备份时的日志中读取所有更改。 这些更改将重新应用于正在恢复的页集中保存的数据。 页集的第 0 页中保存的页集恢复 RBA 描述了页集的介质恢复所需的最低日志 RBA。
- 如果在共享队列上使用持久消息，那么需要一系列日志数据来恢复保存持久消息的 CFSTRUCT。 执行 CFSTRUCT 恢复所需的最早日志数据来自旧 CFSTRUCT BACKUP 的时间。

在正常运行期间，可以使用 DISPLAY USAGE TYPE (DATASET) 命令来查看与这些因素相关联的恢复日志范围 (当然，由于重新引入旧页集，它无法提供信息)。 为避免在异常终止时可能延长队列管理器重新启动的任何问题，请定期监视 DISPLAY USAGE TYPE (DATASET) 输出的值。

此外，队列管理器还会发出与以下因素相关的参考消息:

- CSQJ160I 和 CSQJ161I 警告长时间运行的工作单元。
- CSQR026I 和 CSQR027I 提供有关这些长时间运行的工作单元是否已成功舍入的信息。
- CSQE040I 和 CSQE041E 警告结构备份越来越旧，因此 RECOVER CFSTRUCT 操作将需要很长时间。

确定哪个应用程序具有长时间运行的工作单元

可以确定具有长时间运行的工作单元的应用程序。 要执行此操作，请使用 DISPLAY CONN 命令。

DISPLAY CONN 命令返回连接到队列管理器的所有应用程序的连接信息以及其他信息，这些信息可帮助您确定哪些应用程序当前具有长时间运行的工作单元。 DISPLAY CONN 命令返回的信息与 DISPLAY QSTATUS 命令返回的信息类似，但主要区别在于 DISPLAY CONN 显示有关特定连接的对象和事务信息的信息，而不是与特定对象关联的连接的详细信息。

对于每个已连接的应用程序， DISPLAY CONN 命令会返回以下信息:

- 基本信息，包括连接标识和 PID。
- 该连接的事务信息，包括创建事务的时间和日期 (即，在同步点下执行第一个 MQGET/PUT 时) 以及事务首次写入日志时的时间和日期。
- 记录时间信息，指示哪个应用程序仍具有长时间运行的工作单元。
- 连接当前已打开的所有对象的列表。 每个对象的详细信息将作为单独的消息返回，并将 "连接标识" 用作键。 由于存在不同类型的对象 (例如队列和队列管理器) ，因此随对象显示的信息特定于其对象类型。

重建队列索引

要提高未按顺序检索消息的队列上的 MQGET 操作的速度，可以指定您希望 IBM MQ 维护该队列上所有消息的消息或相关标识或组标识的索引。

重新启动队列管理器时，将为每个队列重建这些索引。 这仅适用于持久消息; 非持久消息将在重新启动时删除。 如果已建立索引的队列包含大量持久消息，那么这将增加重新启动队列管理器所花费的时间。

您可以选择使用 CSQ6SYSP 宏的 QINDXBLD 参数来异步重建索引以启动队列管理器。 如果设置 QINDXBLD=NOWAIT ，那么 IBM MQ 将重新启动而不等待重建索引。

如何解析不确定的恢复单元

如果 IBM MQ 失去与另一个资源管理器的连接，那么它通常会尝试在重新启动时恢复所有不一致的对象。

如果 IBM MQ 失去了与 CICS， IMS 或 RRS 的连接，那么通常会尝试在重新启动时恢复所有不一致的对象。 解析不确定的恢复单元所需的信息必须来自协调系统。 以下各部分描述了针对不同环境的解析过程。

- 如何从 CICS 解析不确定的恢复单元
- 如何从 IMS 解析不确定的恢复单元
- 如何从 RRS 解析不确定的恢复单元
- 如何解析具有 GROUP 恢复处置单元的不确定恢复单元

如何从 CICS 解析不确定的恢复单元

在某些情况下， CICS 无法运行 IBM MQ 进程来解析不确定的恢复单元。 发生此情况时， IBM MQ 将发送下列其中一条消息:

- CSQC404E
- CSQC405E
- CSQC406E
- CSQC407E

后跟消息 CSQC408I。

有关这些消息的含义的详细信息，请参阅 IBM MQ for z/OS 消息，完成和原因码 手册。

不确定单元的解析不会影响 CICS 资源。 CICS 控制恢复协调，并在重新启动时自动落实或回退每个单元，具体取决于是否存在标记落实开始的日志记录。 在重新连接 IBM MQ 时，不确定对象的存在不会锁定 CICS 资源。

CICS 适配器的其中一项功能是在 CICS 和 IBM MQ 之间保持数据同步。 如果队列管理器在连接到 CICS 时异常终止，那么 CICS 可能会在不知道 IBM MQ 的情况下落实或回退工作。 当队列管理器重新启动时，该工作称为 不确定。

IBM MQ 无法解析这些不确定的恢复单元 (即，落实或回退对 IBM MQ 资源所作的更改) ，直到重新启动或重新连接到 CICS 的连接为止。

在 CICS 适配器启动期间，将启动用于解析不确定恢复单元的过程。 当适配器请求不确定的恢复单元列表时，该进程将启动。 然后:

- 适配器从 IBM MQ 接收此连接标识的不确定恢复单元列表，并将其传递到 CICS 以进行解析。
- CICS 将此列表中的条目与其自己的日志中的条目进行比较。 CICS 从其自己的列表中确定它对每个不确定的恢复单元执行的操作。

对于所有已解析的单元， IBM MQ 会根据需要更新队列并释放相应的锁定。 未解析的单元可以在重新启动后保留。 通过 管理 IBM MQ for z/OS 中描述的方法来解决这些问题。

如何从 IMS 解析不确定的恢复单元

在 IMS 中解析不确定的恢复单元不会影响 DL/I 资源。 IMS 控制恢复协调，并且在重新启动时自动落实或回退不完整的 DL/I 工作。 是否存在 IMS 日志记录类型 X'3730 '和 X'3801' 的联机区域 (非快速路径) 的落实或回退决策。 存在不确定的恢复单元并不意味着在 IBM MQ 连接之前 DL/I 记录处于锁定状态。

在队列管理器重新启动期间， IBM MQ 会生成不确定恢复单元的列表。 IMS 会构建其自己的残差恢复条目 (RREs) 列表。 在解析所有条目之前，会在 IMS 检查点记录 RREs。

在将 IMS 区域重新连接到 IBM MQ 期间， IMS 向 IBM MQ 指示是落实还是回退 IBM MQ 中标记为不确定的工作单元。

解析不确定单元时:

1. 如果 IBM MQ 识别到它已标记要落实的条目，并且 IMS 已将其标记为要回退，那么 IBM MQ 会发出消息 CSQQ010E。 IBM MQ 针对 IBM MQ 与 IMS 之间此类型的所有不一致情况发出此消息。
2. 如果 IBM MQ 具有任何剩余的不确定单元，那么适配器将发出消息 CSQQ008I。

对于所有已解析的单元， IBM MQ 会根据需要更新队列并释放相应的锁定。

IBM MQ 维护对未解决的不确定工作的锁定。 如果挂起重要的锁定，那么这可能会导致系统中的积压。 该连接仍处于活动状态，因此您可以解析 IMS RREs。 通过 管理 IBM MQ for z/OS 中描述的方法恢复不确定线程。

除非存在软件或操作问题 (例如 IMS 冷启动) ，否则应解决所有不确定工作。 IMS 控制区域在两种情况下进行不确定解析:

1. 在与 IBM MQ 的连接开始时，将以同步方式完成解析。
2. 当程序异常终止时，在此期间将以异步方式完成解析。

如何从 RRS 解析不确定的恢复单元

RRS 适配器的其中一个功能是在 IBM MQ 和其他参与 RRS 的资源管理器之间保持数据同步。 如果 IBM MQ 已完成落实的第一阶段，并且正在等待来自 RRS (落实协调程序) 的决策时发生故障，那么恢复单元将进入不确定状态。

在 RRS 与 IBM MQ 之间重新建立通信时， RRS 会自动落实或回退每个恢复单元，这取决于是否存在标记落实开始的日志记录。 在重新建立与 RRS 的连接之前， IBM MQ 无法解析这些不确定的恢复单元 (即，落实或回退对 IBM MQ 资源所作的更改)。

在某些情况下， RRS 无法解决不确定的恢复单元。 发生此情况时， IBM MQ 会将下列其中一条消息发送到 z/OS 控制台:

- CSQ3011I
- CSQ3013I
- CSQ3014I
- CSQ3016I

有关这些消息的含义的详细信息，请参阅 IBM MQ for z/OS 消息，完成和原因码 手册。

对于所有已解析的恢复单元， IBM MQ 会根据需要更新队列并释放相应的锁定。 未解析的恢复单元可以在重新启动后保留。 通过 管理 IBM MQ for z/OS 中描述的方法来解决这些问题。

如何解析具有 GROUP 恢复处置单元的不确定恢复单元

具有 GROUP 恢复处置单元的不确定事务可以由事务协调程序由启用了 GROUPUR 队列管理器属性的队列共享组 (QSG) 中的任何队列管理器解析。 每当事务协调程序重新连接时，它通常会请求任何未完成的不确定事务的列表，然后使用其日志中的信息来解析这些事务。

当已连接到 GROUP 恢复处置单元的事务协调程序请求不确定事务列表时，返回的列表包含在队列共享组中存在的具有 GROUP 恢复处置单元的所有不确定事务。 此列表不依赖于那些不确定事务是在哪个队列管理器上启动的。 处理此类请求的队列管理器通过使用 SYSTEM.QSG.UR.RESOLUTION.QUEUE。 然后，队列管理器会从任何不活动队列管理器的最后一个检查点读取这些队列管理器的日志，以确定如果这些队列管理器处于活动状态，它们将报告的任何其他不确定事务。

当事务协调程序请求解析不确定事务时，它所连接的队列管理器会识别事务是否源自自身，如果是，那么会以与具有 QMGR 恢复处置单元的事务相同的方式解析该事务。 如果事务是在 QSG 中的另一个活动队列管理器上发起的，那么将使用 SYSTEM.QSG.UR.RESOLUTION.QUEUE。 如果事务是在 QSG 中的不活动队列管理器上发起的，那么将立即解析任何共享队列工作，并将解析任何剩余专用队列工作的请求放在 SYSTEM.QSG.UR.RESOLUTION.QUEUE。 在接受新工作之前，不活动的队列管理器会在启动时处理此请求。 在此场景中，原始队列管理器的日志仍反映恢复单元处于不确定状态，直到它重新启动并处理请求为止。

共享队列恢复

使用本主题来了解队列共享组环境中各种组件的 IBM MQ 恢复和弹性。

- 第 370 页的『事务恢复』
- 第 370 页的『对等恢复』

- 第 370 页的『共享队列定义』
- 第 370 页的『日志记录』
- 第 370 页的『耦合设施和结构故障』
- 第 371 页的『结构故障场景』
- 第 372 页的『对耦合设施连接故障的弹性』
- 第 372 页的『管理耦合设施连接故障的弹性』
- 第 374 页的『操作行为』

事务恢复

当应用程序发出 MQBACK 调用或异常终止时 (例如，由于 EXEC CICS ROLLBACK 或 IMS 异常终止)，队列管理器中存储的线程级别信息将确保正在进行的工作单元回滚。 共享队列上同步点内的 MQPUT 和 MQGET 操作将以与对非共享队列的更新相同的方式回滚。

对等恢复

如果队列管理器发生故障，那么它将与当前连接到的耦合设施结构异常断开连接。 如果 z/OS 实例与耦合设施之间的连接失败 (例如，耦合设施或分区的物理链路故障或断电)，那么也会检测到队列管理器与所涉及的耦合设施结构之间的连接异常终止。 保持连接到该结构的同一队列共享组中的其他队列管理器会检测异常断开连接的情况，并尝试对该结构上发生故障的队列管理器启动 对等恢复 。 只有其中一个队列管理器成功启动对等恢复，但所有其他队列管理器都协作恢复失败的队列管理器所拥有的工作单元。

如果队列管理器在没有同级连接到结构时失败，那么当另一个队列管理器连接到该结构时，或者当失败的队列管理器重新启动时，将执行恢复。

对等恢复 (通常称为对等级别恢复 (Peer Level Recovery， PLR)) 是按结构执行的，单个队列管理器可以同时参与多个结构的恢复。 但是，根据发生故障时连接到不同结构的队列管理器，合作恢复不同结构的同级的集合可能有所不同。

当失败的队列管理器重新启动时，它将重新连接到它在发生故障时所连接到的结构，并恢复未由同级恢复恢复的任何剩余未解决的工作单元。

对等恢复是一个多阶段过程。 在第一阶段中，将恢复在未完成阶段之后前进的工作单元; 这可能涉及落实正在落实的工作单元的消息，以及锁定不确定的工作单元的消息。 在第二阶段中，将检查在发生故障的队列管理器中具有活动线程的队列，回滚与未完成的工作单元相关的未落实消息，并重置有关发生故障的队列管理器中共享队列上的活动句柄的信息。 这意味着 IBM MQ 会重置任何指示符，这些指示符指示发生故障的队列管理器已打开共享队列以进行互斥输入，从而允许其他活动队列管理器打开队列以进行输入。

共享队列定义

表示共享队列属性的队列对象保存在队列共享组所使用的共享 Db2 存储库中。 确保已执行足够的过程来备份和恢复用于存放 IBM MQ 对象的 Db2 表。 您还可以使用 IBM MQ CSQUTIL 实用程序来创建 MQSC 命令，以重放到队列管理器中，从而重新定义 IBM MQ 对象，包括存储在 Db2 中的共享队列和组定义。

日志记录

队列共享组可以支持持久消息，因为共享队列上的消息可以记录在队列管理器日志中。

耦合设施和结构故障

对于耦合设施 (CF) 结构，可以报告两种类型的故障: 结构故障和连接丢失。 用于数据共享 (XES) 的综合系统服务向 IBM MQ 通知 CF 结构故障或具有结构故障事件的 CF 故障。 如果 XES 创建了丢失连接事件，这不一定表示结构存在问题，可能是没有可用于与结构通信的连接。 可能并非所有队列管理器都接收到该结构的连

接丢失事件; 这取决于与 CF 的连接的配置。 由于操作员命令 (例如 VARY PATH OFFLINE 或 CONFIG CHP OFFLINE) ， 也可能接收到丢失连接事件。

可以将 IBM MQ 使用的 CF 结构配置为使用系统管理的双工。 这意味着如果发生单个故障，那么系统管理的故障转移处理将隐藏结构故障或连接中断，并且不会通知队列管理器该故障。 如果双工结构或连接的两个实例都发生故障，那么队列管理器将接收相应的事件，并以与单纯形法结构的故障事件相同的方式处理该事件。 方案中描述了队列管理器如何处理事件的详细信息。

在不太可能发生 CF 或结构故障的情况下，存储在受影响应用程序结构中的任何非持久消息都将丢失。 可以使用 RECOVER CFSTRUCT 命令来恢复持久消息。 如果可恢复的应用程序结构已失败，那么将阻止此结构的任何进一步应用程序活动，直到该结构已恢复为止。

要确保可以在合理的时间段内恢复 CF 结构，请使用 BACKUP CFSTRUCT 命令进行频繁备份。 您可以选择在队列共享组中的任何队列管理器上执行备份，或者将一个队列管理器专用于执行所有备份。 自动执行备份过程，以确保定期执行备份。

每个备份都将写入执行备份的队列管理器的活动日志数据集。 共享队列 Db2 存储库记录正在备份的 CF 结构的名称，执行备份的队列管理器的名称，该队列管理器日志上此备份的 RBA 范围以及备份时间。

管理结构包含有关在任何应用程序结构发生故障时共享队列上的不完整工作单元的信息，因此管理结构必须在 RECOVER CFSTRUCT 处理期间可用。 如果管理结构失败，那么队列共享组中的所有队列管理器必须先重建其管理结构条目，然后才能发出 RECOVER CFSTRUCT 命令。 版本 6 和更高版本的队列管理器可能能够在不终止 (取决于故障类型) 的情况下执行此操作，否则队列管理器将在启动时重建其管理结构条目。

要恢复应用程序结构，请向要执行恢复的队列管理器发出 RECOVER CFSTRUCT 命令。 您可以恢复单个 CF 结构，也可以同时恢复多个 CF 结构。 您可以恢复队列共享组中的任何队列管理器，它不必是执行备份的队列管理器，也不必是先前已连接到失败结构的队列管理器。 RECOVER CFSTRUCT 命令使用通过 Db2 存储库信息 (因此必须在执行恢复的队列管理器上提供 Db2) 找到的备份，并将此恢复到故障点。 RECOVER CFSTRUCT 命令通过将来自队列共享组中在备份启动和失败时间之间执行 MQPUT 或 MQGET 的每个队列管理器的日志记录应用于映射到 CF 结构的任何共享队列来执行此操作。 生成的日志合并可能需要读取大量日志数据，因为自读取备份以来参与队列管理器写入的所有日志数据。 强烈建议您进行频繁 (例如，每小时) 备份，尤其是在备份中存在大量消息时。

结构故障场景

方案

如果针对 CF 结构报告了故障，那么已连接的队列管理器所执行的操作取决于以下内容:

- z/OS 的 XES 组件向 IBM MQ 报告的故障类型。
- 结构类型 (应用程序或管理)
- 队列管理器级别 (版本 6.0 ， 版本 7.0 和 版本 7.1)
- MQ CFSTRUCT 对象 (2 ， 3 ， 4 或 5) 的 CFLEVEL。 这不是 CFCC 微码的 CFLEVEL)

以下场景描述了针对管理结构报告故障时发生的情况:

- 如果接收到管理结构的结构故障事件，并且队列管理器在 版本 6.0 和更高版本上运行，那么将自动重新分配并重建结构，而不会终止队列管理器。 由于发生了结构故障，因此未在 CF 中分配该结构，因此当队列管理器尝试连接到该结构时，会由 XES 分配该结构。 当队列管理器已连接到结构的新实例时，队列管理器仅将自身的条目写入结构中。 此处理由队列管理器执行，并且不是 XES 重建处理的一部分。

  任何已连接到队列管理器的序列化应用程序都可以继续处理。 任何尝试与 MQCNO_SERIALIZE_CONN_TAG_QSG 或 MQCNO_RESTRICT_CONN_TAG_QSG 参数连接的序列化应用程序都会接收 MQRC_CONN_TAG_NOT_USABLE 返回码，直到队列共享组中的所有队列管理器都已重建其管理结构条目为止。 将暂挂共享队列上的某些操作，直到队列管理器重新连接到管理结构并完成重新构建结构中的条目为止。 暂挂的操作包括:

  – 打开和关闭共享队列。
  – 正在落实或回退恢复单元。
  – 连接到队列管理器或与队列管理器断开连接的序列化应用程序。

重建队列管理器的管理结构条目后，将恢复暂挂的操作。

只有在队列共享组中的所有队列管理器都已重建其管理结构条目之后，才能备份或恢复应用程序结构。 如果队列管理器在发生故障时未运行，或者在其管理结构部分的恢复完成之前终止，那么队列共享组中的另一个队列管理器将重建其管理结构条目 (如果该队列管理器在 版本 7.0.1 或更高版本上运行)。 如果没有队列管理器在 版本 7.0.1 或更高版本的队列共享组中运行，请重新启动队列管理器，以便它可以完成其部分结构的重建。

以下场景描述了针对应用程序结构报告故障时发生的情况:

- 如果接收到应用程序结构的结构故障事件，并且 CFLEVEL 为 1 或 2 ，那么队列管理器将终止。 重新启动队列管理器。 尝试再次连接到结构的第一个队列管理器会导致 XES 分配该结构的新实例。
- 如果接收到应用程序结构的结构故障事件，并且 CFLEVEL 为 3 ， 4 或 5 ，那么连接到该结构的队列管理器将继续运行。 不使用失败结构中的队列的应用程序可以继续正常处理。 但是，尝试对失败结构中的队列执行操作的应用程序会收到 MQRC_CF_STRUC_FAILED 错误，直到 RECOVER CFSTRUCT 命令成功重建失败结构为止，此时应用程序可以再次打开队列。

对耦合设施连接故障的弹性

什么是耦合设施连接故障的弹性?

对耦合设施连接故障的弹性是指队列共享组中的队列管理器能够容忍与耦合设施结构的连接丢失而不终止。 此功能还会尝试在另一个具有更好连接的耦合设施中重建结构，以便尽快重新获得对共享队列的访问权。

什么是部分丢失连接?

IBM MQ 将部分失去连接定义为以下情况: 综合系统中的一个或多个系统失去与耦合设施的连接，在该耦合设施中分配了系统所访问的结构，但综合系统中的至少一个系统维护与同一耦合设施的连接。

什么是完全失去连接?

IBM MQ 定义了完全失去连接的情况，即综合系统中没有任何系统具有与耦合设施的连接以及在其中分配的结构。

为什么要启用此功能?

对耦合设施连接故障的弹性可提高 IBM MQ 的可用性，允许非共享队列在队列管理器与一个或多个耦合设施结构的连接中断后保持可用。 此外，与耦合设施结构失去连接的队列管理器会自动尝试在另一个可用耦合设施中重建该结构，从而提高队列共享组中共享队列的可用性。

启用此功能时的注意事项

如果没有可用的备用耦合设施，那么允许在未终止的情况下断开与耦合设施结构的连接的队列管理器可能无法在一段时间内重新连接到耦合设施结构。 在失去连接的结构上定义的共享队列将保持不可用状态，直到恢复与该结构的连接为止。 在这种情况下，为了执行共享队列工作而连接到队列共享组成员的应用程序可能会发现他们需要访问的共享队列不可用。 为了避免这种情况，建议将队列管理器配置为在与耦合设施结构的连接中断时终止。 此终止会强制应用程序连接到队列共享组的另一个成员，该成员具有与耦合设施结构的连接，在该耦合设施结构中定义了应用程序所需的共享队列。

管理耦合设施连接故障的弹性

如何启用此功能?

必须执行以下步骤才能启用对耦合设施连接的弹性

1. 确保 CFRM 耦合数据集已格式化为支持系统管理的重建。 这允许队列管理器启动系统管理的重建，以在可用耦合设施中重新创建结构。 使用 **DISPLAY XCF,COUPLE,TYPE=CFRM** 命令可确定 CFRM 耦合数据集的格式。 要支持系统管理的重建，应通过指定以下内容来格式化 CFRM 耦合数据集:

   ```
   "ITEM NAME(SMREBLD) NUMBER(1)"
   ```

   请参阅 z/OS MVS Setting Up a Sysplex 手册，以获取有关格式化 CFRM 耦合数据集的更多信息。

2. 确保备用耦合设施可用并且位于所有 IBM MQ 耦合设施结构的 CFRM 首选项列表中。 这使队列管理器能够尝试将结构重建到备用可用耦合设施中，以尽快恢复对结构的访问权。

   必须在 CFRM 策略中使用 ENFORCEORDER (NO) 定义 IBM MQ 结构，以便如果 IBM MQ 需要重新分配结构，那么 XCF 能够在配置中选择最佳 CF。

   有关结构首选项列表的更多信息，请参阅 z/OS MVS Setting Up a Sysplex 手册。
3. 将队列共享组中的所有队列管理器迁移到最新版本的 IBM MQ ，并在设置了 OPMODE 的情况下启动队列管理器以启用新功能。 在队列共享组的任何成员允许与管理结构的连接中断之前，需要执行此操作。
4. 更改需要容许与 CFLEVEL (5) 的连接丢失的所有应用程序耦合设施结构。 这是可容许丢失连接的最低级别。
5. 确定 **QMGR CFCONLOS** 和 **CFSTRUCT CFCONLOS** 属性所需的值，并相应地更改这些值。 **QMGR CFCONLOS** 属性控制是否允许丢失与管理结构的连接， **CFSTRUCT CFCONLOS** 属性控制每个应用程序耦合设施结构是否允许丢失连接。 如果保留这些属性的缺省值，那么队列管理器将在与任何耦合设施结构的连接中断后终止。
6. 确定每个应用程序耦合设施结构的 **CFSTRUCT RECAUTO** 属性所需的值，并相应地更改这些值。 此属性控制在完全失去连接后，是否应该使用记录的数据自动恢复耦合设施结构。 如果保留此属性的缺省值，那么在完全失去连接后不会对应用程序结构执行自动恢复。

方案 1-与管理结构的连接中断

如果队列共享组中的所有队列管理器都处于 IBM MQ v7.1 或更高版本，那么队列管理器可以容忍与管理结构的连接丢失而不终止。 如果队列共享组中存在级别低于 IBM MQ v7.1 的队列管理器，那么当与管理结构的连接中断时，队列共享组中的所有队列管理器都将异常结束，原因码为 00C510AB 。

当已配置为容许与管理结构的连接丢失的任何队列管理器失去与管理结构的连接时，队列共享组的所有成员都将断开与管理结构的连接。 然后，队列共享组中的所有活动队列管理器尝试重新连接到管理结构，从而使其在耦合设施中重新分配，并具有与综合系统中所有系统的最佳连接，然后重新构建管理结构数据。

注: 这不一定是与具有活动队列管理器的所有系统具有最佳连接的耦合设施。

如果队列管理器无法重新连接到管理结构，例如，由于管理结构的 CFRM 首选项列表中的耦合设施都不可用，那么某些共享队列操作将保持不可用状态，直到队列管理器可以成功重新连接到管理结构并重建其管理结构数据为止。 当合适的耦合设施在系统上变为可用时，将自动进行重新连接。

在队列管理器启动期间，由于缺少与耦合设施的连接，或者没有合适的耦合设施可用于分配该结构，因此无法连接到管理结构。 然后，队列共享组中的所有活动队列管理器尝试重新连接到管理结构，使其在另一个耦合设施 (如果有) 中重新分配，并重建管理结构数据。

方案 2-与应用程序结构的连接中断

在不终止队列管理器的情况下，可以容许与 **CFLEVEL(5)** 或更高版本的应用程序结构的连接中断。 连接到 **CFLEVEL(4)** 或更低版本的应用程序结构的队列管理器，或尚未配置为容许连接丢失的 **CFLEVEL(5)** 结构，当与该结构的连接丢失时，异常终止，原因码为 00C510AB 。

当与已配置为容许丢失连接的应用程序结构的连接丢失时，与该结构的连接断开的所有队列管理器都将断开连接。 队列管理器的后续行为取决于连接丢失是部分还是全部。

部分丢失与应用程序结构的连接

如果确定失去了部分连接，那么失去了与结构的连接的队列管理器会尝试启动系统管理的重建，以便将结构移至另一个具有改进的连接的耦合设施。 如果此重建成功，那么会将结构中的持久消息和非持久消息复制到其他耦合设施，并复原对结构上的队列的访问。 在此期间，未失去连接的队列管理器仍能够访问在受影响的应用程序结构上定义的共享队列，尽管在系统管理的重建过程中操作可能会迁到一些延迟。

如果应用程序结构无法重建到具有改进的连接的另一个耦合设施，或者在另一个耦合设施中重建该结构后，某些队列管理器仍然没有与该结构的连接，那么在恢复到该耦合设施的连接之前，该结构上定义的队列在没有与该结构的连接的队列管理器上仍然不可用。 当队列管理器变为可用时，它会自动重新连接到该结构，并且会恢复对该结构上定义的共享队列的访问权。

与应用程序结构的连接完全中断

如果综合系统中的所有 MVS 系统都与分配了应用程序结构的耦合设施失去了连接，那么只要尝试重新连接到该结构， z/OS 就会从该耦合设施取消分配该结构。 队列管理器可能由于多种原因而尝试重新连接到结构，例如应用程序尝试打开共享队列，或者系统通知新的耦合设施资源可能已变为可用。 因此，在完全失去与应用程序结构的连接之后，可能会丢失受影响结构中的所有非持久消息。

如果使用 **RECAUTO(YES)**定义了可恢复的应用程序结构，那么将在完全失去连接后自动恢复这些结构。 如果备用耦合设施可用于在其中分配结构，或者无论何时此类耦合设施变为可用，那么将立即开始恢复。 如果尚未使用 **RECAUTO(YES)**定义结构，那么可以通过发出 **RECOVER CFSTRUCT** 命令来启动恢复。 这将恢复结构中的所有持久消息，但所有非持久消息都将丢失。 由于此过程涉及读取队列管理器日志，因此可能需要一些时间才能完成，因此建议定期执行结构备份以缩短时间，直到恢复对结构上的共享队列的访问。

一旦应用程序尝试打开在该结构上定义的共享队列或从系统接收到新耦合设施资源变为可用的通知，队列管理器就会尝试重新连接到不可恢复的应用程序结构。 如果有合适的耦合设施可用于分配结构，那么将分配新结构并恢复对该结构上定义的共享队列的访问权。 由于无法将持久消息放入不可恢复结构中定义的队列，因此共享队列上的所有消息都将丢失。

操作行为

如果配置为容许与特定耦合设施结构的连接丢失的 IBM MQ v7.1 或更高版本队列管理器失去连接，那么队列共享组的成员尝试从故障中自动恢复并重新连接到该结构。 此活动可能涉及重新分配另一个耦合设施中的结构，如果一个耦合设施可用，那么该耦合设施具有更好的连接性。 但是，仍可能需要操作员干预以从失去连接的情况中恢复。

通常，必需的操作程序操作是:

1. 解决导致连接丢失的故障原因。
2. 确保可以分配 IBM MQ 结构的耦合设施在综合系统中的所有系统上都可用

在丢失连接事件后在另一个耦合设施中自动重新分配的任何结构都可以移动到耦合设施，并具有与队列共享组中所有队列管理器的最佳连接。 如果需要，可以通过启动 *MVS* 系统命令中记录的系统管理的重建命令 **SETXCF START,REBUILD** 来完成此操作。

如果部分失去与应用程序结构的连接，那么失去与该结构的连接的队列管理器将尝试启动系统管理的重建。 仅当另一个耦合设施具有与当前连接到该结构的所有活动队列管理器的连接时，此进程才会在该耦合设施中分配该结构。 因此，当队列共享组中的大多数队列管理器已失去与应用程序结构的连接时，由于仍连接到原始结构的队列管理器，它们可能无法将该结构重建到另一个耦合设施中。 在这种情况下，可以关闭仍连接到原始结构的队列管理器以允许重建结构，也可以发出 **RESET CFSTRUCT ACTION(FAIL)** 命令以使结构失败。 可以通过发出 **RECOVER CFSTRUCT** 命令在适用的结构上启动恢复。

注: 发生故障并恢复结构时，该结构上的所有非持久消息都将丢失。

z/OS z/OS 上的安全概念

使用本主题来了解安全性对于 IBM MQ 的重要性，以及在系统上没有足够的安全性设置所产生的影响。

为什么必须保护 IBM MQ 资源

IBM MQ 处理可能有价值的信息传输。 应用安全性可确保 IBM MQ 拥有和管理的资源不受未经授权的访问。 这种接触可能导致信息的丢失或泄露。

您应确保任何未经授权的用户或进程都不访问或更改以下任何内容:

- 与 IBM MQ 的连接
- IBM MQ 对象，例如队列，进程和名称列表
- IBM MQ 传输链路
- IBM MQ 系统控制命令 (system control commands)
- IBM MQ 消息
- 与消息关联的上下文信息

为了提供必要的安全性， IBM MQ 使用 z/OS 系统授权工具 (SAF) 将授权请求路由到外部安全性管理器 (ESM)，例如 Security Server (以前称为 RACF)。 IBM MQ 不会对其自身进行安全性验证。 在使用分布式排队或客户机的情况下，您可能需要额外的安全措施，对于这些安全措施， IBM MQ 将提供通道认证记录，通道出口， MCAUSER 通道属性以及 SSL 或 TLS。

允许访问对象的决定由 ESM 作出， IBM MQ 遵循该决定。 如果 ESM 无法做出决策，那么 IBM MQ 会阻止对该对象的访问。

如果不保护 IBM MQ 资源会发生什么情况

如果对安全性不执行任何操作，那么最可能的影响是 所有 用户都可以访问和更改 每个 资源。 这不仅包括本地用户，还包括使用分布式排队或客户机的远程系统上的用户，其中登录安全性控制的严格程度可能低于 z/OS 的通常情况。

要启用安全性检查，必须执行以下操作:

- 安装并激活 ESM (例如， Security Server)。
- 如果您使用的 ESM 不是安全服务器，请定义 MQADMIN 类。
- 激活 MQADMIN 类。

您必须考虑使用混合大小写资源名称是否对您的企业有利。 如果在 ESM 概要文件中使用混合大小写资源名称，那么必须定义并激活 MXADMIN 类。

z/OS 数据集加密

"数据集加密" (DSE) 提供对 z/OS 数据集进行加密的功能，以便这些数据集包含的数据只能由授予特定许可权的用户标识查看或修改。 这将对文件系统中的静态数据进行加密，并防止无意中向具有合法业务需求和管理数据集本身的许可权的用户披露敏感信息。

IBM MQ for z/OS 不支持将 DSE 与为 IBM MQ 消息提供主要持久性机制的活动日志，页集和共享消息数据集 (SMDS) 配合使用。 相反， IBM MQ Advanced Message Security 为 IBM MQ 消息传递提供了端到端加密解决方案，该解决方案包含整个 IBM MQ 网络，动态数据加密，静态数据加密，甚至运行时 IBM MQ 进程中的数据加密。

可以使用 DSE 对 IBM MQ 子系统中使用的其他 VSAM 和连续数据集进行加密。 例如:

- BSDS 顺序文件，用于保存在启动时使用 CSQINPx DDNAMEs 读取的系统配置 (MQSC) 命令
- 使用 CSQINPx DDNAMEs 在启动时读取的其他系统配置 (MQSC) 命令
- IBM MQ 归档日志，通常用于长期归档 IBM MQ 日志数据以进行审计

您可以通过分配使用数据集密钥标签定义的数据类来使用 DSE 进行加密。 有关更多信息，请参阅 规划日志归档存储器。

相关信息

安全概念
通道认证记录
用于使用 z/OS 上的 IBM MQ 对象的权限
在 z/OS 上设置安全性
比较链接级别安全性和应用程序级别安全性
加密安全性协议：SSL 和 TLS
IBM MQ for z/OS 的消息
MQSC 引用

安全控制和选项

您可以指定是否对整个 IBM MQ 子系统开启安全性，以及是否要在队列管理器或队列共享组级别执行安全性检查。 您还可以控制为 API-资源安全性检查的用户标识数。

子系统安全性

子系统安全性是一个控件，用于指定是否对整个队列管理器执行任何安全性检查。 如果您不需要安全性检查 (例如，在测试系统上)，或者如果您对可以连接到 IBM MQ 的所有资源 (包括客户机和通道) 的安全性级别感到满意，那么可以对队列管理器或队列共享组关闭安全性检查，以便不再进行进一步的安全性检查。

这是唯一可以完全关闭安全性并确定是否执行任何其他安全性检查的检查。 即，如果对队列管理器或队列共享组关闭检查，那么不会执行其他 IBM MQ 检查; 如果保持打开状态，那么 IBM MQ 将检查其他 IBM MQ 资源的安全性需求。

您还可以打开或关闭特定资源集 (例如命令) 的安全性。

队列管理器或队列共享组级别检查

您可以在队列管理器级别或队列共享组级别实现安全性。 如果在队列共享组级别实现安全性，那么该组中的所有队列管理器都共享相同的概要文件。 这意味着要定义和维护的概要文件更少，使安全管理更容易。 它还使您能够轻松地将新的队列管理器添加到队列共享组，因为它将继承现有安全概要文件。

如果您的安装需要这两者的组合 (例如，在迁移期间)，或者如果队列共享组中有一个队列管理器需要与该组中的其他队列管理器具有不同级别的安全性，那么也可以实现这两者的组合。

队列共享组级别安全性 (queue-sharing group level security)

将对整个队列共享组执行队列共享组级别安全性检查。 它使您能够简化安全管理，因为它要求您定义较少的安全概要文件。 用户标识使用特定资源的授权在队列共享组级别进行处理，并且与用户标识用于访问该资源的队列管理器无关。

例如，假设服务器应用程序在用户标识 SERVER 下运行，并希望访问名为 SERVER.REQUEST，您要在综合系统中的每个 z/OS 映像上运行 SERVER 实例。 而不是允许 SERVER 打开 SERVER.REQUEST (队列管理器级别安全性)，只能允许在队列共享组级别进行访问。

您可以使用队列共享组级别安全概要文件来保护所有类型的资源，无论是本地资源还是共享资源。

队列管理器级别安全性 (queue manager level security)

您可以使用队列管理器级别的安全概要文件来保护所有类型的资源，无论是本地资源还是共享资源。

两个级别的组合

您可以同时使用队列管理器和队列共享组级别安全性的组合。

您可以覆盖属于该组的特定队列管理器的队列共享组级别安全设置。 这意味着您可以对单个队列管理器执行与对组中其他队列管理器执行的安全性检查级别不同的级别。

有关更多信息，请参阅 用于控制队列共享组或队列管理器级别安全性的概要文件。

控制检查的用户标识数

RESLEVEL 是一个安全服务器概要文件，用于控制为 IBM MQ 资源安全性检查的用户标识数。 通常，当用户尝试访问 IBM MQ 资源时， Security Server 会检查相关用户标识以查看是否允许访问该资源。 通过定义 RESLEVEL 概要文件，您可以控制是否检查零个，一个或两个用户标识 (如果适用)。

这些控件是在连接的基础上完成的，并且是在连接的生命周期内完成的。

每个队列管理器只有一个 RESLEVEL 概要文件。 控制由用户标识对此概要文件具有的访问权实现。

混合大小写或大写 IBM MQ RACF 类

现在，您可以使用混合案例 RACF 概要文件支持，这允许您使用混合案例资源名称并定义 IBM MQ RACF 概要文件以保护它们。

您可以选择:

- 继续仅使用先前发行版中的大写 IBM MQ RACF 类，或者
- 使用新的混合大小写 IBM MQ RACF 类。

如果不使用混合大小写 RACF 概要文件，那么您仍可以在 IBM MQ for z/OS 中使用混合大小写资源名称; 但是，这些资源名称只能由大写 IBM MQ 类中的通用 RACF 概要文件保护。 使用混合大小写 IBM MQ RACF 概要文件支持时，您可以通过在混合大小写 IBM MQ 类中定义 IBM MQ RACF 概要文件来提供更精细的保护级别。

您可以保护的资源

当队列管理器启动时，或者当操作员命令指示时， IBM MQ 确定要保护哪些资源。

您可以控制对每个单独的队列管理器执行哪些安全性检查。 例如，您可以在生产队列管理器上实现多项安全性检查，但在测试队列管理器上不实现任何安全性检查。

连接安全性

当应用程序尝试连接到队列管理器时，将执行连接安全性检查。 此操作是通过发出 MQCONN 或 MQCONNX 请求，或者在通道启动程序或 CICS 或 IMS 适配器发出连接请求时完成的。

如果您正在使用队列管理器级别安全性，那么可以对特定队列管理器关闭连接安全性检查。 但是，如果执行此操作，那么任何用户都可以连接到该队列管理器。

对于 CICS 适配器，仅 CICS 地址空间用户标识用于连接安全性检查，而不是单个 CICS 终端用户标识。 对于 IMS 适配器，当 IMS 控件或从属区域连接到 IBM MQ 时，将检查 IMS 地址空间用户标识。 对于通道启动程序，将检查通道启动程序地址空间所使用的用户标识。

您可以在队列管理器或队列共享组级别开启或关闭连接安全性检查。

命令安全性

当用户从 发出命令中描述的任何源发出 MQSC 命令时，将执行命令安全性检查。 您可以对命令指定的资源进行单独检查，如 第 377 页的『命令资源安全性』中所述。

如果关闭命令检查，那么不会检查命令的签发者以查看他们是否有权发出命令。

如果从控制台输入 MQSC 命令，那么控制台必须具有 z/OS SYS 控制台权限属性。 从 CSQINP1 或 CSQINP2 数据集发出或由队列管理器在内部发出的命令将免除所有安全性检查，而针对 CSQINPX 的命令将使用通道启动程序地址空间的用户标识。 您必须控制允许谁通过正常数据集保护来更新这些数据集。

可以在队列管理器或队列共享组级别开启或关闭命令安全性检查。

命令资源安全性

某些 MQSC 命令 (例如，定义本地队列) 涉及 IBM MQ 资源的操作。 当命令资源安全性处于活动状态时，每次发出涉及资源的命令时， IBM MQ 都会检查是否允许用户更改该资源的定义。

您可以使用命令资源安全性来帮助实施命名标准。 例如，可能允许工资单管理员仅删除和定义名称以 "PAYROLL" 开头的队列。 如果命令资源安全性处于不活动状态，那么不会对该命令正在处理的资源进行安全性检查。 不要将命令资源安全性与命令安全性混淆; 两者是独立的。

关闭命令资源安全性检查不会影响专门针对不涉及命令的其他处理类型执行的资源检查。

您可以在队列管理器或队列共享组级别开启或关闭命令资源安全性检查。

通道安全性注意事项

通道安全性

当您使用通道时，可用的安全功能取决于您要使用的通信协议。 如果使用 TCP，那么虽然可以使用 SSL 或 TLS，但没有随通信协议提供的安全功能。 如果您正在使用 APPC，那么可以将用户标识信息从发送 MCA 通过网络传递到目标 MCA 以进行验证。

对于这两种协议，您可以指定要出于安全目的检查的用户标识以及数量。 同样，可供您使用的选项取决于您正在使用的协议，您在定义通道时指定的内容以及通道启动程序的 RESLEVEL 设置。

有关可用通道安全性类型的更多信息，请参阅 通道认证记录 和 安全性出口概述

相关参考

第 378 页的『API-资源安全性』

当应用程序使用 MQOPEN 或 MQPUT1 调用打开对象时，将检查资源。 打开对象所需的访问权取决于打开队列时指定的打开选项。

API-资源安全性

当应用程序使用 MQOPEN 或 MQPUT1 调用打开对象时，将检查资源。 打开对象所需的访问权取决于打开队列时指定的打开选项。

API-资源安全性细分为以下检查:

- 队列
- 流程
- 名称列表
- 备用用户
- Context

打开队列管理器对象或访问存储类对象时，不会执行安全性检查。

队列

队列安全性检查控制允许谁打开哪个队列，以及允许他们使用哪些选项来打开该队列。 例如，可能允许用户打开名为 PAYROLL.INCREASE.SALARY 用于浏览队列上的消息 (使用 MQOO_BROWSE 选项)，但不从队列中除去消息 (使用 MQOO_INPUT_ * 选项之一)。 如果关闭队列检查，那么任何用户都可以使用任何有效打开选项 (即， MQOPEN 或 MQPUT1 调用上的任何有效 MQOO_ * 选项) 打开任何队列。

您可以在队列管理器或队列共享组级别开启或关闭队列安全性检查。

进程

当用户打开进程定义对象时，将执行进程安全性检查。 如果关闭进程检查，那么任何用户都可以打开任何进程。

您可以在队列管理器或队列共享组级别开启或关闭进程安全性检查。

名称列表

当用户打开名称列表时，将执行名称列表安全性检查。 如果关闭对名称列表的检查，那么任何用户都可以打开任何名称列表。

您可以在队列管理器或队列共享组级别开启或关闭名称列表安全性检查。

备用用户

备用用户安全性控制一个用户标识是否可以使用另一个用户标识的权限来打开 IBM MQ 对象。

例如:

- 在用户标识 PAYSERV 下运行的服务器程序从用户标识 USER1 放入队列的队列中检索请求消息。
- 当服务器程序获取请求消息时，它将处理请求并将应答重新放入与请求消息一起指定的应答队列中。

- 服务器可以指定其他用户标识 (在本例中为 USER1) ，而不是使用自己的用户标识 (PAYSERV) 来授权打开应答队列。 在此示例中，备用用户安全性将控制是否允许用户标识 PAYSERV 在打开应答队列时将用户标识 USER1 指定为备用用户标识。

在对象描述符 (MQOD) 的 *AlternateUserId* 字段中指定备用用户标识。

您可以在任何 IBM MQ 对象 (例如，流程或名称列表) 上使用备用用户标识。 它不会影响任何其他资源管理器 (例如，用于 CICS 安全性或用于 z/OS 数据集安全性) 所使用的用户标识。

如果备用用户安全性未处于活动状态，那么任何用户都可以使用任何其他用户标识作为备用用户标识。

您可以在队列管理器或队列共享组级别开启或关闭备用用户安全性检查。

Context

上下文是适用于特定消息的信息，包含在作为消息一部分的消息描述符 (MQMD) 中。 上下文信息有两个部分:

身份部分
第一次将消息放入队列的应用程序的用户。 它由以下字段组成:

- *UserIdentifier*
- *AccountingToken*
- *ApplIdentityData*

"源" 部分
将消息放入当前存储该消息的队列的应用程序。 它由以下字段组成:

- *PutApplType*
- *PutApplName*
- *PutDate*
- *PutTime*
- *ApplOriginData*

在进行 MQPUT 或 MQPUT1 调用时，应用程序可以指定上下文数据。 应用程序可能会生成数据，可能会从另一条消息传递数据，或者缺省情况下队列管理器可能会生成数据。 例如，服务器程序可以使用上下文数据来检查请求者的身份，即，此消息是否来自正确的应用程序? 通常， *UserIdentifier* 字段用于确定备用用户的用户标识。

您可以使用上下文安全性来控制用户是否可以在任何 MQOPEN 或 MQPUT 调用上指定任何上下文选项。 有关上下文选项的信息，请参阅 与消息上下文相关的 MQOPEN 选项。 有关与上下文相关的消息描述符字段的描述，请参阅 MQMD-消息 descriptorMQMD -消息描述符。

如果关闭上下文安全性检查，那么任何用户都可以使用队列安全性允许的任何上下文选项。

您可以在队列，队列管理器或队列共享组级别打开或关闭上下文安全性检查。

可用性

IBM MQ for z/OS 具有许多用于实现高可用性的功能。 本主题描述了可用性的一些注意事项。

如果队列管理器或通道启动程序发生故障，那么 IBM MQ 的一些功能可以提高系统可用性。 有关这些功能的更多信息，请参阅以下部分:

- 综合系统注意事项
- 共享队列
- 共享通道
- IBM MQ 网络可用性
- 使用 z/OS 自动重新启动管理器 (ARM)
- 使用 z/OS Extended Recovery Facility (XRF)
- 使用 z/OS GROUPUR 属性在队列共享组中进行恢复

- 在何处查找有关可用性的更多信息

综合系统注意事项

在 综合系统中，许多 z/OS 操作系统映像在单个系统映像中进行协作，并使用耦合设施进行通信。 IBM MQ 可以使用综合系统环境的设施来增强可用性。

除去队列管理器与特定 z/OS 映像之间的亲缘关系允许在发生映像故障时在另一 z/OS 映像上重新启动队列管理器。 重新启动机制可以是手动，使用 ARM 或使用系统自动化，如果您确保执行以下操作:

- 所有页集，日志，引导数据集，代码库和队列管理器配置数据集都在共享卷上定义。
- 子系统定义具有综合系统作用域和综合系统中的唯一名称。
- 在 IPL 时，安装在每个 z/OS 映像上的 早期代码 的级别相同。
- TCP 虚拟 IP 地址 (VIPA) 在综合系统中的每个 TCP 堆栈上都可用，并且您已将 IBM MQ TCP 侦听器和入站连接配置为使用 VIPA 而不是缺省主机名。

有关在综合系统中使用 TCP 的更多信息，请参阅 IBM Redbooks 出版物 综合系统中的 *TCP/IP* SG24-5235。

您还可以配置在综合系统中的不同操作系统映像上运行的多个队列管理器，以作为队列共享组运行，这可以利用共享队列和共享通道来实现更高的可用性和工作负载均衡。

共享队列

在队列共享组环境中，应用程序可以连接到队列共享组中的任何队列管理器。 由于队列共享组中的所有队列管理器都可以访问同一组共享队列，因此应用程序不依赖于特定队列管理器的可用性; 队列共享组中的任何队列管理器都可以为任何队列提供服务。 如果由于队列共享组中的所有其他队列管理器都可以继续处理队列而导致队列管理器停止，那么这将提供更大的可用性。 有关共享队列的高可用性的信息，请参阅 第 306 页的『使用共享队列的优点』。

为了进一步增强队列共享组中消息的可用性， IBM MQ 会检测组中的另一个队列管理器是否与耦合设施异常断开连接，并在可能的情况下完成仍处于暂挂状态的该队列管理器的工作单元。 这称为 对等恢复 ，在 第 370 页的『对等恢复』中进行了描述。

对等恢复无法恢复在发生故障时处于不确定状态的工作单元。 您可以使用自动重新启动管理器 (ARM) 来重新启动故障中涉及的所有系统 (例如 CICS， Db2 和 IBM MQ) ，并确保它们都在同一新处理器上重新启动。 这意味着它们可以再同步，并快速恢复不确定的工作单元。 在第 381 页的『使用 z/OS Automatic Restart Manager (ARM)』中对此进行了描述。

共享通道

在队列共享组环境中， IBM MQ 提供了向网络提供高可用性的功能。 通道启动程序使您能够使用网络产品来平衡一组合格服务器之间的网络请求，并隐藏网络中的服务器故障 (例如， VTAM 通用资源)。 IBM MQ 将通用端口用于入站请求，以便可以将连接请求路由到队列共享组中的任何可用通道启动程序。 在第 323 页的『共享通道』中对此进行了描述。

共享出站通道采用它们从共享传输队列发送的消息。 有关共享通道状态的信息保存在整个队列共享组级别的一个位置。 这意味着如果通道启动程序，队列管理器或通信子系统发生故障，那么可以在队列共享组中的其他通道启动程序上自动重新启动通道。 这称为 对等通道恢复 ，在 共享出站通道中进行了描述。

IBM MQ 网络可用性

使用通道将 IBM MQ 消息从队列管理器传输到 IBM MQ 网络中的队列管理器。 您可以在多个级别更改配置，以提高队列管理器的网络可用性以及 IBM MQ 通道检测网络问题和重新连接的能力。

TCP *Keepalive* 可用于 TCP/IP 通道。 它使 TCP 在会话之间定期发送包以检测网络故障。 KAINT 通道属性确定通道的这些包的频率。

AdoptMCA 允许终止由于网络中断而在接收处理中阻塞的通道，并将其替换为新的连接请求。 您可以使用带有 MQSC 实用程序的 ADOPTMCA 队列管理器属性或带有可编程命令格式接口的 AdoptNewMCAType 属性来控制 AdoptMCA 。

ReceiveTimeout 可防止通道在网络接收调用中被永久阻塞。 RCVTIME 和 RCVTMIN 通道启动程序参数确定通道的接收超时特征 (作为其脉动信号间隔的函数)。 有关更多详细信息，请参阅 队列管理器参数 。

使用 z/OS Automatic Restart Manager (ARM)

您可以将 IBM MQ for z/OS 与 z/OS 自动重新启动管理器 (ARM) 结合使用。 如果队列管理器或通道启动程序失败，那么 ARM 将在同一 z/OS 映像上重新启动该队列管理器或通道启动程序。 如果 z/OS 失败，那么整个相关子系统和应用程序组也会失败。 ARM 可以按预定义顺序在综合系统中的另一个 z/OS 映像上自动重新启动所有发生故障的系统。 这称为跨系统重新启动。

ARM 支持在共享队列环境中快速恢复不确定事务。 如果您未使用队列共享组，那么它还会提供更高的可用性。

在发生 z/OS 故障时，可以使用 ARM 在综合系统中的其他 z/OS 映像上重新启动队列管理器。

要启用自动重新启动，必须执行以下操作:

1. 设置 ARM 耦合数据集。
2. 定义您希望 z/OS 在 *ARM* 策略中执行的自动重新启动操作。
3. 启动 ARM 策略。

如果要自动重新启动不同 z/OS 映像中的队列管理器，那么必须使用综合系统范围的唯一 4 字符子系统名称来定义可能重新启动该队列管理器的每个 z/OS 映像中的每个队列管理器。

在 IBM MQ 网络中使用 ARM 中描述了将 ARM 与 IBM MQ 配合使用。

使用 z/OS Extended Recovery Facility (XRF)

您可以在扩展恢复设施 (XRF) 环境中使用 IBM MQ 。 所有 IBM MQ 拥有的数据集 (可执行代码， BSD ，日志和页集) 都必须在活动和备用 XRF 处理器之间共享的 DASD 上。

如果使用 XRF 进行恢复，那么必须在活动处理器上停止队列管理器并在备用处理器上启动该队列管理器。 对于 CICS，可以使用 CICS 提供的命令列表 (CLT) 来执行此操作，或者系统操作员可以手动执行此操作。 对于 IMS，这是手动操作，您必须在协调 IMS 系统完成处理器切换后执行此操作。

必须先完成或终止 IBM MQ 实用程序，然后才能将队列管理器切换到备用处理器。 在规划 XRF 恢复计划时，请仔细考虑此潜在中断的影响。

在活动处理器上的队列管理器终止之前，请注意防止队列管理器在备用处理器上启动。 过早启动可能会导致数据，目录和日志中出现严重的完整性问题。 使用全局资源序列化 (GRS) 可防止在两个系统上同时使用 IBM MQ ，从而帮助避免完整性问题。 必须包含 BSDS 作为受保护资源，并且必须在 GRS 环中包含活动和备用 XRF 处理器。

在队列共享组中使用 z/OS GROUPUR 属性进行恢复

队列共享组 (QSG) 允许使用本主题中描述的其他事务性工具。 GROUPUR 属性允许 XA 客户机应用程序对 QSG 的任何成员执行任何可能需要的不确定事务恢复。

如果 XA 客户机应用程序通过综合系统连接到队列共享组 (QSG) ，那么无法保证它连接到哪个特定队列管理器。 QSG 中的队列管理器使用 GROUPUR 属性可以启用可能需要在 QSG 的任何成员上进行的任何不确定事务恢复。 即使应用程序最初连接到的队列管理器不可用，也可以进行事务恢复。

此功能部件将 XA 客户机应用程序从对 QSG 的特定成员的任何依赖关系中释放出来，从而扩展队列管理器的可用性。 对于事务应用程序， QSG 显示为单个实体，提供所有 IBM MQ 功能部件，并且没有单个队列管理器故障点。

此功能对于事务应用程序并不明显。

在何处查找有关可用性的更多信息

您可以从以下来源找到有关这些主题的更多信息:

表 *28:* 在何处查找有关可用性的更多信息

Topic	**在哪里查找信息**
队列共享组	第 291 页的『共享队列和队列共享组』
系统参数	配置系统参数
使用自动重新启动管理器实用程序	在 IBM MQ 网络中使用 ARM
MQSC 命令	MQSC 命令

监控和统计信息

IBM MQ for z/OS 具有一组用于监视队列管理器和收集统计信息的工具。

IBM MQ 提供了用于监视系统和收集统计信息的工具。 有关这些设施的更多信息，请参阅以下部分:

- 第 382 页的『联机监视』
- 第 382 页的『IBM MQ 跟踪』
- 第 383 页的『事件』

联机监视

IBM MQ 包含用于监视 IBM MQ 对象状态的以下命令:

- DISPLAY CHSTATUS 显示指定通道的状态。
- DISPLAY QSTATUS 显示指定队列的状态。
- DISPLAY CONN 显示指定连接的状态。

有关这些命令的更多信息，请参阅 MQSC 命令。

IBM MQ 跟踪

IBM MQ 提供了可用于在队列管理器运行时收集以下信息的跟踪工具:

性能统计信息

统计信息跟踪收集以下信息以帮助您监视性能并调整系统:

- 不同 MQI 请求的计数 (消息管理器统计信息)
- 不同对象请求的计数 (数据管理器统计信息)
- 有关 Db2 使用情况的信息 (Db2 管理器统计信息)
- 有关耦合设施使用情况的信息 (耦合设施管理器统计信息)
- 有关 SMDS 使用情况的信息 (共享消息数据集统计信息)
- 有关缓冲池使用情况的信息 (缓冲区管理器统计信息)
- 有关日志记录的信息 (日志管理器统计信息)
- 有关存储器使用情况的信息 (存储管理器统计信息)

- 有关锁定请求的信息 (锁定管理器统计信息)

记帐数据

- 记帐跟踪收集有关处理 MQI 调用所耗用的处理器时间以及特定用户发出的 MQPUT 和 MQGET 请求数的信息。
- IBM MQ 还可以使用 IBM MQ 收集有关每个任务的信息。 此数据作为线程级记帐记录收集。 对于每个线程， IBM MQ 还会收集有关该线程所使用的每个队列的信息。

跟踪生成的数据将发送到系统管理设施 (SMF) 或通用跟踪设施 (GTF)。

事件

IBM MQ 事件提供有关队列管理器中的错误，警告和其他重要事件的信息。 通过将这些事件合并到您自己的系统管理应用程序中，可以针对多个 IBM MQ 应用程序监视多个队列管理器中的活动。 尤其是，您可以从单个队列管理器监视系统中的所有队列管理器。

可以通过用户编写的报告机制向支持向操作员呈现事件的管理应用程序报告事件。 事件还使应用程序能够充当其他管理网络 (例如 NetView) 的代理程序，以监视报告并创建相应的警报。

相关信息

使用 IBM MQ 跟踪

使用 IBM MQ 事件

恢复单元处置

某些事务应用程序在连接到队列共享组 (QSG) 中的队列管理器时，可以使用 GROUP (而不是 QMGR) 恢复处置单元，方法是在它们连接时指定 QSG 名称而不是队列管理器名称。 这将通过除去重新连接到 QSG 中的同一队列管理器的需求，使事务恢复更加灵活和稳健。

已使用 QSG 名称连接的应用程序启动的事务也具有 GROUP 恢复处置单元。

当事务应用程序与 GROUP 恢复单元处置连接时，它在逻辑上连接到队列共享组，并且与任何特定队列管理器都没有亲缘关系。 在连接到 QSG 中的任何队列管理器时，可以查询并解决它已启动的已完成落实进程的 phase-1 阶段的任何 2 阶段落实事务 (即，它们处于不确定状态)。 在恢复方案中，这意味着事务协调程序不必重新连接到同一队列管理器，而此时该队列管理器可能不可用。

与 QMGR 恢复单元处置连接的应用程序与它们所连接的队列管理器具有直接亲缘关系。 在恢复方案中，事务协调程序必须重新连接到同一队列管理器以解决任何不确定事务，而不考虑队列管理器是否属于队列共享组。

当应用程序指定队列共享组名，并因此连接到具有 GROUP 恢复处置单元的 QSG 中的队列管理器时， QSG 在逻辑上是单独的资源管理器。 这意味着仅当应用程序与同一恢复处置单元重新连接时，不确定事务才对其可视。 具有 QMGR 恢复处置单元的不确定事务对于已与 GROUP 恢复处置单元连接的应用程序不可见，反之亦然。

相关概念

第 383 页的『启用 GROUP 恢复单元』
队列共享组可以配置并启用对 GROUP 恢复单元的支持。

第 384 页的『应用程序支持』
使用此页面来确定哪些应用程序可以与 GROUP 恢复处置单元连接。

启用 GROUP 恢复单元

队列共享组可以配置并启用对 GROUP 恢复单元的支持。

要在 QSG 中的队列管理器上使用 GROUP 恢复单元，请启用 GROUPUR 队列管理器属性。 有关此概念的更多信息，请先参阅 第 383 页的『恢复单元处置』，然后再阅读本主题的其余部分。

启用 GROUPUR 队列管理器属性后，队列管理器接受具有 GROUP 恢复处置单元的新连接。 如果禁用此属性，那么将不接受具有此处置的新连接，尽管已连接的应用程序在断开连接之前不受影响。

当应用程序与 GROUP 恢复处置单元连接，并询问哪些事务存在疑问或尝试解决队列共享组 (QSG) 中其他位置启动的事务时，它现在所连接的队列管理器必须能够与 QSG 的其他成员通信，以便它可以处理请求。 为此，它使用名为 SYSTEM.QSG.UR.RESOLUTION.QUEUE。 此队列必须位于称为 CSQSYSAPPL 的可恢复应用程序结构上。 该结构必须可恢复，因为在处理解析请求时，持久消息存储在此队列中。

必须先确保已定义耦合设施结构和共享队列，然后才能启用 GROUP 恢复单元。 您可以使用 CSQ4INSS 样本中的定义。 在定义队列或在启动期间检测到队列时，队列共享组中的每个队列管理器都会打开该队列，以便它可以接收入局请求。 如果要删除或移动队列，因为未正确定义该队列，那么可以通过更新队列对象以禁止 MQGET 请求来请求队列管理器关闭其打开的句柄。 当您进行了必要的更正时，允许应用程序再次从队列中获取消息将指示每个队列管理器重新打开该队列。 使用 DISPLAY QSTATUS 命令可识别在队列上打开的句柄。

完成此设置后，您可以在希望事务应用程序能够通过 GROUP 恢复单元处置连接到的每个队列管理器上启用 GROUP 恢复单元。 这不需要是队列共享组中的所有队列管理器，但是如果选择仅在 QSG 的子集上启用此功能，那么必须确保应用程序仅尝试连接到已启用此功能的队列管理器。 有关更多信息，请参阅第 384 页的『应用程序支持』。

当您尝试启用 GROUPUR 队列管理器属性时，将执行许多配置检查。 队列管理器将检查:

- 它属于队列共享组。
- 已根据 CSQ4INSS 中的定义定义了名为 SYSTEM.QSG.UR.RESOLUTION.QUEUE 的共享队列。
- SYSTEM.QSG.UR.RESOLUTION.QUEUE 位于名为 CSQSYSAPPL 的可恢复 CF 结构上。
- 组恢复单元不受其操作方式 (OPMODE) 限制。

如果上述任何检查失败，那么 GROUPUR 属性将保持禁用状态，并返回消息代码。

如果启用了队列管理器属性，那么还会在队列管理器启动时执行这些配置检查。 如果在启动 GROUP 恢复单元期间有任何检查失败，并且队列管理器发出一条消息，指示哪些检查失败。 执行必要的更正操作后，必须重新启用队列管理器属性。

应用程序支持

使用此页面来确定哪些应用程序可以与 GROUP 恢复处置单元连接。

对 GROUP 恢复单元处置的支持仅限于某些类型的事务应用程序，对于这些应用程序， IBM MQ for z/OS 是资源管理器而不是事务协调程序。 当前支持的事务应用程序包括:

- IBM MQ 扩展事务客户机应用程序
- 在应用程序服务器中运行的 IBM MQ classes for JMS 应用程序，例如 WebSphere Application Server。
- 当使用 RESYNCMEMBER (GROUpresYNC) 配置了 CICS MQCONN 资源定义时，在 CICS TS 4.2 中运行的 CICS 应用程序。

相关概念

第 384 页的『IBM MQ 扩展事务客户机应用程序』
使用此页面来确定 IBM MQ 扩展事务客户机应用程序如何使用 GROUP 恢复处置单元。

第 385 页的『CICS 应用程序』
使用此页面来确定 CICS 如何使用 GROUP 恢复处置单元。

IBM MQ 扩展事务客户机应用程序

使用此页面来确定 IBM MQ 扩展事务客户机应用程序如何使用 GROUP 恢复处置单元。

IBM MQ 扩展事务客户机应用程序的一个示例是使用 JMS 并在 WebSphere Application Server 中运行的应用程序，通过 TCP/IP 连接到 IBM MQ ，而不是本地绑定。 这些客户机应用程序通过网络连接 (例如，通过 TCP/IP) 连接到 IBM MQ for z/OS 。 对于这些应用程序，它是为 xa_open 调用中传递的 xa_info 字符串的 QMNAME 参数指定的值，用于指定是使用 QMGR 还是 GROUP 恢复处置单元。 有关 xa_open 的更多信息，请参阅 xa_open 字符串的格式 和 xa_open 的其他错误处理。 对于 JMS 应用程序，通过在 ConnectionFactory 中指定队列共享组 (QSG) 的名称来完成此操作，而不是指定特定队列管理器的名称。

要使 XA 客户机应用程序能够利用 GROUP 恢复单元处置，必须配置 TCP/IP 设置，以允许客户机应用程序路由到 QSG 中已启用 GROUPUR 属性的队列管理器，而不是特定队列管理器。 可用于执行此操作的动态虚拟 IP 地址技术之一是 z/OS SysPlex 分发器。 有关更多详细信息，请参阅 Communications Server 和 动态虚拟

寻址 。 如果要在 QSG 中的队列管理器子集上启用 GROUP 恢复单元，请确保无法将客户机应用程序路由到未启用该应用程序的那些应用程序，这包括版本低于 V7.0.1 的任何队列管理器。

要在 JMS 客户机应用程序中使用组恢复单元处置功能，必须使用 IBM MQ V7.0.1 JMS 客户机库。 较早级别的 JMS 客户机库使用队列管理器 (QMGR) 恢复处置单元。

客户机应用程序不必使用共享通道连接到队列共享组。

CICS 应用程序

使用此页面来确定 CICS 如何使用 GROUP 恢复处置单元。

CICS 4.2 和更高版本在 MQCONN 资源定义中提供组再同步选项 RESYNCMEMBER (GROUpresYNC)。 使用此选项配置的 CICS 可以连接到与该 CICS 区域在同一 LPAR 上运行的队列共享组中的任何合适的队列管理器。 要支持 CICS GROUpresYNC 选项，队列管理器必须在 MQ V7.1 或更高版本上运行，并且必须启用 GROUPUR 支持。

使用 GROUpresYNC 在连接到 MQ 的 CICS 区域中运行的事务将创建具有 GROUP 恢复处置单元的工作单元。

您可以使用 RESYNCMEMBER (GROUpresYNC) 在队列管理器发生故障后启用更快的恢复，因为它使 CICS 区域能够立即连接到在同一 LPAR 上运行的备用合格队列管理器，从而根据需要解决任何不确定事务，而无需等待队列管理器重新启动。

RESYNCMEMBER (GROUPRESYNC) 还会为 CICS 启用更灵活的重新启动选项。 具有配置为使用 GROUpresYNC 和 MQ 共享队列的 MQ 连接的 CICS 区域可以在作为同一队列共享组的成员运行队列管理器的任何 LPAR 上重新启动。

IBM MQ 和其他 z/OS 产品

使用本主题来了解 IBM MQ 如何使用其他 z/OS 产品。

相关概念

第 385 页的『IBM MQ 和 CICS』
您可以通过配置 CICS 适配器和 CICS bridge 将 IBM MQ 与 CICS 配合使用。

第 390 页的『IBM MQ for z/OS 和 WebSphere Application Server』
使用本主题来了解 WebSphere Application Server 对 IBM MQ for z/OS 的使用。

相关参考

第 386 页的『IBM MQ 和 IMS』
使用本主题来了解 IBM MQ 如何使用 IMS。 IMS 适配器允许您将队列管理器连接到 IMS，并使 IMS 应用程序能够使用 MQI。

第 389 页的『IBM MQ 和 z/OS 批处理， TSO 和 RRS 适配器』
使用本主题来了解 IBM MQ 如何使用 z/OS Batch ， TSO 和 RRS 适配器。

IBM MQ 和 CICS

您可以通过配置 CICS 适配器和 CICS bridge 将 IBM MQ 与 CICS 配合使用。

CICS 适配器允许您将队列管理器连接到 CICS，并使 CICS 应用程序能够使用 MQI。

可选的附加 IBM MQ CICS bridge 使应用程序能够运行不使用 MQI 的 CICS 程序或事务。 这意味着您可以将旧应用程序与 IBM MQ 配合使用，而无需对其进行重写。

在 WebSphere MQ 的系统需求中的 z/OS 下指定了所需的最低级别 CICS 。

有关 CICS 的更多信息，请参阅 CICS 产品文档。

相关信息

将 IBM MQ 与 CICS 结合使用

CICS 组连接

CICS 组连接使 CICS 区域能够连接到同一 LPAR 上 IBM MQ 队列共享组的任何活动成员，而不是指定单个队列管理器。 CICS 仍一次连接到单个队列管理器。

LPAR 上至少需要两个队列管理器来支持 CICS 组连接。 使用组连接可提供更高的可用性，因为您不需要特定队列管理器处于活动状态。 CICS 连接到 LPAR 上的队列共享组中的任何队列管理器。

有关更多信息，请参阅有关 MQCONN 资源的 CICS 文档。

CICS 尝试连接到传递的 MQNAME ，就像它是队列管理器一样:

- 如果队列管理器存在并且处于活动状态，那么连接将起作用。
- 如果连接失败， CICS 将查询组中队列管理器的状态，以确定哪些队列管理器在同一 LPAR 上处于活动状态。
- 如果多个队列管理器处于活动状态，那么 CICS 会检查 RESYNCMEMBER (YES) 和 UOW 状态， 以确定 CICS 是需要连接到特定成员，还是应该连接到特定成员，或者如果未处于活动状态，请等待。
- 如果不需要连接到特定成员， CICS 将选择队列管理器 (使用随机算法)。
- CICS 尝试连接到所选队列管理器。
- 如果尝试失败，那么根据返回码， CICS 将选择下一个成员，然后再次执行选择循环。
- 如果没有任何队列管理器处于活动状态，那么 CICS 会向队列管理器列表发出多个连接，并在 ECBLIST 上等待，直到第一个队列管理器变为可用。

相关概念

第 386 页的『CICS 的组恢复单元 (GROUPUR)』
IBM MQ GROUPUR for CICS 为队列共享组中的不确定工作单元提供对等恢复。 一个 IBM MQ 队列管理器可以代表队列共享组中的另一个队列管理器来解析不确定的工作单元。 这意味着如果 CICS 通过组连接重新连接到 QSG 中的另一个队列管理器，那么它可以解决来自先前 IBM MQ 连接的不确定事务。

相关信息

支持 IBM MQ 队列共享组

CICS 的组恢复单元 (GROUPUR)

IBM MQ GROUPUR for CICS 为队列共享组中的不确定工作单元提供对等恢复。 一个 IBM MQ 队列管理器可以代表队列共享组中的另一个队列管理器来解析不确定的工作单元。 这意味着如果 CICS 通过组连接重新连接到 QSG 中的另一个队列管理器，那么它可以解决来自先前 IBM MQ 连接的不确定事务。

如果 CICS 区域正在使用队列管理器，并且队列管理器异常结束，那么将恢复任何不确定事务。 这样就不需要 CICS 区域等待它正在使用的队列管理器重新启动，然后解决任何不确定的工作单元。 这意味着在 LPAR 上至少需要两个队列管理器，以便在第一个队列管理器异常终止时， CICS 可以连接到另一个队列管理器。

CICS MQCONN 定义上的新 RESYNCMEMBER (GROUpresYNC) 设置:

- 使用 IBM MQ 组连接功能和对等恢复。
- 需要启用了 GROUPUR 属性的队列管理器。
- 仍然支持现有 CICS MQCONN RESYNCMEMBER 设置 (YES 和 NO):
  - 使用现有 CICS 组连接功能，而不进行对等恢复。
  - 更改 RESYNCMEMBER 设置将在 CICS 下次连接到 IBM MQ 时生效。

相关概念

第 383 页的『启用 GROUP 恢复单元』
队列共享组可以配置并启用对 GROUP 恢复单元的支持。

IBM MQ 和 IMS

使用本主题来了解 IBM MQ 如何使用 IMS。 IMS 适配器允许您将队列管理器连接到 IMS，并使 IMS 应用程序能够使用 MQI。

可选的其他 IBM MQ - IMS 网桥使应用程序能够运行不使用 MQI 的 IMS 应用程序。 这意味着您可以将旧应用程序与 IBM MQ 配合使用，而无需对其进行重写。

有关这些组件的更多信息，请参阅以下子主题:

相关信息
设置 IMS 适配器
设置 IMS 网桥
操作 IMS 适配器
IBM MQ for z/OS 上的 IMS 和 IMS 桥接应用程序
MQIIH - IMS 信息头

IMS 适配器

IMS 适配器是 IMS 应用程序与 IBM MQ 子系统之间的接口。

IBM MQ 适配器支持不同的应用程序环境通过消息排队网络发送和接收消息。 IMS 适配器是 IMS 应用程序与 IBM MQ 子系统之间的接口。 这使 IMS 应用程序可以使用 MQI。

IMS 适配器接收并解释使用 IMS 提供的外部子系统连接设施 (ESAF) 访问 IBM MQ 的请求。 此工具在 *IMS* 定制指南中进行了描述。 通常， IMS 会自动连接到 IBM MQ ，而无需操作员干预。

IMS 适配器为以以下方式或状态运行的程序提供对 IBM MQ 资源的访问权:

- 任务 (TCB) 方式
- 问题状态
- 非交叉内存模式
- 非访问注册方式

适配器提供从应用程序任务控制块 (TCB) 到 IBM MQ 的连接线程。

The adapter supports a two-phase commit protocol for changes made to resources owned by IBM MQ with IMS acting as the syncpoint coordinator. IMS 不是同步点协调程序的对话 (例如 APPC 保护的 (SYNCLVL = SYNCPT) 对话) 不受 IMS 适配器支持。

适配器还提供触发器监视器事务 (CSQQTRMN)。 在第 387 页的『IMS 触发器监视器』中对此进行了描述。

您可以将 IBM MQ 与 IMS 扩展恢复设施 (XRF) 配合使用，以帮助从 IMS 错误进行恢复。 有关 XRF 的更多信息，请参阅 *IMS Administration Guide: System* 手册。

使用适配器

应用程序和 IMS 适配器在同一地址空间中运行。 队列管理器在其自己的地址空间中是独立的。

必须链接编辑每个向适当的 IMS 语言接口模块发出一个或多个 MQI 调用的程序，并且除非它使用动态 MQI 调用，否则必须链接编辑 IBM MQ 提供的 API 存根程序 CSQQSTUB。 当应用程序发出 MQI 调用时，存根通过 IMS 外部子系统接口将控制权移交给适配器，该接口管理消息队列管理器对请求的处理。

使用 IMS 进行系统管理和操作

授权的 IMS 终端操作员可以发出 IMS 命令来控制和监视与 IBM MQ 的连接。 但是， IMS 终端操作员无法控制 IBM MQ 地址空间。 例如，操作员无法从 IMS 地址空间关闭 IBM MQ 。

限制

使用 IMS 适配器的应用程序内不支持以下 IBM MQ API 调用:

- MQCB
- MQCB_FUNCTION
- MQCTL

IMS 触发器监视器

IMS 触发器监视器 (**CSQQTRMN**) 是 IBM MQ 提供的 IMS 应用程序，用于在发生 IBM MQ 事件时 (例如，将消息放入特定队列时) 启动 IMS 事务。

运作方式

将消息放入应用程序消息队列时，如果满足触发器条件，那么将生成触发器。 然后，队列管理器将消息 (包含一些用户定义的数据) (称为 触发器消息) 写入为该消息队列指定的启动队列。 在 IMS 环境中，您可以启动 CSQQTRMN 实例以监视启动队列，并在触发消息到达时从中检索触发器消息。 通常，CSQQTRMN 通过 INSERT (ISRT) 将另一个 IMS 事务调度到 IMS 消息队列。 启动的 IMS 应用程序从应用程序消息队列中读取消息，然后对其进行处理。 CSQQTRMN 必须作为非消息 BMP 运行。

CSQQTRMN 的每个副本都服务一个启动队列。 启动后，触发器监视器将运行至 IBM MQ 或 IMS 结束。

CSQQTRMN 的 APPLCTN 宏必须指定 SCHDTYP=PARALLEL。

由于触发器监视器是面向批处理的 BMP ，因此触发器监视器启动的 IMS 事务包含以下内容:

- IOPCB 的 LTERM 字段中的空白
- IOPCB 的 "用户标识" 字段中触发器监视器 BMP 的 PSB 名称

如果目标 IMS 事务受安全服务器 (先前称为 RACF) 保护，那么您可能需要将 CSQQTRMN 定义为安全服务器的用户标识。

IMS 网桥

IBM MQ - IMS 网桥是 IBM MQ for z/OS 的组件，允许从 IBM MQ 应用程序直接访问 IMS 系统上的应用程序。

IBM MQ - IMS 网桥启用 隐式 *MQI* 支持。 这意味着您可以重新设计由 3270 连接的终端控制的旧应用程序，使其由 IBM MQ 消息控制，而不必重写，重新编译或重新链接这些应用程序。 网桥是 IMS *Open Transaction Manager Access* (OTMA) 客户机。

在网桥应用程序中， IMS 应用程序中没有 IBM MQ 调用。 应用程序使用 GET UNIQUE (GU) 将其输入发送到 IOPCB ，并使用 ISRT 将其输出发送到 IOPCB。 IBM MQ 应用程序在消息数据中使用 IMS 头 (MQIIH 结构) 来确保应用程序可以像在由非可编程终端驱动时那样执行。 如果您正在使用处理多段消息的 IMS 应用程序，请注意，所有段都应包含在一条 IBM MQ 消息中。

第 388 页的图 80 中说明了 IMS 网桥。

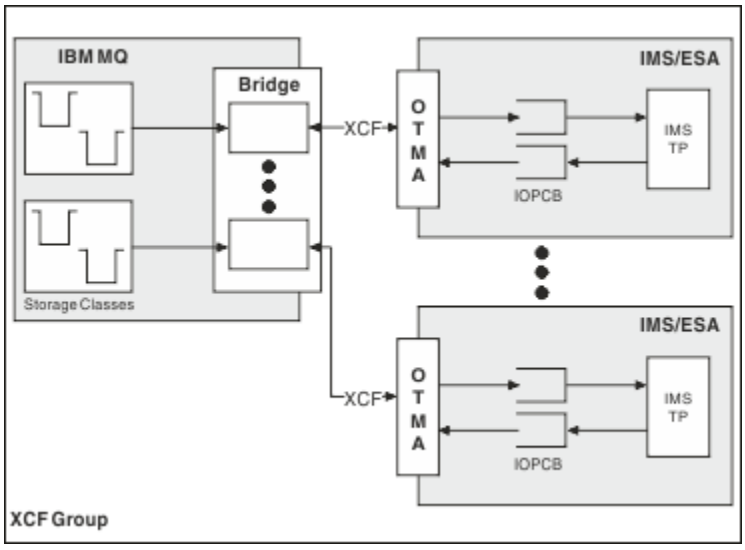


图 *80: IBM MQ - IMS* 网桥

队列管理器可以连接到一个或多个 IMS 系统，多个队列管理器可以连接到一个 IMS 系统。 唯一的限制是它们必须全部属于同一 XCF 组，并且必须全部位于同一综合系统中。

请参阅 设置 IMS 网桥，以获取有关设置 IMS 网桥以及将其他 IMS 连接添加到同一队列管理器的信息。

什么是 OTMA?

IMS OTMA 工具是在 IMS V 5.1 或更高版本上运行的基于事务的无连接客户机/服务器协议。 它充当通过 z/OS 跨系统耦合设施 (XCF) 访问 IMS TM 应用程序的基于主机的通信服务器的接口。

OTMA 使客户机能够连接到 IMS，以便为客户机与 IMS 之间的交互提供高性能，用于大型网络或大量会话。 OTMA 在 z/OS 综合系统环境中实现。 因此， OTMA 的域仅限于 XCF 的域。

OTMA 资源监视

对 IMS v10 或更高版本中提供的 x "3C" OTMA 协议消息的支持已添加到 IBM MQ for z/OS v7.1 中的 IBM MQ - IMS 网桥。 这些消息由 IMS 发送到 OTMA 客户机以报告其运行状态。 如果 IMS 合作伙伴无法处理正在发送的事务请求量，那么它将通知 IBM MQ 已发生洪流警告。 作为响应， IBM MQ 将降低通过网桥发送请求的速率。 如果 IMS 仍无法处理事务请求，并且发生完全洪流情况，那么将暂挂所有到 IMS 合作伙伴的 TPIPE。 当 IMS 合作伙伴通知已解除洪水或洪水警告条件时， IBM MQ 将恢复所有暂挂的 TPIPE (如果适用)，并逐步提高发送事务请求的速率，直到达到最大速率。 控制台消息由 IBM MQ 发出，以响应 IMS 合作伙伴状态的更改。

除非已启用新功能，否则 IBM MQ for z/OS v7.1 将不执行任何操作来响应 x "3C" 消息-请参阅 z/OS: OPMODE。

如果正在使用 IMS v10 合作伙伴，那么应确保已应用 PTF UK45082 。

从 IBM MQ 提交 IMS 个事务

要提交使用网桥的 IMS 事务，应用程序会像往常一样将消息放入 IBM MQ 队列中。 这些消息包含 IMS 事务数据; 它们可以具有 IMS 头 (MQIIH 结构) 或允许 IBM MQ - IMS 网桥对消息中的数据进行假设。

IBM MQ 然后将消息放入 IMS 队列 (它首先在 IBM MQ 中排队，以允许使用同步点来确保数据完整性)。 IBM MQ 队列的存储类确定队列是否为 *OTMA* 队列 (即，用于将消息传输到 IBM MQ - IMS 网桥的队列) 以及将消息数据发送到的特定 IMS 伙伴。

远程队列管理器还可以通过写入 IBM MQ for z/OS 上的这些 OTMA 队列来启动 IMS 事务。

从 IMS 系统返回的数据将直接写入消息描述符结构 (MQMD) 中指定的 IBM MQ 应答队列。 (这可能是到 MQMD 的 *ReplyToQMgr* 字段中指定的队列管理器的传输队列。)

相关参考

第 386 页的『IBM MQ 和 IMS』
使用本主题来了解 IBM MQ 如何使用 IMS。 IMS 适配器允许您将队列管理器连接到 IMS，并使 IMS 应用程序能够使用 MQI。

相关信息

定制 IMS 网桥
IBM MQ for z/OS 上的 IMS 和 IMS 桥接应用程序

IBM MQ 和 z/OS 批处理， TSO 和 RRS 适配器

使用本主题来了解 IBM MQ 如何使用 z/OS Batch， TSO 和 RRS 适配器。

批处理适配器简介

Batch/TSO 适配器是在 JES， TSO 或 UNIX 和 Linux 系统服务和 IBM MQ 下运行的 z/OS 应用程序之间的接口。 它们使 z/OS 应用程序能够使用 MQI。

适配器提供对以以下方式或状态运行的程序的 IBM MQ 资源的访问权:

- 任务 (TCB) 方式

- 问题或管理程序状态
- 非交叉内存模式
- 非访问注册方式

应用程序和 IBM MQ 之间的连接处于任务级别。 适配器提供从应用程序任务控制块 (TCB) 到 IBM MQ 的连接线程。

Batch/TSO 适配器支持对 IBM MQ 拥有的资源进行更改的单阶段落实协议。 它不支持多阶段落实协议。 RRS 适配器使 IBM MQ 应用程序能够与其他支持 RRS 的产品一起参与两阶段落实协议，由 z/OS 资源恢复服务 (RRS) 进行协调。

适配器使用 z/OS STIMERM 服务来调度每秒异步事件。 此事件运行不涉及批处理应用程序任务的任何等待的中断请求块 (IRB)。 此 IRB 检查是否已发布 IBM MQ 终止 ECB。 如果已发布终止 ECB，那么 IRB 将发布正在 IBM MQ 中等待事件 (例如，信号或等待) 的任何应用程序 ECB。

Batch/TSO 适配器

IBM MQ Batch/TSO 适配器为 z/OS 批处理和 TSO 应用程序提供 IBM MQ 支持。 在 z/OS Batch 或 TSO 下运行的所有应用程序都必须具有 API 存根程序 CSQBSTUB 链接-与它们一起编辑。 存根为应用程序提供对所有 MQI 调用的访问权。 通过发出 MQI 调用 MQCMIT 和 MQBACK，可以对应用程序使用单阶段落实和回退。

RRS 适配器

资源恢复服务 (RRS) 是 z/OS 的子组件，提供系统范围的服务，用于在 z/OS 产品之间协调两阶段落实。 IBM MQ Batch/TSO RRS 适配器 (RRS 适配器) 为要使用这些服务的 z/OS 批处理和 TSO 应用程序提供 IBM MQ 支持。 RRS 适配器使 IBM MQ 能够成为 RRS 协调的完整参与者。 应用程序可以与支持 RRS 的其他产品 (例如， Db2) 一起参与两阶段落实处理。

RRS 适配器提供了两个存根; 您必须链接编辑要将 RRS 与其中一个存根配合使用的应用程序。

CSQBRSTB
此存根允许您通过使用 RRS 可调用资源恢复服务 (而不是 MQI 调用 MQCMIT 和 MQBACK) 对应用程序使用两阶段落实和回退。

您还必须从库 SYS1.CSSLIB 与您的应用程序。 如果使用 MQI 调用 MQCMIT 和 MQBACK，那么将接收到返回码 MQRC_ENVIRONMENT_ERROR。

CSQBRRSI
此存根允许您使用 MQI 调用 MQCMIT 和 MQBACK ; IBM MQ 实际上将这些调用实现为 SRRCMIT 和 SRRBACK RRS 调用。

有关构建使用 RRS 适配器的应用程序的信息，请参阅 RRS 批处理适配器。

在何处查找有关 z/OS 批处理， TSO 和 RRS 适配器的更多信息

您可以在以下源中找到有关此部分中主题的更多信息:

表 *29:* 在何处查找有关将 *z/OS Batch* 与 *IBM MQ* 配合使用的更多信息	
Topic	**在哪里查找信息**
设置批处理适配器	任务 19: 设置批处理， TSO 和 RRS 适配器
RRS 可调用资源恢复服务	*MVS Programming: Callable Services for High Level Languages*

IBM MQ for z/OS 和 WebSphere Application Server

使用本主题来了解 WebSphere Application Server 对 IBM MQ for z/OS 的使用。

在 WebSphere Application Server 下运行的以 Java 编写的应用程序可以使用 Java 消息传递服务 (JMS) 规范来执行消息传递。 此环境中的点到点消息传递可由 IBM MQ for z/OS 队列管理器提供。

使用 IBM MQ for z/OS 队列管理器来提供消息传递的好处是，连接 JMS 应用程序可以完全参与 IBM MQ 网络的功能。 例如，他们可以使用 IMS 网桥，或者与在其他平台上运行的队列管理器交换消息。

WebSphere Application Server 与队列管理器之间的连接

您可以为队列连接工厂对象选择 客户机传输 或 绑定传输 。 如果选择绑定传输，那么 WebSphere Application Server 和队列管理器必须同时存在于同一 z/OS 映像上。

请注意，您需要用于 绑定传输的本机库。

这两种类型的连接都支持事务应用程序: 使用 XA 协议的客户机传输; 使用使用 RRS 服务的 WebSphere Application Server 存根 CSQBWSTB 的绑定传输。

有关配置队列连接工厂的更多信息，请参阅 *IBM MQ* 使用 *Java* 。

使用 JMS 应用程序中的 IBM MQ 函数

缺省情况下，保留在 IBM MQ 队列上的 JMS 消息使用 MQRFH2 头来保存某些 JMS 消息头信息。 很多旧 IBM MQ 应用程序不能处理带有这些头的消息，它们需要使用自己的特性头，例如 MQCIH 用于 CICS 桥接应用程序，或者 MQWIH 用于 IBM MQ 工作流程应用程序。 有关这些特殊注意事项的更多详细信息，请参阅 将 JMS 消息映射到 IBM MQ 消息。

方案

每个方案都会指导您完成一组重要任务，并帮助您配置主要产品功能部件。 这些场景包括指向其他内容的有用链接，以帮助您更好地了解感兴趣的区域。

以下子主题描述了可用的 IBM MQ 方案。 *IBM* 产品连接方案和模式 产品文档提供了使用多个 IBM 产品 (例如， IBM MQ 和 WebSphere Application Server) 的工作示例 连接在一起。

相关信息

IBM 产品连接方案和模式产品文档

IBM MQ 入门

此场景说明如何在 Windows 平台上开始使用 IBM MQ 。 如果从未使用过 IBM MQ 并且想要快速入门，请使用此方案。

此方案描述了在 Windows 上安装，配置和验证 IBM MQ (如果尚未将其安装在系统上) 的基本步骤。 您可以使用图形用户界面或命令行界面来完成方案的步骤。

此方案已在 Windows 7 Professional 64 位 (SP 1) 操作系统上使用 IBM MQ 8.0.0 修订包 2 进行测试。

规划解决方案

选择用于在 Windows 上安装 IBM MQ 的方法。 使用引导您完成安装和配置过程的图形用户界面和向导，或者使用命令行来执行静默安装。

概述: 交付的逻辑拓扑

完成方案后交付的逻辑拓扑。

已安装的 IBM MQ 服务器实例允许创建 IBM MQ 对象: 队列和队列管理器。 您可以使用 MQ Explorer 通过队列管理器从本地队列中放入和获取消息。 此场景完成后，交付的拓扑将类似于 图 1。

© Copyright IBM Corp. 2007, 2023

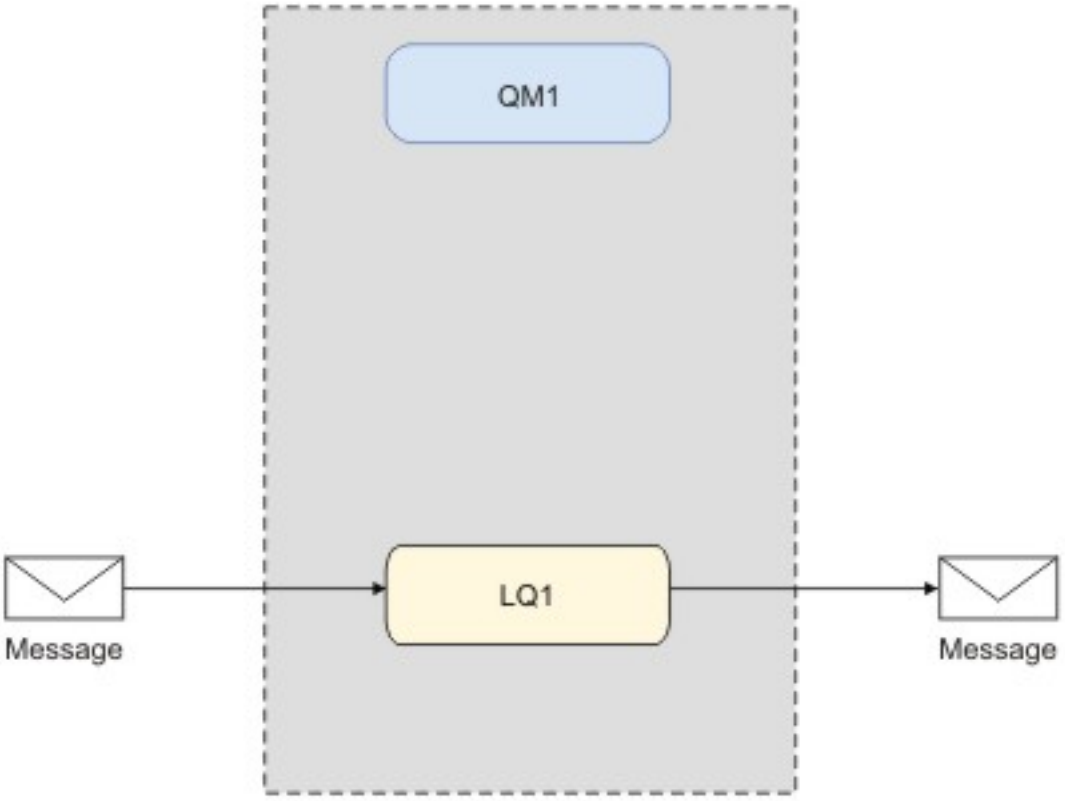


图 *81:* 将消息放在 *LQ1* 上，从 *LQ1* 获取消息。

基本概念和主要术语

描述了开始使用 IBM MQ 方案之前必须了解的基本概念和主要术语。

基本概念

通过 IBM MQ，应用程序可以针对队列读写消息。 读消息的应用程序和写消息的应用程序是相互独立的。 不要求两个应用程序同时运行。 如果没有可用的读消息应用程序，那么消息将在 IBM MQ 队列上排队，直到应用程序读取该消息。

在此方案中，您可以选择通过以下某种方式来安装和配置 IBM MQ：

第 395 页的『使用图形用户界面安装和配置』

在使用图形用户界面进行安装期间，您将通过几个向导进行指导，以帮助您应用相关选项和设置：

启动板
检查软件需求，指定网络信息并启动 IBM MQ 安装向导。

IBM MQ 安装向导
安装软件并启动“准备 IBM MQ”向导。

准备 IBM MQ 向导
启动 IBM MQ 服务和 MQ Explorer。

MQ Explorer
管理队列和队列管理器，访问“缺省配置”向导和 Postcard 应用程序。

缺省配置向导
创建 IBM MQ 对象，将消息放入队列或从队列获取消息，以测试安装是否成功。

Postcard 应用程序
在两个用户之间交换消息以验证安装。

第 400 页的『使用命令行界面安装和配置』

命令行界面安装可以是静默安装或者交互式安装。 静默安装完全可访问，本方案中涉及的即为静默安装。 使用命令行安装期间，系统将通过几个步骤指导您应用相关选项和设置：

- 安装 IBM MQ
- 创建和配置 IBM MQ 对象、队列管理器和队列。
- 使用 `amqsput` 将消息放入队列，并使用 `amqsget` 从队列获取消息，以验证安装。

使用 MQ Explorer 和命令行创建 IBM MQ 对象，也可以使用可编程接口完成此操作。 当前的方案中未包括此内容。

关键术语

下面是关于消息排队的关键术语列表。
有关消息排队的关键术语。

术语	描述
队列管理器	队列管理器负责维护它所拥有的队列，并负责将它接收到的所有消息存储到适当的队列中。
消息	消息是对使用它的应用程序有意义的以字节为单位的字符串。 消息用来将信息从一个应用程序传输到另一个应用程序。 应用程序可以在相同或不同的计算机上运行。
本地队列	本地队列是用于存储消息的数据结构。 该队列可以是常规队列或者传输队列。 直接从队列管理器读取消息的应用程序要读取的消息保留在常规队列中。 正在转移给其他队列管理器的消息保留在传输队列中。

实现解决方案

实现方案的解决方案。 在 Windows 上安装 IBM MQ 并创建 IBM MQ 对象: 队列管理器和队列。 使用样本应用程序通过将消息放入队列以及从队列获取消息来验证安装。

使用图形用户界面安装和配置

使用安装启动板在 Windows 上安装 IBM MQ，然后使用 Postcard 应用程序来验证安装。 验证安装后，创建队列管理器和队列，然后尝试将消息放入队列并从队列中获取消息。

此方案已在 Windows 7 Professional 64 位 (SP 1) 操作系统上使用 IBM MQ 8.0.0.0 进行测试。

使用启动板进行安装

使用安装启动板在 Windows 上安装 IBM MQ，并使用 Postcard 应用程序验证安装。

开始之前

在完成此任务之前，请完成以下检查：

- 安装时必须具有本地管理员权限。 通过 Windows 工具定义此权限。
- 确保机器名不包含任何空格。
- 确保您有足够的磁盘空间 (最多 1005 MB) 来完全安装 IBM MQ for Windows。
- 确定是否需要为任何 IBM MQ 用户定义 Windows 域用户标识。

在安装 IBM MQ 之前，请检查系统是否满足硬件和软件需求。 有关所有受支持平台上的硬件和软件需求的最新详细信息，请参阅 IBM MQ 的系统需求。

关于此任务

启动板和后续向导将引导您完成安装过程，并帮助您查看软件需求和 IBM MQ 设置。 您还将完成创建缺省队列管理器 (本地队列) 的初始任务，并且可以使用 "明信片" 应用程序来验证安装。

此任务假定您是首次在机器上安装 IBM MQ，并且您将使用缺省位置。 缺省情况下， IBM MQ 程序文件的位置为 C:\Program Files\IBM\WebSphere MQ，数据和日志文件位置为 C:\ProgramData\IBM\MQ。

注: 如果要安装 IBM MQ 8.0，并且您的机器上有任何先前安装的 IBM MQ，那么程序和数据文件的位置将与缺省位置不同。 有关更多信息，请参阅 程序和数据目录位置。 如果您先前已完成此方案，并且想要使用缺省位置通过单个全新安装重复此方案，请先除去先前的安装，然后再再次启动此方案。 要从机器卸载 IBM MQ 的现有实例，请参阅 第 405 页的『卸载 IBM MQ』。

如果您在安装过程中需要安装程序，那么安装程序包含指向进一步信息的链接。

过程

1. 启动启动板，查看并在必要时修改软件需求和网络配置。
   a) 浏览至 IBM MQ 软件目录，然后双击文件 Setup.exe 以启动启动板。
   b) 选择 **软件需求** 选项卡以显示 **软件需求** 设置。
   c) 检查是否已满足软件需求，以及需求条目是否显示带有 OK 字样的绿色刻度标记。 进行任何指示的更正。

      注:

      有关任何需求的详细信息，请单击该复选框以展开 "信息" 选项卡。
   d) 选择 **网络配置** 选项卡以显示 **网络配置** 设置。
   e) 选择**无**。

      注: 此场景假定您不需要为 IBM MQ 配置域用户标识。 有关配置 IBM MQ for Windows 域用户的更多信息，请单击 **更多信息**。
   f) 在启动板的 **IBM MQ 安装** 选项卡上，选择安装语言，然后单击 **启动 IBM MQ 安装程序** 以启动 IBM MQ 安装向导。

   您已通过满足或指定安装需求来完成设置 IBM MQ，并且已启动 IBM MQ 安装向导。
2. 使用 IBM MQ 安装向导来安装软件，然后启动 "准备 IBM MQ " 向导。
   a) 在 "准备 IBM MQ " 向导中，阅读 "许可协议"，单击 **我接受许可协议中的条款** 复选框，然后单击 **下一步** 按钮。
   b) 单击 **典型**，然后单击 **下一步**。
   c) 在 " **准备安装 IBM MQ** " 页面中，查看安装信息，然后单击 **安装** 按钮。

      注: 请注意以下详细信息:

      - 安装名称
      - 程序文件的顶级文件夹
      - 数据文件的顶级文件夹

      安装了以下功能部件:

      - IBM MQ 服务器
      - IBM MQ: 用于管理和监视 IBM MQ 资源的图形界面
      - Java 和 .NET 消息传递和 Web Service
      - IBM MQ 开发工具箱

      安装过程开始。 根据您的系统，安装过程可能需要几分钟时间。

      在安装过程结束时， " IBM MQ 安装" 窗口将显示消息 Installation Wizard Completed Successfully。
   d) 单击**完成**。

   您已成功安装 IBM MQ。 "准备 IBM MQ " 向导将自动启动，并显示 **欢迎使用 "准备 IBM MQ** 向导" 页面。
3. 使用 "准备 MQ " 向导来启动 IBM MQ 服务并启动 "缺省配置" 向导。

注:

如果已创建其他队列管理器，那么无法创建缺省配置; 必须先删除其他队列管理器，然后运行 "缺省配置" 向导。 要删除队列管理器，请参阅 删除队列管理器的步骤

a) 在 "欢迎使用准备 IBM MQ 向导" 上，选择 **下一步**。

"准备 IBM MQ 向导" 会显示消息 `Status: Checking IBM MQ Configuration` 和进度条。 完成此过程后，将显示 " IBM MQ 网络配置" 页面。

b) 在 " IBM MQ 准备 IBM MQ " 向导的 "网络配置" 页面上，选择 **否**。

c) 单击**下一步**。

"准备 IBM MQ 向导" 显示消息 `Status: starting the IBM MQ Service` 和进度条。 完成此过程后，向导将显示 "完成准备 IBM MQ 向导"。

d) 选择 **启动 IBM MQ 资源管理器** 并选择是否查看发行说明，然后单击 **完成** 按钮。

IBM MQ Explorer 将启动。

已安装 IBM MQ。 您还启动了 MQ Explorer。

4. 可选: 如果要使用 IBM MQ Postcard 应用程序来验证安装，请创建缺省配置。

a) 如果尚未显示 "内容" 页面，请单击 **窗口** > **显示视图** > **MQ Explorer -内容** 以显示该页面。

b) 单击 **创建缺省配置**。

此时将打开 " IBM MQ 缺省配置" 窗口。

c) 单击 **设置缺省配置**。

此时将打开 "缺省配置向导"。

d) 再次单击 **下一步** 和 **下一步** 以在信息页面之间移动。

e) 在 "缺省配置" 页面上，清除 **允许远程管理队列管理器** 和 **将队列管理器加入缺省集群** 选项，然后单击 **下一步**。

使用 "明信片" 应用程序时，请记下队列管理器名称，因为稍后将需要此名称。

f) 在“摘要”页面上，单击**完成**。

将关闭 "缺省配置向导" ，并显示一个对话框，其中显示消息 `Setting up the default default configuration.` 完成此操作后，焦点将返回到 IBM MQ "缺省配置" 对话框，并显示以下消息: `Default configuration is partially complete` 。

注: IBM MQ 缺省配置还可能显示以下消息: `Join the default cluster by clicking "Join default cluster" to complete the default configuration on this computer` 。 仅当您想要加入集群时才需要选择此选项。 此场景中未包含此内容，因为它超出了此讨论的范围。

g) 单击**关闭**。

现在已设置 "缺省配置" ，您已准备好验证安装。

5. 可选: 如果创建了缺省配置，请使用随 IBM MQ 提供的 Postcard 应用程序来验证安装。 您可以启动 Postcard 应用程序的两个实例，并在它们之间交换消息。

如果未创建缺省配置，那么仍可以执行以下步骤来验证安装，但必须首先配置队列管理器，如此方案的先前任务中所述。

注:

在非缺省配置上运行 Postcard 应用程序会自动在队列管理器上创建名为 `postcard` 的队列。 您可以在使用明信片应用程序后删除此队列。

a) 如果尚未显示 "内容" 页面，请单击 **窗口** > **显示视图** > **MQ Explorer -内容** 以显示该页面。

b) 单击 **启动明信片** 以打开 "明信片-登录" 窗口。

c) 输入第一个用户的昵称，例如: `Jim` ，然后单击 **确定**。

d) 在 " IBM MQ 明信片网络" 窗口中，选择 **仅在此计算机上继续** ，然后单击 **确定**。

将打开 "jim" 的明信片应用程序。

e) 将 `Jim` 的 "明信片" 移动到屏幕的一侧，然后启动另一个 "明信片"。

f) 单击 **启动明信片** 以打开 "明信片-登录" 窗口。

g) 输入第二个用户的昵称，例如: Sue ，然后单击 **确定**。

h) 在 " IBM MQ 明信片网络" 窗口中，选择 **仅在此计算机上继续** ，然后单击 **确定**。

将打开 "sue" 的明信片应用程序。

i) 在 "明信片-" jim " 窗口中，在 **收件人:** 文本框中输入 sue 。

j) 在 "明信片-" jim " 窗口中，输入您在步骤 4e 中定义的队列管理器的名称，或者从 **消息:** 文本框下的条目 **On:** 中获取。

k) 在 "明信片-" jim " 窗口中输入消息，例如 **消息:** 文本框中的 `Hi Sue!`

l) 单击 **发送** 按钮以将消息发送给用户 Sue。

m) 在 "明信片-告" 应用程序窗口中观察用户 Sue 接收到的消息。 要查看接收到的消息，请双击 "已发送和已接收的明信片" 网格中的条目。

注: 单击此 "明信片" 应用程序窗口和其他 "明信片" 应用程序窗口中的 **帮助** 以查看有关运行 "明信片" 应用程序的更多指示信息。

您已使用 "明信片" 应用程序验证了 IBM MQ 安装。

结果

已安装并验证 IBM MQ ，并且您已准备好配置诸如队列管理器和队列之类的对象。

下一步做什么

遵循 第 398 页的『创建名为 QM1 的队列管理器』中的指示信息。

相关信息

磁盘空间需求

Windows 系统上的硬件和软件需求

IBM MQ 简介

安装 IBM MQ 服务器

使用 "明信片" 应用程序验证安装

安装后任务

创建名为 *QM1* 的队列管理器

使用 MQ Explorer 创建名为 QM1 的队列管理器。 队列管理器是 IBM MQ 消息传递网络中的主要组件。

开始之前

必须已安装 IBM MQ 。 如果未执行此操作，请参阅 第 395 页的『使用启动板进行安装』 以获取有关如何执行此操作的信息。

关于此任务

在此示例中，所有名称都以大写形式输入，并且由于 IBM MQ 名称区分大小写，因此您还必须以大写形式输入所有名称。

要使用 MQ Explorer 创建和启动队列管理器，请完成以下步骤。

过程

1. 以管理员身份启动 MQ Explorer 。
2. 在 **Navigator** 视图中，右键单击 **队列管理器** 文件夹，然后单击 **新建> 队列管理器**。 将启动 " **创建队列管理器** " 向导。
3. 在 **队列管理器名称** 字段中，输入 QM1。
4. 选中 `Make this the default queue manager` 复选框。
5. 在 **死信队列** 字段中，输入 SYSTEM.DEAD.LETTER.QUEUE。

   这是创建队列管理器时自动创建的死信队列的名称。
6. 将其他字段留空，然后单击 **完成**，如果该按钮处于禁用状态，请单击 **下一步**。

如果端口号与现有队列管理器 (例如，作为缺省配置的一部分创建的队列管理器) 冲突，那么将禁用 **完成** 按钮。 您必须继续完成向导以更改缺省端口号。

7. 如果单击了 **下一步**，请继续接受缺省值，并在每个页面上单击 **下一步** ，直到到达向导的最终页面 (当 **完成** 按钮变为可用时)。 更改指定的端口号 (例如，更改为 1415) ，然后单击 **完成**。

   创建并启动队列管理器时， IBM MQ 将显示 " **创建队列管理器** " 对话框窗口。

下一步做什么

要创建队列，请参阅 第 399 页的『创建名为 LQ1 的队列』。

相关信息

在分布式平台上创建和管理队列管理器

创建名为 *LQ1* 的队列

使用 MQ Explorer 创建队列。 队列是用于存储消息的数据结构，并且是 IBM MQ 队列管理器对象。

关于此任务

在此任务中，您可以使用 MQ Explorer 创建 IBM MQ 对象。

要使用 MQ Explorer 创建和启动队列，请完成以下步骤。

过程

1. 在“**导航器**”视图中，展开**队列管理器**文件夹。
2. 展开队列管理器 **QM1**。
3. 右键单击 **队列** 文件夹，然后单击 **新建> 本地队列 ... 新建本地队列** 向导将启动。
4. 在 **名称** 字段中，输入 LQ1。
5. 单击**完成**。

   新队列 LQ1 将显示在 " **内容** " 视图中。 如果该队列未显示在 " **内容** " 视图中，请单击 **内容** 视图顶部的 **刷新** 按钮。

下一步做什么

您已准备好将消息放入队列。 要将消息放入队列中，请参阅 第 399 页的『将消息放入队列 LQ1』。

将消息放入队列 *LQ1*

使用 MQ Explorer 将消息放入队列 LQ1 。

关于此任务

此任务假定您已创建名为 QM1 的队列管理器 (如 第 402 页的『创建名为 QM1 的队列管理器』 中所述) 和名为 LQ1 的队列 (如 第 399 页的『创建名为 LQ1 的队列』中所述)。

要使用 MQ Explorer 将消息放入队列，请完成以下步骤。

过程

1. 在“**导航器**”视图中，展开**队列管理器**文件夹。
2. 展开您创建的队列管理器 QM1。
3. 单击**队列**文件夹。 队列管理器的队列在 "内容" 视图中列出。
4. 在 "内容" 视图中，右键单击本地队列 LQ1，然后单击 **放置测试消息 ...**。

   此时会打开“**放入测试消息**”对话框。
5. 在 **消息数据** 字段中，输入一些文本，例如 Hello World，然后单击 **放入消息**。

   系统会清空**消息数据**字段，并将消息放入队列。
6. 单击**关闭**。

在 "内容" 视图中，请注意 LQ1 **当前队列深度** 值现在为 1。 如果 **当前队列深度** 列不可见，那么您可能需要滚动到 " **内容视图**" 的右侧。

下一步做什么

要从队列获取消息，请参阅 第 400 页的『从队列 LQ1 获取消息』。

从队列 *LQ1* 获取消息

使用 MQ Explorer 从队列 LQ1 获取消息。

关于此任务

此任务假定您已放入消息 QM1 ，如 第 399 页的『将消息放入队列 LQ1』中所述。

要使用 MQ Explorer 从队列中获取消息，请完成以下步骤。

过程

1. 在 " **Navigator** " 视图中，展开 **队列管理器** 文件夹，然后展开 QM1。
2. 单击**队列**文件夹。
3. 在 " **内容** " 视图中，右键单击本地队列 LQ1，然后单击 **浏览消息 ...**。 将打开 **消息浏览器** 以显示当前位于 QM1 上的消息的列表。
4. 双击最后一条消息以打开 "属性" 对话框。

   在属性对话框的**数据**页上，**消息数据**字段以人类可读的格式显示了消息的内容。

下一步做什么

遵循后续场景中的指示信息以探索进一步的 IBM MQ 功能。

要了解有关编写排队应用程序，连接到队列管理器以及与队列管理器断开连接，发布/预订以及打开和关闭对象的信息，请参阅 编写用于排队的过程应用程序。

使用命令行界面安装和配置

使用命令行在 Windows 上安装 IBM MQ 以执行静默安装并设置环境变量。 验证安装后，创建队列管理器和队列，然后尝试将消息放入队列并从队列中获取消息。

关于此任务

此方案已在 Windows 7 Professional 64 位 (SP 1) 操作系统上使用 IBM MQ 8.0.0.2 进行测试。

使用静默安装进行安装

使用命令行在 Windows 上安装 IBM MQ ， 以执行静默安装，并确认已正确设置安装环境。

开始之前

在启动此任务之前，请完成以下检查:

- 安装时必须具有本地管理员权限。 通过 Windows 工具定义此权限。
- 确保机器名不包含任何空格。
- 确保您有足够的磁盘空间。 您最多需要 1005 MB 才能完全安装 IBM MQ 8.0 for Windows。
- 确定是否需要为任何 IBM MQ 用户定义 Windows 域用户标识。

在安装 IBM MQ 之前，请检查系统是否满足硬件和软件需求。 有关所有受支持平台上的硬件和软件需求的最新详细信息，请参阅 IBM MQ 的系统需求。

关于此任务

此方案假定您是首次在机器上安装 IBM MQ，并且您正在使用缺省位置。 缺省情况下， IBM MQ 8.0 程序文件的位置为 `C:\Program Files\IBM\WebSphere MQ`，数据和日志文件位置为 `C:\ProgramData\IBM\MQ`。

注: 如果您的机器上有任何先前安装的 IBM MQ，那么程序和数据文件的缺省位置可能会更改。 有关更多信息，请参阅 程序和数据目录位置。 如果您先前已完成此方案，并且想要使用缺省位置通过单个全新安装重复此方案，请先除去先前的安装，然后再再次启动此方案。 要从机器卸载 IBM MQ 的现有实例，请参阅 第 405 页的『卸载 IBM MQ』。

Windows 上的 IBM MQ 使用 MSI 技术来安装软件。 有关使用 MSI 技术进行安装的更多信息，请参阅 使用 msiexec 进行高级安装。

要使用命令行安装 IBM MQ，必须指定以下参数:

- **/i "<WMQ_INSTALLATION_MEDIA>\MSI\IBM WebSphere MQ.msi"** where <WMQ_INSTALLATION_MEDIA> is the location of the `IBM WebSphere MQ.msi` file. 此参数指定 .msi 文件的位置。
- **/l*v <USER_LOGFILE_LOCATION>\install.log** where <USER_LOGFILE_LOCATION> is where you want the installation logs to be written to.
- **/q** 此参数必须用于执行静默安装。
- **USEINI="<RESPONSE_FILE>"** where <RESPONSE_FILE> is the name and location of the response file to be used by the silent installation. 此方案使用样本 `Response.ini` 文件，该文件包含在 IBM MQ 安装介质中。
- **TRANSFORMS="<TRANSFORM_FILE>"** where <TRANSFORM_FILE> is the name of the transform file to be applied to the installation. 此方案使用美国英语变换 `1033.mst`。
- **AGREETOLICENSE="YES"** 必须包含此参数，否则安装无法完成。
- **ADDLOCAL="Server"** 此参数列出要安装的组件。

过程

1. 使用命令行来执行静默安装。
   a) 要从提升的命令提示符调用静默安装，请单击 **Windows 任务栏** 上的 **启动按钮**，并在 **搜索程序和文件** 字段中输入 `cmd`。 右键单击 **cmd.exe** 程序，然后选择 **以管理员身份运行**。
   b) 在 Windows 命令提示符中，输入以下命令:

   注: 此命令显示在此处的多行上，但必须在一行上输入此命令。

   ```
   msiexec /i "<MQ_INSTALLATION_MEDIA>\MSI\IBM WebSphere MQ.msi"
   /l*v c:\wmqinslogs\install.log
   /q USEINI="<MQ_INSTALLATION_MEDIA>\Response.ini"
   TRANSFORMS="1033.mst"
   AGREETOLICENSE="yes"
   ADDLOCAL="Server"
   ```

   Where <MQ_INSTALLATION_MEDIA> is the path to your IBM MQ installation media.

   注:

   输入命令后，命令行将返回提示符。

   c) 要查看安装进度，请打开指定的日志文件。 如果安装成功完成，那么您将看到消息 `Product: IBM MQ (Installation1) -- Installation operation completed successfully.` 从日志文件底部向上显示两段。
   d) 安装完成后，服务将启动， IBM MQ 图标将显示在系统托盘中。
   您已安装 IBM MQ，并且已启动 IBM MQ 服务。
2. 使用 **setmqenv** 命令为安装设置环境变量。
   a) 在命令行中输入以下命令:

注: 如果使用了缺省位置，那么安装的路径将为 C:\Program Files\IBM\WebSphere MQ。

```
"<MQ_INSTALLATION_PATH>/bin/setmqenv" -s
```

where <MQ_INSTALLATION_PATH> refers to the location where IBM MQ is installed. 确保将 bin 文件夹中 **setmqenv** 的路径括在引号中，以防止提示返回错误。

b) 通过输入以下命令来检查是否正确设置了环境:

```
dspmqver
```

如果该命令成功完成，并且返回了期望的版本号和安装名称，那么将正确设置环境。 对于此场景，消息应包含以下行:

```
Version: 8.0.0.2
```

如果未指定非缺省安装名称，那么该行:

```
InstName: Installation1
```

您已使用静默安装成功安装 IBM MQ 。

结果

您已执行 IBM MQ 静默安装，并确认您的环境已正确设置。

下一步做什么

- 您可以运行 "准备 WebSphere MQ " 向导。 有关更多信息，请参阅 准备 IBM MQ 向导。
- 按照第 402 页的『创建名为 QM1 的队列管理器』中的指示信息操作。

如果在安装期间迂到任何问题，请在 **msiexec** 命令中指定的位置检查安装日志，在此场景中，日志文件的位置为: c:\wmqinslogs\install.log。 请执行日志中指定的任何操作，然后重新运行安装。 您还可以检查随命令传递的参数，以屏蔽确保包含所有必需参数。

相关信息

使用 msiexec 进行高级安装
将变换与 msiexec 配合使用
安装 IBM MQ

创建名为 *QM1* 的队列管理器

使用命令行界面创建名为 QM1 的队列管理器。 队列管理器是 IBM MQ 消息传递网络中的主要组件。

开始之前

必须已安装 IBM MQ 。 如果未执行此操作，请参阅第 400 页的『使用静默安装进行安装』以获取有关如何执行此操作的信息。

关于此任务

在此示例中，所有名称都以大写形式输入，并且由于 IBM MQ 名称区分大小写，因此您还必须以大写形式输入所有名称。

过程

1. 以管理员身份打开命令提示符。
2. 通过输入以下命令，创建名为 QM1 的队列管理器:

```
crtmqm QM1
```

当系统创建队列管理器时，将显示以下输出:

```
C:\>crtmqm QM1
IBM MQ queue manager created.
Creating or replacing default objects for QM1.
Default objects statistics : 61 created. 0 replaced. 0 failed.
Completing setup.
Setup completed.
```

队列管理器已创建，并且已停止。 必须先启动队列管理器，然后才能对其进行管理，然后才能从其队列中读写消息。

3. 通过输入以下命令来启动队列管理器:

```
strmqm QM1
```

当队列管理器成功启动时，将显示以下输出:

```
C:\>strmqm QM1
IBM MQ queue manager 'QM1' starting.
5 log records accessed on queue manager 'QM1' during the log replay phase.
Log replay for queue manager 'QM1' complete.
Transaction manager state recovered for queue manager 'QM1'.
IBM MQ queue manager 'QM1' started.
```

队列管理器已启动。

下一步做什么

要创建队列，请参阅 第 403 页的『创建名为 LQ1 的队列』。

相关信息

在分布式平台上创建和管理队列管理器

创建名为 *LQ1* 的队列

使用命令行界面创建队列。 队列是用于存储消息的数据结构，并且是 IBM MQ 队列管理器对象。

关于此任务

有三种方法可创建 IBM MQ 对象:

- 命令行。
- MQ Explorer.
- 使用可编程接口。

在此任务中，您可以使用命令行来创建 IBM MQ 对象。

命令行界面具有名为 IBM MQ Script Commands (MQSC) 的脚本语言。 脚本编制工具 **runmqsc** 用于对队列管理器运行脚本。 要使用命令行界面创建和启动队列，请完成以下步骤。

过程

1. 通过输入以下命令来启动脚本编制工具:

```
runmqsc QM1
```

当脚本编制工具启动时，将显示以下输出:

```
C:\>runmqsc QM1
5724-H72 (C) Copyright IBM Corp. 1994, 2023. ALL RIGHTS RESERVED.
Starting MQSC for queue manager QM1.
```

该工具已准备好接受 MQSC 命令。

2. 通过输入以下 MQSC 命令来创建名为 LQ1 的本地队列:

```
define qlocal(LQ1)
```

创建队列时，将显示以下输出:

```
define qlocal(LQ1)
2 : define qlocal(LQ1)
AMQ8006: IBM MQ queue created.
```

3. 通过输入以下 MQSC 命令来停止脚本编制工具:

```
end
```

脚本编制工具结束时，将显示以下输出:

```
One MQSC command read.
No commands have a syntax error.
All valid MQSC commands were processed.

C:\>
```

下一步做什么

您已准备好将消息放入队列。 要将消息放入队列中，请参阅 第 404 页的『将消息放入队列 LQ1』。

将消息放入队列 *LQ1*

使用命令行界面将消息放入队列 LQ1 。

关于此任务

IBM MQ 随附一个名为 amqsput 的样本应用程序。 此应用程序将消息放入预定义队列。

要使用命令行界面将消息放入队列，请完成以下步骤。

过程

1. 通过输入以下命令，使用 amqsput 样本应用程序将消息放入队列 LQ1:

```
amqsput LQ1 QM1
```

样本应用程序启动时，将显示以下输出:

```
C:\>amqsput LQ1 QM1
Sample AMQSPUT0 start
target queue is LQ1
```

2. 输入 Hello World 并按 Enter 键。 您在名为 QM1 的队列管理器管理的队列 LQ1 上放置了一条包含文本 "Hello World" 的消息。
3. 要结束 amqsput ，请按 Enter 键。 将显示以下输出:

```
C:\>amqsput LQ1 QM1
Sample AMQSPUT0 start
```

```
target queue is LQ1
Hello World

Sample AMQSPUT0 end
```

下一步做什么

要从队列获取消息，请参阅 第 405 页的『从队列 LQ1 获取消息』。

从队列 *LQ1* 获取消息

使用命令行界面从队列 LQ1 获取消息。

关于此任务

IBM MQ 随附一个名为 amqsget 的样本应用程序。 此应用程序从队列中读取消息。

要使用命令行界面从队列获取消息，请完成以下步骤。

过程

通过输入以下命令，使用 amqsget 样本应用程序来读取队列 LQ1 上的消息:

```
amqsget LQ1 QM1
```

样本应用程序启动时，将显示以下输出:

```
C:\>amqsget LQ1 QM1
Sample AMQSGET0 start
message <Hello World>
no more messages
Sample AMQSGET0 end
```

amqsget 应用程序在读取消息后 30 秒结束。

下一步做什么

遵循后续场景中的指示信息以探索进一步的 IBM MQ 功能。

要了解有关编写排队应用程序，连接到队列管理器以及与队列管理器断开连接，发布/预订以及打开和关闭对象的信息，请参阅 编写用于排队的过程应用程序。

卸载 IBM MQ

停止，然后卸载 IBM MQ，包括除去任何队列管理器及其对象。 在此任务结束时，您已准备好重新安装 IBM MQ。

关于此任务

此任务描述了使用安装介质在 Windows 7 操作系统上卸载 IBM MQ 的步骤。

"入门" 方案将指导您使用启动板或命令行来完成安装 IBM MQ 的选项。 虽然可以有多个 IBM MQ 安装，但此方案基于单个服务器上的新安装。 因此，如果要重复此场景，或者尝试其他安装方法，那么必须首先卸载现有 IBM MQ 组件 (包括任何现有队列管理器及其对象) ，以便可以重新开始执行全新安装。

您可能还需要卸载，以便可以对本部分中的某些其他方案执行全新安装。

过程

1. 停止 IBM MQ 服务。
   a) 右键单击系统托盘中的 **Websphere MQ** 图标，然后单击 **停止 Websphere MQ** 以停止 IBM MQ 服务。
   将显示包含以下消息的对话框:

```
关闭 WebSphere MQ 安装 "Installation1" 将终止所有正在运行的队列管理器和
该安装的 WebSphere MQ 进程 (Microsoft Failover Cluster 控制下的进程除外)。
确定要继续吗？
```

b) 单击 **是**，然后等待 IBM MQ 停止。

c) 当 IBM MQ 停止时，右键单击系统托盘中的 **Websphere MQ** 图标，然后单击 **退出**

2. 通过以下两种方法之一开始卸载过程:

a) 在 Windows 资源管理器中，使用安装映像浏览到临时文件夹，然后双击 setup.exe。

b) 将 IBM MQ for Windows Server DVD 插入 DVD 驱动器。 如果启用了自动运行，那么安装过程将启动。 否则，双击 DVD 的根文件夹中的 "设置" 图标以启动卸载过程。

此时将打开 " IBM MQ **安装启动板** " 窗口。

3. 除去 IBM MQ。

a) 单击 **IBM MQ 安装**。

b) 单击 **启动 IBM MQ 安装程序**，然后单击 **下一步**，直到显示带有欢迎消息的 IBM MQ **程序维护窗格**。

如果未显示此窗格，那么当前未安装 IBM MQ for Windows 。

c) 单击 **维护或升级现有实例**。 选择 **Installation1** 以将其除去。 单击 **下一步**，然后在 " **程序维护**" 窗格中单击 **除去**，然后单击 **下一步**。

此时将显示 "除去服务器功能部件" 窗格。

d) 选择 **除去**: 除去现有队列管理器及其对象。

单击**下一步**。

将显示 "除去 IBM MQ " 窗格，其中包含要除去的安装的摘要。

e) 单击 **除去** 以继续。

如果显示一条消息指出找到了锁定的文件，请确保没有 IBM MQ 程序正在运行; 请参阅 在 Windows 系统上卸载 IBM MQ。

卸载 IBM MQ 时，将显示一条指示完成的消息。

f) 单击**完成**。

您已成功卸载 IBM MQ。

相关信息

在 Windows 系统上卸载 IBM MQ

后续操作

完成 " IBM MQ 入门" 方案后的下一步操作。

您可以在 IBM MQ 产品文档中查看其他主题。 您可能想要查看以下部分:

- 管理 IBM MQ

  IBM MQ 提供了您可以使用的控制命令。 在此场景中使用以下两个命令: **crtmqm** 和 **strmqm**。 此部分还提供了有关消息排队的良好概述。

- MQSC 引用

  在此场景中，您使用 `define qlocal('LQ1')` 命令来定义名为 LQ1 的本地队列; 此命令是 MQSC 命令。 IBM MQ 系统管理员使用这些命令来管理其队列管理器。 本部分介绍了这些命令，并说明了在详细描述这些命令之前如何按字母顺序使用这些命令。

- 配置队列管理器集群

  本节描述如何在称为集群的虚拟组中组织，使用和管理队列管理器。 集群可确保集群中的每个队列管理器都知道同一集群中的所有其他队列管理器。 集群还使复杂队列管理器网络的管理更简单。

产品连接方案和模式产品文档 提供了指导您完成在各种方案中将 WebSphere Application Server 连接到 IBM MQ 所需的关键任务的信息。 每个方案都包含在业务环境中实现解决方案的指示信息，使您能够随时随地学习，而无需使用其他信息资源。

点到点方案

在点到点拓扑中连接两个 IBM MQ 队列管理器以启用分布式排队。

关于此任务

创建两个队列管理器以及相应的队列和通道，以创建单向点到点消息传递基础结构。 在单独的主机上创建队列管理器以允许通过网络进行通信。 作为方案的扩展，将传输层安全性添加到通道以实现数据的安全通信。

规划解决方案

点到点消息传递是 IBM MQ 中最简单的消息传递形式。 在点到点消息传递中，发送应用程序必须先知道有关接收应用程序的某些信息，然后才能发送消息。 发送应用程序将需要一种方法来寻址远程队列。 使用点到点消息传递将消息发送到具有样本应用程序的远程队列管理器。

概述: 交付的逻辑拓扑

完成方案后交付的逻辑拓扑。

点到点基础结构允许在不同主机上的队列管理器之间进行一次定向消息传递。 队列管理器 1 (在主机 1 上) 将消息发送到队列管理器 2 (在主机 2 上)。 此场景完成后，交付的拓扑将类似于 图 1。

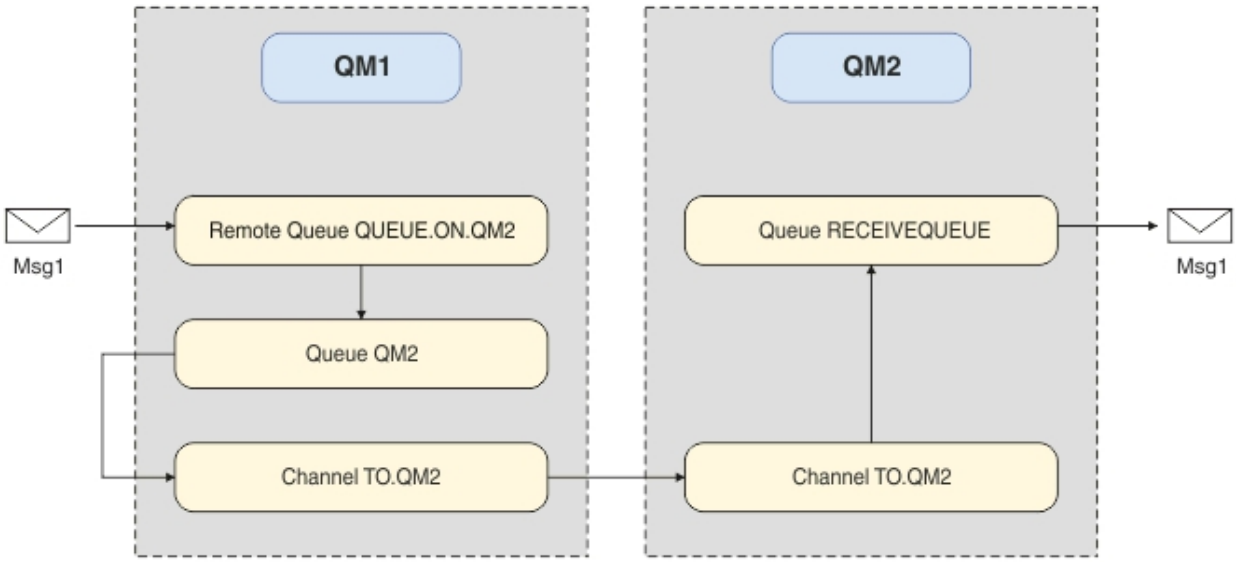


图 *82: QM1* 向 *QM2* 发送消息

基本概念和主要术语

完成点到点方案必须知道的基本概念和关键术语的描述。

基本概念

通过 IBM MQ，应用程序可以针对队列读写消息。 读消息的应用程序和写消息的应用程序是相互独立的。 不要求两个应用程序同时运行。 如果没有可用的读消息应用程序，那么消息将在 IBM MQ 队列上排队，直到应用程序读取该消息。

关键术语

下面是关于消息排队的关键术语列表。

有关消息排队的关键术语。

术语	描述
队列管理器	队列管理器负责维护它所拥有的队列，并负责将它接收到的所有消息存储到适当的队列中。
消息	消息是对使用它的应用程序有意义的以字节为单位的字符串。 消息用来将信息从一个应用程序传输到另一个应用程序。 应用程序可以在相同或不同的计算机上运行。
本地队列	本地队列是用于存储消息的数据结构。 该队列可以是常规队列或者传输队列。 直接从队列管理器读取消息的应用程序要读取的消息保留在常规队列中。 正在转移给其他队列管理器的消息保留在传输队列中。
远程队列	远程队列用于将消息发送到另一个队列管理器。
通道	通道用于在队列管理器之间发送和接收消息。
侦听器	侦听器是接受来自其他队列管理器或客户机应用程序的网络请求并启动关联通道的进程。

实现解决方案

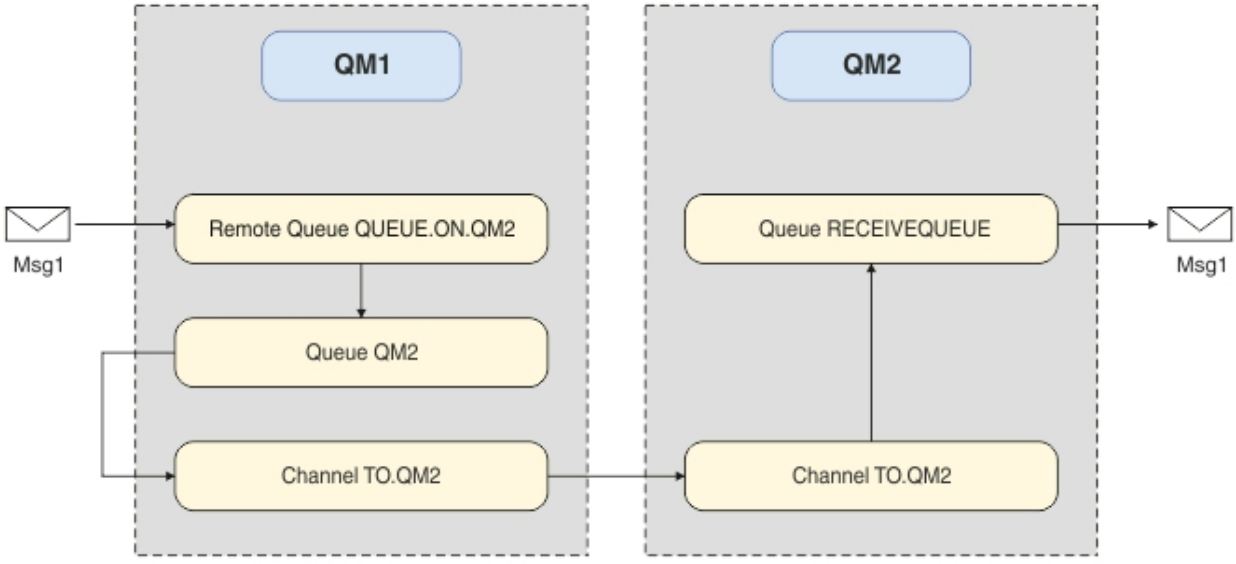
实现方案的解决方案。 在两个不同的主机上创建两个 IBM MQ 队列管理器，源队列管理器用于发送消息，目标队列管理器用于接收消息。

开始之前

此方案的起点是已验证的现有 IBM MQ 安装。 有关安装 IBM MQ 的指示信息，请遵循 安装 IBM MQ 服务器 中的步骤。

关于此任务

使用命令行界面创建两个队列管理器，定义所需的侦听器，队列和通道。 交付的逻辑拓扑显示通过实现解决方案添加的函数。



创建队列管理器

创建 IBM MQ 队列管理器以将消息发送到目标队列管理器。

开始之前

- 必须已安装 IBM MQ 。 有关安装 IBM MQ 的更多信息，请参阅 安装和卸载。

关于此任务
使用命令行界面创建 IBM MQ 队列管理器。

过程
1. 创建名为 QM1 的队列管理器。 在命令行上，输入:

```
crtmqm QM1
```

   将显示以下消息以确认已创建队列管理器:

```
IBM MQ queue manager created.
Creating or replacing default objects for QM1.
Default objects statistics : 61 created. 0 replaced. 0 failed.
Completing setup.
Setup completed.
```

2. 启动队列管理器。 在命令行上，输入:

```
strmqm QM1
```

   将显示以下消息以确认队列管理器已启动:

```
IBM MQ queue manager 'QM1' starting.
5 log records accessed on queue manager 'QM1' during the log replay phase.
Log replay for queue manager 'QM1' complete.
Transaction manager state recovered for queue manager 'QM1'.
IBM MQ queue manager 'QM1' started.
```

结果
将创建并启动 IBM MQ 队列管理器 QM1 。

下一步做什么
要创建要与 QM1 配合使用的队列，请遵循 第 409 页的『创建请求』中的指示信息。

创建请求

创建由 IBM MQ 队列管理器管理的 IBM MQ 队列。

开始之前
您必须具有如 第 408 页的『创建队列管理器』中所述设置的 IBM MQ 队列管理器。

关于此任务
启动 **MQSC** 接口以管理连接到队列管理器的对象。 创建传输队列和远程队列定义。 退出 **MQSC** 接口。

过程
1. 在命令行上，输入:

```
runmqsc QM1
```

   在确认消息后，该工具已准备好接受命令。
2. 创建名为 QM2 的传输队列。 最好为传输队列提供与远程队列管理器相同的名称。 在 MQSC 接口中，输入:

```
DEFINE QLOCAL(QM2) DESCR('Transmission queue to QM2') USAGE(XMITQ)
```

将创建传输队列。

3. 创建名为 QUEUE.ON.QM2。 远程队列定义必须引用为远程主机上的本地队列提供的名称。 在 MQSC 接口中，输入:

```
DEFINE QREMOTE(QUEUE.ON.QM2) DESCR('Remote queue for QM2') XMITQ(QM2) RNAME(RECEIVEQUEUE)
RQMNAME(QM2)
```

将创建远程队列定义。

4. 输入 end 以退出 MQSC 接口。

下一步做什么

要创建用于连接到目标队列管理器的发送方通道，请遵循 第 410 页的『创建发送方通道』中的指示信息。

创建发送方通道

在源队列管理器上创建发送方通道，该通道用于连接到目标队列管理器。

开始之前

要创建使用 TLS 的通道，请遵循 第 415 页的『创建通道以使用 TLS』中的指示信息。 如果要在没有 TLS 安全性的情况下测试解决方案，那么可以在之后执行此操作。

关于此任务

启动 **MQSC** 接口以管理连接到队列管理器的对象并创建发送方通道。 此通道用于连接到名为 QM2 的目标队列管理器。

过程

1. 在命令行上，输入:

```
runmqsc QM1
```

在确认消息后，该工具已准备好接受命令。

2. 创建名为 TO.QM2。 在 MQSC 接口中，输入:

```
DEFINE CHANNEL(TO.QM2) CHLTYPE(SDR) CONNAME(' remoteHost ') TRPTYPE(TCP) XMITQ(QM2)
```

注: 变量 *remoteHost* 是目标队列管理器的主机名或 IP 地址。

将创建发送方通道。

下一步做什么

要创建分布式队列管理器拓扑，请遵循 第 410 页的『创建分布式队列管理器拓扑』中的指示信息。

创建分布式队列管理器拓扑

点到点消息传递是 IBM MQ 中最简单的消息传递形式。 在点到点消息传递中，发送应用程序必须先知道有关接收应用程序的某些信息，然后才能发送消息。 发送应用程序将需要一种方法来寻址远程队列。 使用点到点消息传递将消息发送到具有样本应用程序的第二个队列管理器。

开始之前

您必须已设置源队列管理器，如 第 408 页的『创建队列管理器』中所述。

关于此任务

在远程主机上创建目标队列管理器。 使用样本应用程序来验证源队列管理器与目标队列管理器之间的通信。

创建队列管理器
创建 IBM MQ 队列管理器以从远程队列管理器接收消息。

开始之前
必须已安装 IBM MQ 。 有关安装 IBM MQ 的更多信息，请参阅 安装 IBM MQ 服务器。

关于此任务
使用命令行界面创建 IBM MQ 队列管理器。

过程
1. 创建名为 QM2 的队列管理器。 在命令行上，输入:

```
crtmqm QM2
```

   此时会显示以下消息:

```
IBM MQ queue manager created.
Creating or replacing default objects for QM2.
Default objects statistics : 61 created. 0 replaced. 0 failed.
Completing setup.
Setup completed.
```

2. 启动队列管理器。 在命令行上，输入:

```
strmqm QM2
```

   将显示以下消息以确认队列管理器已启动:

```
IBM MQ queue manager 'QM2' starting.
5 log records accessed on queue manager 'QM2' during the log replay phase.
Log replay for queue manager 'QM2' complete.
Transaction manager state recovered for queue manager 'QM2'.
IBM MQ queue manager 'QM2' started.
```

结果
将创建并启动 IBM MQ 队列管理器 QM2 。

下一步做什么
要创建要与 QM2 配合使用的队列，请遵循 第 411 页的『创建队列』中的指示信息。

创建队列
创建用于在目标队列管理器上接收消息的本地队列，以及用于接受入站通道连接的侦听器。

关于此任务
启动 **runmqsc** 脚本编制工具后，可以使用 MQSC 命令来创建本地队列和侦听器。

过程
1. 通过输入以下命令来启动脚本编制工具:

```
runmqsc QM2
```

   将显示一条消息以确认工具已启动。
2. 创建名为 RECEIVEQUEUE 的本地队列。 队列必须与源队列管理器上的远程队列定义中引用的名称相同。 在 MQSC 接口中，输入:

```
DEFINE QLOCAL(RECEIVEQUEUE) DESCR('Receiving queue')
```

已创建本地队列。

3. 创建名为 LISTENER1 的侦听器。 在 MQSC 接口中，输入:

```
DEFINE LISTENER(LISTENER1) TRPTYPE(TCP) PORT(1414) CONTROL(QMGR)
```

注: 端口 1414 是 IBM MQ 的缺省端口。 如果选择其他端口号，那么必须将其添加到发送队列管理器上发送方通道的 CONNAME 。

4. 启动侦听器，以便它可以接受入站连接。 在 MQSC 接口中，输入:

```
START LISTENER(LISTENER1)
```

注: 由于侦听器是使用选项 CONTROL （QMGR)创建的，因此下次启动队列管理器时，该侦听器也将自动启动。

5. 输入 end 以退出 **MQSC** 接口。

下一步做什么

要创建接收方通道以在源队列管理器与目标队列管理器之间创建连接，请遵循 第 412 页的『创建接收方通道』 中的指示信息。

创建接收方通道

为目标队列管理器创建接收方通道，以启用源队列管理器与目标队列管理器之间的通信。

开始之前

要创建使用 TLS 的通道，请遵循 第 415 页的『创建通道以使用 TLS』 中的指示信息。 如果要在没有 TLS 安全性的情况下测试解决方案，那么可以在之后执行此操作。

关于此任务

使用 **MQSC** 接口来创建由 QM2 管理的接收方通道。

过程

1. 在命令行上，输入:

```
runmqsc QM2
```

在确认消息后，该工具已准备好接受命令。

2. 创建名为 TO.QM2。 该通道必须与源队列管理器上的发送方通道同名。 在 MQSC 接口中，输入:

```
DEFINE CHANNEL(TO.QM2) CHLTYPE(RCVR) TRPTYPE(TCP)
```

已创建接收方通道。

下一步做什么

要在源队列管理器上启动发送方通道，进而在目标队列管理器上启动接收方通道，请执行 第 412 页的『启动发送方通道』 中的指示信息。

启动发送方通道

在源队列管理器上启动发送方通道，在目标队列管理器上启动接收方通道。 可以将消息从源队列管理器发送到目标队列管理器。

关于此任务
启动 **MQSC** 接口以管理连接到队列管理器的对象。 启动发送方通道以连接到目标队列管理器，从而启用通信。 当启动源通道时，接收方通道会自动启动。

过程
1. 在命令行上，输入:

```
runmqsc QM1
```

   在确认消息后，该工具已准备好接受命令。
2. 在源队列管理器上启动发送方通道。 在 MQSC 接口中，输入:

```
START CHANNEL(TO.QM2)
```

   发送方通道启动，目标队列管理器上的接收方通道也启动。
3. 请检查通道是否正在运行。 在 MQSC 接口中，输入:

```
DISPLAY CHSTATUS(TO.QM2)
```

   如果通道正在运行，那么您将看到它报告 STATUS （RUNNING）。 如果它在 STATUS 中报告任何其他值，请检查 错误日志。

下一步做什么
要验证源队列管理器是否可以向目标队列管理器发送消息，请遵循 第 413 页的『验证解决方案』中的指示信息。

验证解决方案
验证源队列管理器是否可以将消息放入远程队列。 验证目标队列管理器是否可以从队列中获取消息。

关于此任务
使用样本应用程序 **amqsput** 和 **amqsget** 来验证解决方案。

过程
1. 从源队列管理器向目标队列管理器 QM2 发送消息。
   a) 在命令行界面中，输入:

```
amqsput QUEUE.ON.QM2 QM1
```

   必须使用远程队列定义的名称将消息发送到目标队列管理器。
   此时会显示以下消息:

```
Sample AMQSPUT0 start
target queue is QUEUE.ON.QM2
```

   b) 输入 Hello world.，按 Enter 键两次。
2. 获取目标队列管理器上的消息。
   a) 在命令行界面中，输入:

```
amqsget RECEIVEQUEUE QM2
```

   此时会显示以下消息:

```
Sample AMQSGET0 start
message <Hello world.>
no more messages
Sample AMQSGET0 end
```

结果
目标队列管理器从源队列管理器接收消息，并验证是否已实现点到点通信。

下一步做什么
如果要向解决方案添加安全性，请遵循 第 414 页的『保护点到点拓扑』中的指示信息。

保护点到点拓扑
保护点到点拓扑，以便可以在生产环境中传输消息。

关于此任务
保护源和目标队列管理器对象，以便授予正确的访问级别。 定义哪些用户组有权访问队列和队列管理器。 使用数字签名证书通过传输层安全性 (TLS) 进行连接来保护网络连接。

保护源队列管理器对象
设置源队列管理器上对象的授权值。

关于此任务
使用 **setmqaut** 命令向运行应用程序的用户组授予权限。

过程
1. 要将具有 *connect* 权限的指定用户组授予队列管理器，请在命令行界面上输入:

```
setmqaut -m QM1 -t qmgr -g userGroup +connect
```

2. 要在远程队列定义上授予具有 *put* 权限的指定用户组，请在命令行界面上输入:

```
setmqaut -m QM1 -t q -n "QUEUE.ON.QM2" -g userGroup +put
```

保护目标队列管理器对象
设置目标队列管理器上对象的授权值。

关于此任务
使用 **setmqaut** 命令向运行应用程序的用户组授予权限。

过程
1. 要将具有 *connect* 权限的指定用户组授予队列管理器，请在命令行界面上输入:

```
setmqaut -m QM2 -t qmgr -g userGroup +connect
```

2. 要在远程队列定义上授予具有 *get* 权限的指定用户组，请在命令行界面中输入:

```
setmqaut -m QM2 -t q -n "RECEIVEQUEUE" -g userGroup +get
```

保护网络

保护源队列管理器与远程队列管理器之间的网络连接。

关于此任务

使用签名证书来验证源队列管理器和远程队列管理器的真实性。 使用 SSL 或 TLS 网络传输消息以加密消息。

准备队列管理器以使用 *TLS*

IBM MQ 队列管理器的密钥存储库用于存储队列管理器的个人证书和公共认证中心 (CA) 证书。 来自 IBM MQ 队列管理器的个人证书请求必须由 CA 签署，公用证书由其他实体用于认证 IBM MQ 队列管理器。

开始之前

您必须在文件中具有公用认证中心证书。

关于此任务

创建 IBM MQ 队列管理器的密钥存储库，导入认证中心的签署者证书并创建队列管理器的个人证书请求。

过程

1. 为名为 `key.kdb` 的队列管理器创建 CMS 密钥存储库文件。 浏览至 `Qmgrs\QM1\ssl` 目录，并在命令行上输入:

   ```
   runmqckm -keydb -create -db key.kdb -pw passw0rd -type cms -stash
   ```

   注: 对于此简单示例，我们使用了密码 `passw0rd`。 您可能希望选择不同的密码，并更改以下每个命令以改为使用您自己的密码。
2. 在命令行上将文件中包含的 CA 证书添加到密钥存储库中，请输入:

   ```
   runmqckm -cert -add -file CA-certificate-file -db key.kdb -pw passw0rd -label TrustedCA
   ```
3. 请求将写入名为 `QM1req.req` 的请求文件的个人证书。

   在命令行上，输入:

   ```
   runmqckm -certreq -create -db key.kdb -pw passw0rd -label ibmwebspheremqqm1
   -dn CN="QM1" -size 1024 -file QM1req.req
   -sig_alg SHA1WithRSA
   ```

   此示例中显示了缺省证书标签名称。 如果您愿意，可以设置自己的名称。 有关详细信息，请参阅 数字证书标签。
4. 将证书请求文件发送到 CA ，他们将发放数字签名证书。 将接收到的已签名证书文件放在要接收到队列管理器的密钥存储库中的合适位置。
5. 将签名的个人证书接收到队列管理器的密钥存储库中。

   ```
   runmqckm -cert -receive -file Signed-certificate-file -db key.kdb -pw passw0rd -format ascii
   ```
6. 对每个队列管理器完成这些步骤，并相应地更改队列管理器名称。

下一步做什么

要通过发送方和接收方通道启用安全通信，请遵循 第 415 页的『创建通道以使用 TLS』中的指示信息。

创建通道以使用 *TLS*

创建使用 TLS 创建连接的新通道。

开始之前

要通过使用 TLS 的通道进行通信，首先必须具有连接的每一端所需的证书。 要创建所需证书，请遵循 第 415 页的『准备队列管理器以使用 TLS』中的指示信息。

关于此任务

使用 MQSC 接口定义设置了 SSL/TLS 属性的通道。 即使您在先前步骤中通过使用 REPLACE 关键字定义了没有 SSL/TLS 的通道，也可以执行此任务。

过程

1. 在命令行上，输入:

```
runmqsc QM1
```

2. 在 QM1 上的 MQSC 接口中创建名为 TO.QM2 的发送方通道，请输入:

```
DEFINE CHANNEL(TO.QM2) CHLTYPE(SDR) TRPTYPE(TCP)
CONNAME('remoteHost') XMITQ(QM2)
SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA256)
DESCR('Sender channel using TLS from QM1 to QM2')
REPLACE
```

   注: 变量 *remoteHost* 是目标队列管理器的主机名或 IP 地址。

   可以为通道指定 CERTLABL 属性。 如果执行此操作，那么它必须与先前在 第 415 页的『准备队列管理器以使用 TLS』的步骤 3 中运行的 **runmqckm** 命令的 **-label** 参数上的值相匹配。 有关证书标签的更多信息，请参阅 数字证书标签，了解需求。

3. 输入 end 以退出 MQSC 接口。
4. 在命令行上，输入:

```
runmqsc QM2
```

5. 在 QM2 上创建名为 TO.QM2，在 MQSC 接口中，输入:

```
DEFINE CHANNEL(TO.QM2) CHLTYPE(RCVR) TRPTYPE(TCP)
SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA256) SSLCAUTH(REQUIRED)
DESCR('Receiver channel using TLS from QM1 to QM2')
REPLACE
```

6. 输入 end 以退出 MQSC 接口。

下一步做什么

要验证源队列管理器是否可以使用 TLS 将消息发送到目标队列管理器，请遵循 第 413 页的『验证解决方案』中的指示信息。

发布/预订方案

演示发布/预订集群和发布/预订层次结构的使用的两组方案。

以下子主题描述了可用的发布/预订方案:

发布/预订集群方案

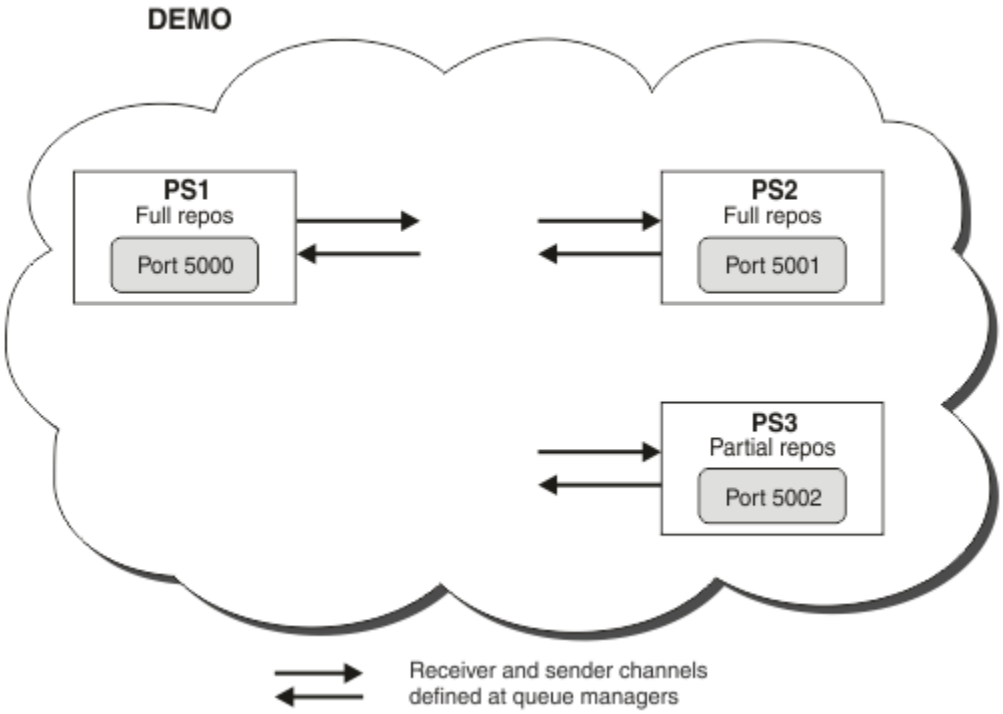
在此场景中，您将创建一个简单的三个队列管理器集群，并将其配置为允许在一个队列管理器上创建的预订接收由连接到另一个队列管理器的应用程序发布的消息。

开始之前

此方案的起点是现有 IBM MQ 安装。 有关安装 IBM MQ 的指示信息，请遵循 安装 IBM MQ 服务器中的步骤。

关于此任务

通过完成此场景中的步骤，您首先创建以下集群:



。

此集群由三个队列管理器组成，其中两个队列管理器定义为完整存储库队列管理器。

然后在队列管理器 PS3 上定义集群主题。 通过创建集群主题，您已将集群设置为发布/预订集群。 要测试发布/预订集群，请在任何队列管理器上预订该主题，然后从另一个队列管理器将消息发布到该主题，并检查您的预订是否接收到该消息。

相关信息

设计发布/预订集群
配置队列管理器集群

创建和启动队列管理器

创建并启动三个队列管理器，称为 PS1， PS2 和 PS3。

过程

1. 创建并启动队列管理器 PS1。
   a) 创建队列管理器。

      在命令行中，输入以下命令:

      ```
      crtmqm PS1
      ```

   b) 启动队列管理器。

      在命令行中，输入以下命令:

```
strmqm PS1
```

2. 重复步骤 1 以创建并启动队列管理器 PS2。
3. 重复步骤 1 以创建并启动队列管理器 PS3。

下一步做什么

现在，您已准备好 配置第一个队列管理器。

配置第一个队列管理器

使用 MQSC 接口为 PS1 定义侦听器和接收方通道，将队列管理器设置为集群的完整存储库，并定义从 PS1 到 PS2 的发送方通道，以便这两个完整存储库可以交换信息。

开始之前

此任务假定您已完成 第 417 页的『创建和启动队列管理器』中的步骤。

过程

1. 定义并启动 PS1 的侦听器。
   a) 启动 MQSC 接口。

   在命令行中，输入以下命令:

   ```
   runmqsc PS1
   ```

   b) 定义侦听器。

   输入以下 MQSC 命令:

   ```
   DEFINE LISTENER(PS1_LS) TRPTYPE(TCP) CONTROL(QMGR) PORT(5000)
   ```

   c) 启动侦听器。

   输入以下 MQSC 命令:

   ```
   START LISTENER(PS1_LS)
   ```

2. 将队列管理器设置为集群的完整存储库。

   输入以下 MQSC 命令:

   ```
   ALTER QMGR REPOS(DEMO)
   ```

3. 为 PS1 定义接收方通道，以允许集群中的其他队列管理器与其通信。

   输入以下 MQSC 命令:

   ```
   DEFINE CHANNEL(DEMO.PS1) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5000)')
   CLUSTER(DEMO)
       DESCR('TCP Cluster-receiver channel for queue manager PS1')
   ```

4. 定义从 PS1 到 PS2 的发送方通道，以允许两个完整存储库交换信息。

   输入以下 MQSC 命令:

   ```
   DEFINE CHANNEL(DEMO.PS2) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5001)')
   CLUSTER(DEMO)
       DESCR('TCP Cluster-sender channel from PS1 to queue manager PS2')
   ```

下一步做什么

现在，您已准备好 配置第二个队列管理器。

配置第二个队列管理器

使用 MQSC 接口为 PS2 定义侦听器和接收方通道，将队列管理器设置为集群的完整存储库，并定义从 PS2 到 PS1 的发送方通道，以便这两个完整存储库可以交换信息。

开始之前

此任务假定您已完成 第 418 页的『配置第一个队列管理器』中的步骤。

过程

1. 定义并启动 PS2 的侦听器。
   a) 启动 MQSC 接口。

      在命令行中，输入以下命令:

      ```
      runmqsc PS2
      ```

   b) 定义侦听器。

      输入以下 MQSC 命令:

      ```
      DEFINE LISTENER(PS2_LS) TRPTYPE(TCP) CONTROL(QMGR) PORT(5001)
      ```

   c) 启动侦听器。

      输入以下 MQSC 命令:

      ```
      START LISTENER(PS2_LS)
      ```

2. 将队列管理器设置为集群的完整存储库。

   输入以下 MQSC 命令:

   ```
   ALTER QMGR REPOS(DEMO)
   ```

3. 为 PS2 定义接收方通道，以允许集群中的其他队列管理器与其进行通信。

   输入以下 MQSC 命令:

   ```
   DEFINE CHANNEL(DEMO.PS2) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5001)')
   CLUSTER(DEMO)
       DESCR('TCP Cluster-receiver channel for queue manager PS2')
   ```

4. 定义从 PS2 到 PS1 的发送方通道，以允许两个完整存储库交换信息。

   输入以下 MQSC 命令:

   ```
   DEFINE CHANNEL(DEMO.PS1) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5000)')
   CLUSTER(DEMO)
       DESCR('TCP Cluster-sender channel from PS2 to PS1')
   ```

下一步做什么

现在，您已准备好 配置第三个队列管理器。

配置第三个队列管理器

使用 MQSC 接口为 PS3 定义侦听器和接收方通道。 通过定义从 PS3 到其中一个完整存储库队列管理器的发送方通道，将 PS3 连接到集群中。

开始之前

此任务假定您已完成 第 419 页的『配置第二个队列管理器』中的步骤。

过程

1. 定义并启动 PS3 的侦听器。
   a) 启动 MQSC 接口。

      在命令行中，输入以下命令:

      ```
      runmqsc PS3
      ```

   b) 定义侦听器。

      输入以下 MQSC 命令:

      ```
      DEFINE LISTENER(PS3_LS) TRPTYPE(TCP) CONTROL(QMGR) PORT(5002)
      ```

   c) 启动侦听器。

      输入以下 MQSC 命令:

      ```
      START LISTENER(PS3_LS)
      ```

2. 为 PS3 定义接收方通道，以允许集群中的其他队列管理器与其进行通信。

   输入以下 MQSC 命令:

   ```
   DEFINE CHANNEL(DEMO.PS3) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5002)')
   CLUSTER(DEMO)
       DESCR('TCP Cluster-receiver channel for queue manager PS3')
   ```

3. 定义从 PS3 到某个完整存储库队列管理器 (例如 PS1) 的发送方通道。 这会将 PS3 连接到集群中。

   输入以下 MQSC 命令:

   ```
   DEFINE CHANNEL(DEMO.PS1) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5000)')
   CLUSTER(DEMO)
       DESCR('TCP Cluster-sender channel from PS3 to PS1')
   ```

4. 验证 PS3 是否已成功加入集群。

   输入以下 MQSC 命令:

   ```
   DISPLAY CLUSQMGR(*) QMTYPE
   ```

   此命令返回三个条目，分别用于 QM1， QM2 和 QM3。 QM1 和 QM2 应该具有 **QMTYPE** REPOS，而 QM3 应该具有 **QMTYPE** NORMAL。

下一步做什么

现在，您已准备好 定义集群主题。

定义集群主题

发布和预订应用程序可以发布到任何主题字符串，而无需定义受管主题对象。 但是，如果发布应用程序连接到与创建预订的队列管理器不同的集群队列管理器，那么必须定义受管主题对象并将其添加到集群。 要使主题成为集群主题，请在其定义中指定集群的名称。

开始之前

此任务假定您已完成 第 420 页的『配置第三个队列管理器』中的步骤。

关于此任务

受管主题对象标识主题树中通过其主题字符串进行集群的点。 发布和预订应用程序可以在该点或以下使用任何主题字符串，并且它们的消息会在队列管理器之间自动传输。

定义集群主题时，还应选择其路由模型。 有关集群中发布路由的更多信息，请参阅 设计发布/预订集群。

对于此场景，我们使用缺省路由 *DIRECT*。 这意味着将消息从发布队列管理器直接发送到预订队列管理器。

过程

1. 在 PS3 上定义集群主题 SCORES 。

   要使主题成为集群主题，请指定集群的名称，并设置集群路由 (**CLROUTE**) 要用于此主题的发布和预订。

   a) 启动 MQSC 接口。

   在命令行中，输入以下命令:

   ```
   runmqsc PS3
   ```

   b) 定义集群主题 SCORES。

   输入以下 MQSC 命令:

   ```
   DEFINE TOPIC(SCORES) TOPICSTR('/Sport/Scores') CLUSTER(DEMO) CLROUTE(DIRECT)
   ```

   c) 输入 end 以退出 PS3 的 MQSC 接口。
2. 验证 PS1 上的主题定义。

   a) 启动 PS1 的 MQSC 接口。

   在命令行中，输入以下命令:

   ```
   runmqsc PS1
   ```

   b) 显示集群主题 SCORES 的集群状态。

   输入以下 MQSC 命令:

   ```
   DISPLAY TCLUSTER(SCORES) CLSTATE
   ```

   集群主题 SCORES 的 **CLSTATE** 显示为 ACTIVE。

下一步做什么

有关此任务的更详细探索，请参阅 配置发布/预订集群。

现在，您已准备好验证解决方案。 请参阅第 421 页的『测试发布/预订集群』。

测试发布/预订集群

通过发布和预订来自集群中不同队列管理器的主题字符串来测试发布/预订集群。

开始之前
此任务假定您已完成第 421 页的『定义集群主题』中的步骤。

关于此任务
通过使用命令行以及 IBM MQ 随附的 amqspub 和 amqssub 样本应用程序，您可以从一个队列管理器发布主题，并与其他队列管理器一起预订该主题。 将消息发布到主题时，预订队列管理器将接收到该消息。

过程
1. 在命令行中，输入以下命令:

```
amqspub /Sport/Scores/Football PS1
```

2. 同时，在单独的命令行中，输入以下命令:

```
amqssub /Sport/Scores/Football PS2
```

```
amqssub /Sport/Scores/Football PS3
```

3. 在第一个命令行中，输入消息。
   消息显示在两个预订命令行中。

   注: 如果在 10 秒内未收到发布，那么 amqssub 应用程序将超时。

结果
已完成发布/预订集群设置。

下一步做什么
尝试为主题树的不同分支定义不同的主题对象，并使用不同的路由模型。

发布/预订层次结构方案
演示发布/预订层次结构的使用的三种方案。 这三个方案都设置了相同的简单发布/预订拓扑。 在每个方案中，队列管理器都依赖于不同的方法来连接到层次结构中的相邻队列管理器。

以下子主题中描述了可用的发布/预订层次结构方案:

相关信息
发布/预订层次结构

发布/预订层次结构方案 1: 使用具有队列管理器名称别名的点到点通道
这是以不同方式设置发布/预订层次结构以在队列管理器之间建立连接的三个方案中的第一个方案。 此方案设置发布/预订层次结构，该层次结构使用具有队列管理器名称别名的点到点通道。

关于此任务
这组方案都使用名为 QM1 的父队列管理器以及名为 QM2 和 QM3 的两个子队列管理器。

方案 1 将拆分为更小的部分，以使过程更易于遵循。

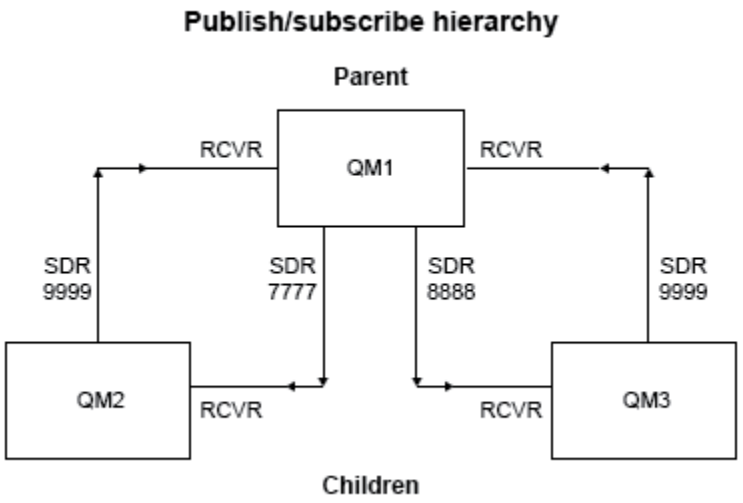


图 *83:* 此拓扑图显示典型发布者/预订层次结构中的队列管理器之间的关系。

方案 *1 part 1:* 创建队列管理器

过程

1. 使用以下命令创建并启动三个名为 QM1， QM2 和 QM3 的队列管理器:

```
crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM1
strmqm QM1

crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM2
strmqm QM2

crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM3
strmqm QM3
```

2. 通过在所有三个队列管理器上使用以下命令来启用队列管理器发布/预订方式:

```
ALTER QMGR PSMODE(ENABLED)
```

方案 *1 part 2:* 点到点通道连接

关于此任务

使用与父队列管理器同名的队列管理器别名在队列管理器之间建立点到点通道连接。

过程

1. 在 QM2 到 QM1 上定义传输队列和队列管理器别名。 为 QM1 定义发送方通道，为 QM2 定义在 QM1 上创建的发送方通道的接收方通道:

```
DEFINE QLOCAL(QM1.XMITQ) USAGE(XMITQ)

DEFINE QREMOTE (QM1) RNAME('') RQMNAME(QM1) XMITQ(QM1.XMITQ)

DEFINE CHANNEL('QM2.TO.QM1') CHLTYPE(SDR) CONNAME('localhost(9999)') XMITQ(QM1.XMITQ)
TRPTYPE(TCP)

DEFINE CHANNEL('QM1.TO.QM2') CHLTYPE(RCVR) TRPTYPE(TCP)
```

2. 在 QM3 到 QM1 上定义传输队列和队列管理器别名。 定义 QM1 的发送方通道以及在 QM1 上为 QM3 创建的发送方通道的接收方通道:

```
DEFINE QLOCAL(QM1.XMITQ) USAGE(XMITQ)

DEFINE QREMOTE (QM1) RNAME('') RQMNAME(QM1) XMITQ(QM1.XMITQ)

DEFINE CHANNEL('QM3.TO.QM1') CHLTYPE(SDR) CONNAME('localhost(9999)') XMITQ(QM1.XMITQ)
TRPTYPE(TCP)

DEFINE CHANNEL('QM1.TO.QM3') CHLTYPE(RCVR) TRPTYPE(TCP)
```

3. 在 QM1 到 QM2 和 QM3 上定义传输队列和队列管理器别名。 定义到 QM2 和 QM3 的发送方通道，以及在 QM2 和 QM3 上为 QM1 创建的发送方通道的接收方通道:

```
DEFINE QLOCAL(QM2.XMITQ) USAGE(XMITQ)

DEFINE QREMOTE (QM2) RNAME('') RQMNAME(QM2) XMITQ(QM2.XMITQ)

DEFINE CHANNEL('QM1.TO.QM2') CHLTYPE(SDR) CONNAME('localhost(7777)') XMITQ(QM2.XMITQ)
TRPTYPE(TCP)

DEFINE CHANNEL('QM2.TO.QM1') CHLTYPE(RCVR) TRPTYPE(TCP)

DEFINE QLOCAL(QM3.XMITQ) USAGE(XMITQ)

DEFINE QREMOTE (QM3) RNAME('') RQMNAME(QM3) XMITQ(QM3.XMITQ)

DEFINE CHANNEL('QM1.TO.QM3') CHLTYPE(SDR) CONNAME('localhost(8888)') XMITQ(QM3.XMITQ)
TRPTYPE(TCP)

DEFINE CHANNEL('QM3.TO.QM1') CHLTYPE(RCVR) TRPTYPE(TCP)
```

4. 在队列管理器上启动相应的侦听器:

```
runmqlsr -m QM1 -t TCP -p 9999 &
runmqlsr -m QM2 -t TCP -p 7777 &
runmqlsr -m QM3 -t TCP -p 8888 &
```

5. 启动以下通道:
   a. 在 QM1 上:

```
START CHANNEL('QM1.TO.QM2')

START CHANNEL('QM1.TO.QM3')
```

   b. 在 QM2 上:

```
START CHANNEL('QM2.TO.QM1')
```

   c. 在 QM3 上:

```
START CHANNEL('QM3.TO.QM1')
```

6. 检查所有通道是否已启动:

```
DISPLAY CHSTATUS('QM1.TO.QM2')

DISPLAY CHSTATUS('QM1.TO.QM3')

DISPLAY CHSTATUS('QM2.TO.QM1')

DISPLAY CHSTATUS('QM3.TO.QM1')
```

方案 *1* 部件 *3:* 连接队列管理器并定义主题

关于此任务

将子队列管理器 QM2 和 QM3 连接到父队列管理器 QM1。

过程

1. 在 QM2 和 QM3 上，将父队列管理器设置为 QM1:

```
ALTER QMGR PARENT (QM1)
```

2. 在所有队列管理器上运行以下命令以检查子队列管理器是否已连接到父队列管理器:

```
DISPLAY PUBSUB TYPE(ALL)
```

将显示命令输出。 例如，以下是 QM1 的输出，其中突出显示了关键详细信息:

```
DISPLAY PUBSUB ALL
1 : DISPLAY PUBSUB ALL
AMQ8723: Display pub/sub status details.
QMNAME(QM1)                TYPE(LOCAL)
STATUS(ACTIVE)              SUBCOUNT(6)
TPCOUNT(9)
AMQ8723: Display pub/sub status details.
QMNAME(QM2) TYPE(CHILD)
STATUS(ACTIVE) SUBCOUNT(NONE)
TPCOUNT(NONE)
AMQ8723: Display pub/sub status details.
QMNAME(QM3) TYPE(CHILD)
STATUS(ACTIVE) SUBCOUNT(NONE)
TPCOUNT(NONE)
```

方案 *1 part 4:* 发布和预订主题

关于此任务

使用 amqspub.exe 和 amqssub.exe 应用程序来发布和预订主题。

过程

1. 在第一个命令窗口中运行以下命令:

```
amqspub Sport/Soccer QM2
```

2. 在第二个命令窗口中运行此命令:

```
amqssub Sport/Soccer QM1
```

3. 在第三个命令窗口中运行以下命令:

```
amqssub Sport/Soccer QM3
```

结果

第二个和第三个命令窗口中的 amqssub.exe 应用程序接收在第一个命令窗口中发布的消息。

相关任务

第 426 页的『发布/预订层次结构方案 2: 将同名的点到点通道用于传输队列和远程队列管理器』

这是三种场景中的第二种场景，它们以不同方式设置发布/预订层次结构，以建立队列管理器之间的连接。此方案设置一个发布/预订层次结构，该层次结构使用与远程队列管理器具有相同传输队列名称的点到点通道。

第 429 页的『发布/预订层次结构方案 3: 使用集群通道添加队列管理器』
这是三种场景中的第三种场景，它们以不同方式设置发布/预订层次结构以建立队列管理器之间的连接。 此方案使用集群通道将队列管理器添加到层次结构。

相关信息
将队列管理器连接到发布/预订层次结构

发布/预订层次结构方案 2: 将同名的点到点通道用于传输队列和远程队列管理器

这是三种场景中的第二种场景，它们以不同方式设置发布/预订层次结构，以建立队列管理器之间的连接。此方案设置一个发布/预订层次结构，该层次结构使用与远程队列管理器具有相同传输队列名称的点到点通道。

关于此任务

这组方案都使用名为 QM1 的父队列管理器以及名为 QM2 和 QM3 的两个子队列管理器。

方案 2 将拆分为更小的部分，以使过程更易于遵循。 此方案复用 第 422 页的『发布/预订层次结构方案 1: 使用具有队列管理器名称别名的点到点通道』中的方案 1 部件 1 ，方案 1 部件 3 和方案 1 部件 4。

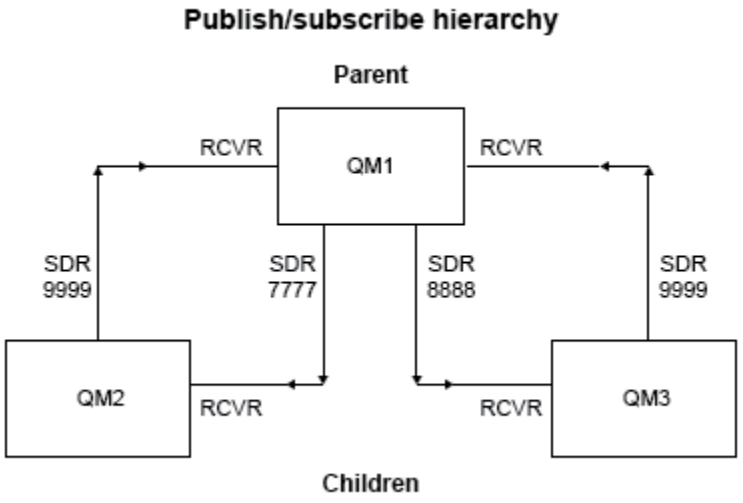


图 *84:* 此拓扑图显示典型发布者/预订层次结构中的队列管理器之间的关系。

方案 *2 part 1:* 创建队列管理器并设置 *PSMODE*

过程

1. 使用以下命令创建并启动三个名为 QM1， QM2 和 QM3 的队列管理器:

```
crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM1
strmqm QM1

crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM2
strmqm QM2

crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM3
strmqm QM3
```

2. 通过在所有三个队列管理器上使用以下命令来启用队列管理器发布/预订方式:

```
ALTER QMGR PSMODE(ENABLED)
```

方案 *2 part 2:* 点到点通道连接

关于此任务

使用与父队列管理器同名的传输队列在队列管理器之间建立点到点通道连接。

过程

1. 在 QM2 到 QM1 上定义传输队列。 为 QM1 上创建的 QM2 的发送方通道定义到 QM1 的发送方通道和接收方通道:

```
DEFINE QLOCAL(QM1) USAGE(XMITQ)
DEFINE CHANNEL('QM2.TO.QM1') CHLTYPE(SDR) CONNAME('localhost(9999)') XMITQ(QM1) TRPTYPE(TCP)
DEFINE CHANNEL('QM1.TO.QM2') CHLTYPE(RCVR) TRPTYPE(TCP)
```

2. 在 QM3 到 QM1 上定义传输队列。 定义 QM1 的发送方通道以及在 QM1 上为 QM3 创建的发送方通道的接收方通道:

```
DEFINE QLOCAL(QM1) USAGE(XMITQ)
DEFINE CHANNEL('QM3.TO.QM1') CHLTYPE(SDR) CONNAME('localhost(9999)') XMITQ(QM1) TRPTYPE(TCP)
DEFINE CHANNEL('QM1.TO.QM3') CHLTYPE(RCVR) TRPTYPE(TCP)
```

3. 在 QM1 到 QM2 和 QM3 上定义传输队列。 定义到 QM2 和 QM3 的发送方通道， 以及在 QM2 和 QM3 上为 QM1 创建的发送方通道的接收方通道:

```
DEFINE QLOCAL(QM2) USAGE(XMITQ)
DEFINE CHANNEL('QM1.TO.QM2') CHLTYPE(SDR) CONNAME('localhost(7777)') XMITQ(QM2) TRPTYPE(TCP)
DEFINE CHANNEL('QM2.TO.QM1') CHLTYPE(RCVR) TRPTYPE(TCP)
DEFINE QLOCAL(QM3) USAGE(XMITQ)
DEFINE CHANNEL('QM1.TO.QM3') CHLTYPE(SDR) CONNAME('localhost(8888)') XMITQ(QM3) TRPTYPE(TCP)
DEFINE CHANNEL('QM3.TO.QM1') CHLTYPE(RCVR) TRPTYPE(TCP)
```

4. 在队列管理器上启动相应的侦听器:

```
runmqlsr -m QM1 -t TCP -p 9999 &
runmqlsr -m QM2 -t TCP -p 7777 &
runmqlsr -m QM3 -t TCP -p 8888 &
```

5. 启动以下通道:
   a. 在 QM1 上:

```
START CHANNEL('QM1.TO.QM2')
START CHANNEL('QM1.TO.QM3')
```

   b. 在 QM2 上:

```
START CHANNEL('QM2.TO.QM1')
```

c. 在 QM3 上：

```
START CHANNEL('QM3.TO.QM1')
```

6. 检查所有通道是否已启动：

```
DISPLAY CHSTATUS('QM1.TO.QM2')
DISPLAY CHSTATUS('QM1.TO.QM3')
DISPLAY CHSTATUS('QM2.TO.QM1')
DISPLAY CHSTATUS('QM3.TO.QM1')
```

方案 *2* 部件 *3:* 连接队列管理器并定义主题

关于此任务

将子队列管理器 QM2 和 QM3 连接到父队列管理器 QM1。

过程

1. 在 QM2 和 QM3 上，将父队列管理器设置为 QM1：

```
ALTER QMGR PARENT (QM1)
```

2. 在所有队列管理器上运行以下命令以检查子队列管理器是否已连接到父队列管理器：

```
DISPLAY PUBSUB TYPE(ALL)
```

将显示命令输出。 例如，以下是 QM1 的输出，其中突出显示了关键详细信息：

```
DISPLAY PUBSUB ALL
1 : DISPLAY PUBSUB ALL
AMQ8723: Display pub/sub status details.
QMNAME(QM1)               TYPE(LOCAL)
STATUS(ACTIVE)             SUBCOUNT(6)
TPCOUNT(9)
AMQ8723: Display pub/sub status details.
QMNAME(QM2) TYPE(CHILD)
STATUS(ACTIVE) SUBCOUNT(NONE)
TPCOUNT(NONE)
AMQ8723: Display pub/sub status details.
QMNAME(QM3) TYPE(CHILD)
STATUS(ACTIVE) SUBCOUNT(NONE)
TPCOUNT(NONE)
```

方案 *2 part 4:* 发布和预订主题

关于此任务

使用 amqspub.exe 和 amqssub.exe 应用程序来发布和预订主题。

过程

1. 在第一个命令窗口中运行以下命令：

```
amqspub Sport/Soccer QM2
```

2. 在第二个命令窗口中运行此命令：

```
amqssub Sport/Soccer QM1
```

3. 在第三个命令窗口中运行以下命令:

```
amqssub Sport/Soccer QM3
```

结果

第二个和第三个命令窗口中的 amqssub.exe 应用程序接收在第一个命令窗口中发布的消息。

相关任务

第 422 页的『发布/预订层次结构方案 1: 使用具有队列管理器名称别名的点到点通道』
这是以不同方式设置发布/预订层次结构以在队列管理器之间建立连接的三个方案中的第一个方案。 此方案设置发布/预订层次结构，该层次结构使用具有队列管理器名称别名的点到点通道。

第 429 页的『发布/预订层次结构方案 3: 使用集群通道添加队列管理器』
这是三种场景中的第三种场景，它们以不同方式设置发布/预订层次结构以建立队列管理器之间的连接。 此方案使用集群通道将队列管理器添加到层次结构。

相关信息

将队列管理器连接到发布/预订层次结构

发布/预订层次结构方案 3: 使用集群通道添加队列管理器

这是三种场景中的第三种场景，它们以不同方式设置发布/预订层次结构以建立队列管理器之间的连接。 此方案使用集群通道将队列管理器添加到层次结构。

关于此任务

这组方案都使用名为 QM1 的父队列管理器以及名为 QM2 和 QM3 的两个子队列管理器。

注: 此方案仅使用集群配置将队列管理器连接在一起，而不是通过集群主题传播发布/预订流量。 在同一集群中定义队列管理器之间的子/父层次结构关系时，将根据主题树中主题的发布和预订作用域设置在队列管理器之间传播发布内容。 请务必不要使用主题的集群名称设置将主题添加到集群中。 如果使用集群名称，那么拓扑将成为发布/预订集群，并且不需要定义子级/父级层次结构关系。 请参阅 第 416 页的『发布/预订集群方案』 和 规划分布式发布/预订网络。

方案 3 将拆分为更小的部分，以使过程更易于遵循。 此方案复用 第 422 页的『发布/预订层次结构方案 1: 使用具有队列管理器名称别名的点到点通道』中的方案 1 部件 1 ，方案 1 部件 3 和方案 1 部件 4。

此方案创建名为 DEMO 的集群，其中 QM1 和 QM2 是完整存储库， QM3 是部分存储库。 队列管理器 QM1 是队列管理器 QM2 和 QM3 的父代。

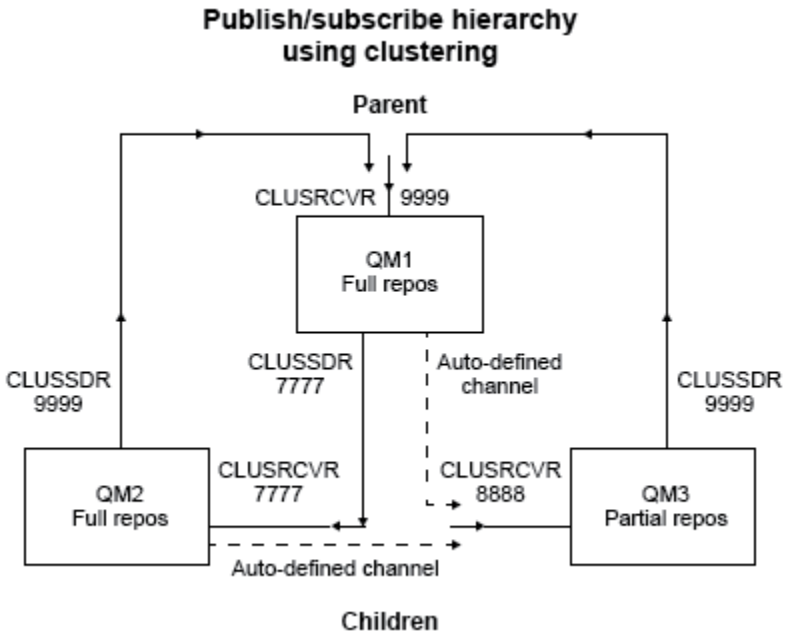


图 *85:* 显示使用集群通道的队列管理器之间的关系的拓扑图。

方案 *3 part 1:* 创建队列管理器并设置 *PSMODE*

过程

1. 使用以下命令创建并启动三个名为 QM1， QM2 和 QM3 的队列管理器:

```
crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM1
strmqm QM1

crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM2
strmqm QM2

crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM3
strmqm QM3
```

2. 通过在所有三个队列管理器上使用以下命令来启用队列管理器发布/预订方式:

```
ALTER QMGR PSMODE(ENABLED)
```

方案 *3 part 2:* 点到点通道连接

关于此任务

在集群的队列管理器之间建立点到点通道连接。

过程

1. 在 QM1 和 QM2 上，将 **REPOS** 参数设置为集群 DEMO 的名称:

```
ALTER QMGR REPOS(DEMO)
```

2. 在队列管理器上启动相应的侦听器:

```
runmqlsr -m QM1 -t TCP -p 9999 &
```

```
runmqlsr -m QM2 -t TCP -p 7777 &
runmqlsr -m QM3 -t TCP -p 8888 &
```

3. 在每个队列管理器上定义集群接收方通道:

   a. 在 QM1 上:

   ```
   DEFINE CHANNEL(TO.QM1) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('localhost(9999)')
   CLUSTER(DEMO)
   ```

   b. 在 QM2 上:

   ```
   DEFINE CHANNEL(TO.QM2) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('localhost(7777)')
   CLUSTER(DEMO)
   ```

   c. 在 QM3 上:

   ```
   DEFINE CHANNEL(TO.QM3) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('localhost(8888)')
   CLUSTER(DEMO)
   ```

4. 定义集群中每个队列管理器上的完整存储库的集群发送方通道:

   a. 在 QM1 上:

   ```
   DEFINE CHANNEL(TO.QM2) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('localhost(7777)')
   CLUSTER(DEMO)
   ```

   b. 在 QM2 上:

   ```
   DEFINE CHANNEL(TO.QM1) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('localhost(9999)')
   CLUSTER(DEMO)
   ```

   c. QM3 可以具有到 QM1 或 QM2 上的完整存储库的集群发送方通道。 此示例定义到 QM1 的通道:

   ```
   DEFINE CHANNEL(TO.QM1) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('localhost(9999)')
   CLUSTER(DEMO)
   ```

方案 *3* 部件 *3:* 连接队列管理器并定义主题

关于此任务

将子队列管理器 QM2 和 QM3 连接到父队列管理器 QM1。

过程

1. 在 QM2 和 QM3 上，将父队列管理器设置为 QM1:

   ```
   ALTER QMGR PARENT (QM1)
   ```

2. 在所有队列管理器上运行以下命令以检查子队列管理器是否已连接到父队列管理器:

   ```
   DISPLAY PUBSUB TYPE(ALL)
   ```

   将显示命令输出。 例如，以下是 QM1 的输出，其中突出显示了关键详细信息:

   ```
   DISPLAY PUBSUB ALL
   1 : DISPLAY PUBSUB ALL
   AMQ8723: Display pub/sub status details.
   QMNAME(QM1)               TYPE(LOCAL)
   STATUS(ACTIVE)             SUBCOUNT(6)
   ```

```
TPCOUNT(9)
AMQ8723: Display pub/sub status details.
QMNAME(QM2) TYPE(CHILD)
STATUS(ACTIVE) SUBCOUNT(NONE)
TPCOUNT(NONE)
AMQ8723: Display pub/sub status details.
QMNAME(QM3) TYPE(CHILD)
STATUS(ACTIVE) SUBCOUNT(NONE)
TPCOUNT(NONE)
```

方案 *3 part 4:* 发布和预订主题

关于此任务

使用 amqspub.exe 和 amqssub.exe 应用程序来发布和预订主题。

过程

1. 在第一个命令窗口中运行以下命令:

```
amqspub Sport/Soccer QM2
```

2. 在第二个命令窗口中运行此命令:

```
amqssub Sport/Soccer QM1
```

3. 在第三个命令窗口中运行以下命令:

```
amqssub Sport/Soccer QM3
```

结果

第二个和第三个命令窗口中的 amqssub.exe 应用程序接收在第一个命令窗口中发布的消息。

相关任务

第 422 页的『发布/预订层次结构方案 1: 使用具有队列管理器名称别名的点到点通道』
这是以不同方式设置发布/预订层次结构以在队列管理器之间建立连接的三个方案中的第一个方案。 此方案设置发布/预订层次结构，该层次结构使用具有队列管理器名称别名的点到点通道。

第 426 页的『发布/预订层次结构方案 2: 将同名的点到点通道用于传输队列和远程队列管理器』
这是三种场景中的第二种场景，它们以不同方式设置发布/预订层次结构，以建立队列管理器之间的连接。此方案设置一个发布/预订层次结构，该层次结构使用与远程队列管理器具有相同传输队列名称的点到点通道。

相关信息

将队列管理器连接到发布/预订层次结构

业务支持方案

通过使用事务支持，您可以使应用程序能够可靠地处理数据库。

本部分介绍了事务支持。 使应用程序能够将 IBM MQ 与数据库产品配合使用所需的工作跨应用程序编程和系统管理领域。 将此处的信息与 落实和回退工作单元一起使用。

我们首先介绍构成事务的工作单元，然后描述您启用 IBM MQ 以将事务与数据库进行协调的方式。

相关概念

第 433 页的『介绍工作单元』
本主题介绍并定义工作单元，落实，回退和同步点的一般概念。 它还包含说明全局工作单元的两种方案。

相关信息

IBM MQ 和 HP NonStop TMF

介绍工作单元

本主题介绍并定义工作单元，落实，回退和同步点的一般概念。 它还包含说明全局工作单元的两种方案。

当程序将消息放入工作单元中的队列时，仅当程序 落实 工作单元时，这些消息才会对其他程序可视。 要落实工作单元，所有更新都必须成功，才能保持数据完整性。

如果程序检测到错误并决定不使 put 操作永久存在，那么它可以 回退 工作单元。 当程序执行回退时， IBM MQ 通过除去该工作单元放在队列上的消息来复原队列。

同样，当程序从工作单元中的一个或多个队列获取消息时，这些消息将保留在队列上，直到程序落实工作单元为止，但这些消息不可供其他程序检索。 当程序落实工作单元时，将从队列中永久删除这些消息。 如果程序回退工作单元，那么 IBM MQ 会通过使消息可供其他程序检索来复原队列。

在最简单的情况下，将在任务结束时作出落实或回退更改的决定。 但是，应用程序在任务中的其他逻辑点同步数据更改可能更有用。 这些逻辑点称为同步点 (或同步点) ，在两个同步点之间处理一组更新的时间段称为工作单元。 可以有几个 MQGET 调用和 MQPUT 调用作为一个工作单元的一部分。

通过 IBM MQ，我们需要区分 本地 和 全局 工作单元:

本地工作单元
是将唯一操作放入并从 IBM MQ 队列中获取的操作，并且使用 单阶段落实 进程在队列管理器中提供每个工作单元的协调。

当要更新的唯一资源是由单个 IBM MQ 队列管理器管理的队列时，请使用本地工作单元。 使用 MQCMIT 动词落实更新或使用 MQBACK 回退更新。

除了使用本地工作单元所涉及的日志管理以外，没有任何系统管理任务。 在将 MQPUT 和 MQGET 调用与 MQCMIT 和 MQBACK 配合使用的应用程序中，请尝试使用 MQPMO_SYNCPOINT 和 MQGMO_SYNCPOINT 选项。 (有关日志管理的信息，请参阅 管理日志文件。)

全局工作单元
是否还会更新其他资源 (例如关系数据库中的表)。 如果涉及多个 资源管理器 ，那么需要使用 两阶段落实 进程来协调全局工作单元的 事务管理器 软件。

如果还需要包含对关系数据库管理器软件 (例如 Db2， Oracle， Sybase 和 Informix) 的更新，请使用全局工作单元。

有几种使用全局工作单元的可能方案。 此处记录了两种场景:

1. 首先，队列管理器本身充当事务管理器。 在此场景中， MQI 动词控制全局工作单元; 它们在应用程序中使用 MQBEGIN 动词启动，然后使用 MQCMIT 落实或使用 MQBACK 回退。
2. 在第二个角色中，事务管理员角色由其他软件 (例如 TXSeries， Encina 或 Tuxedo) 执行。 在此场景中，事务管理器软件提供的 API 用于控制工作单元 (例如， EXEC CICS SYNCPOINT for TXSeries)。

以下部分描述了使用全局工作单元所需的所有步骤 (按两种方案组织):

- 场景 1: 队列管理器执行协调
- 第 454 页的『方案 2: 其他软件提供协调』

distributed 方案 1: 队列管理器执行协调

在方案 1 中，队列管理器充当事务管理器。 在此场景中， MQI 动词控制全局工作单元; 它们在应用程序中使用 MQBEGIN 动词启动，然后使用 MQCMIT 落实或使用 MQBACK 回退。

隔离级别

在 IBM MQ 中，根据在数据库中实现的事务隔离设计，队列上的消息可能在数据库更新之前可视。

当 IBM MQ 队列管理器作为 XA 事务管理器工作时，要协调 XA 资源管理器的更新，请遵循以下落实协议:

1. 准备所有 XA 资源管理器。
2. 落实 IBM MQ 队列管理器资源管理器。
3. 落实其他资源管理器。

在步骤 2 和 3 之间，应用程序可能会看到一条落实到队列的消息，但数据库中的相应行不会反映此消息。

如果配置了数据库以使应用程序的数据库 API 调用等待暂挂更新完成，那么这不是问题。

您可以通过以不同方式配置数据库来解决此问题。 所需配置的类型称为 "隔离级别"。 有关隔离级别的更多信息，请参阅数据库文档。 或者，您可以配置队列管理器以按以下反向顺序落实资源管理器:

1. 准备所有 XA 资源管理器。
2. 落实其他资源管理器。
3. 落实 IBM MQ 队列管理器资源管理器。

更改协议时，将最后落实 IBM MQ 队列管理器，因此从队列中读取消息的应用程序只有在完成相应的数据库更新后才会看到消息。

要配置队列管理器以使用此已更改的协议，请设置 **AMQ_REVERSE_COMMIT_ORDER** 环境变量。

在运行 **strmqm** 以启动队列管理器的环境中设置此环境变量。 例如，在启动队列管理器之前，请在 shell 中运行以下命令:

```
export AMQ_REVERSE_COMMIT_ORDER=1
```

注: 设置此环境变量可能会导致每个事务有额外的日志条目，因此这将对每个事务的性能产生较小的影响。

数据库协调

当队列管理器协调全局工作单元本身时，可以在工作单元内集成数据库更新。 即，可以编写混合 MQI 和 SQL 应用程序，并且可以使用 MQCMIT 和 MQBACK 动词来落实或回滚对队列和数据库的更改。

队列管理器使用 *X/Open* 分布式事务处理*: XA* 规范中描述的两阶段落实协议来实现此目标。 要落实工作单元时，队列管理器首先会询问每个参与的数据库管理器是否准备落实其更新。 仅当所有参与者 (包括队列管理器本身) 准备落实时，才会落实所有队列和数据库更新。 如果任何参与者无法准备其更新，那么将改为回退工作单元。

通常，全局工作单元通过以下方法在应用程序中实现 (伪代码):

```
MQBEGIN
MQGET (在消息选项中包含标志 MQGMO_SYNCPOINT)
MQPUT (在消息选项中包含标志 MQPMO_SYNCPOINT)
SQL INSERT
MQCMIT
```

MQBEGIN 的目的是表示全局工作单元的开始。 MQCMIT 的目的是表示全局工作单元的结束，并使用两阶段落实协议在所有参与的资源管理器中完成该工作单元。

当工作单元 (也称为 事务) 时 使用 MQCMIT 成功完成，在该工作单元中执行的所有操作都是永久或不可逆的。 如果工作单元因任何原因而失败，那么将改为回退所有操作。 无法使工作单元中的一个操作成为永久操作，而另一个操作已回退。 这是工作单元的原则: 要么工作单元内的所有操作都是永久性的，要么都不是永久性的。

注:

1. 应用程序员可以通过调用 MQBACK 来强制回退工作单元。 如果在调用 MQCMIT 之前应用程序或数据库失败 ，那么队列管理器也会回退工作单元。
2. 如果应用程序在不调用 MQCMIT 的情况下调用 MQDISC ，那么队列管理器的行为就像调用了 MQCMIT 一样，并落实工作单元。

在 MQBEGIN 和 MQCMIT 之间，队列管理器不会对数据库进行任何调用以更新其资源。 即，更改数据库表的唯一方式是由您的代码 (例如，伪代码中的 SQL INSERT) 进行更改。

如果队列管理器在落实协议期间失去与任何数据库管理器的联系，那么将提供完全恢复支持。 如果数据库管理器在处于不确定状态时变为不可用，即，它已成功准备落实，但尚未收到落实或回退决策，那么队列管理器将记住工作单元的结果，直到该结果成功交付到数据库为止。 同样，如果队列管理器因未完成的落实操作

而终止，那么将在队列管理器重新启动时记住这些操作。 如果应用程序意外终止，那么工作单元的完整性不会受到损害，但结果取决于应用程序在流程中终止的位置，如 第 435 页的表 31 中所述。

下表概述了在数据库或应用程序失败时发生的情况:

表 *30:* 数据库服务器发生故障时发生的情况

故障发生	**结果**
在应用程序调用 MQCMIT 之前。	工作单元已回退。
在应用程序调用 MQCMIT 期间， **之前** 所有数据库都指示已成功准备。	工作单元已回退，原因码为 MQRC_BACKED_OUT。
在应用程序调用 MQCMIT 期间， **之后** 所有数据库都指示已成功准备，但之前都指示已成功落实。	工作单元由队列管理器保持在可恢复状态，原因码为 MQRC_OUTCOME_PENDING。
在应用程序调用 MQCMIT 期间， **之后** 所有数据库都指示它们已成功落实。	已落实工作单元，原因码为 MQRC_NONE。
在应用程序调用 MQCMIT 之后。	已落实工作单元，原因码为 MQRC_NONE。

表 *31:* 应用程序失败时发生的情况

故障发生	**结果**
在应用程序调用 MQCMIT 之前。	工作单元已回退。
在应用程序调用 MQCMIT 期间， **之前** 队列管理器已接收到应用程序的 MQCMIT 请求。	工作单元已回退。
在应用程序调用 MQCMIT 期间， **在** 队列管理器接收到应用程序的 MQCMIT 请求之后。	队列管理器尝试使用两阶段落实 (取决于数据库产品成功执行并落实其部分工作单元) 进行落实。

如果从 MQCMIT 返回的原因码为 MQRC_OUTCOME_PENDING，那么队列管理器将记住工作单元，直到它能够重新建立与数据库服务器的联系，并告诉它落实其部分工作单元为止。 请参阅 第 449 页的『丢失与 XA 资源管理器的联系时的注意事项』，以获取有关如何以及何时执行恢复的信息。

队列管理器使用 XA 接口与数据库管理器通信，如 *X/Open* 分布式事务处理*: XA* 规范中所述。 这些函数调用的示例包括 xa_open， xa_start， xa_end， xa_prepare 和 xa_commit。 我们使用术语 事务管理器 和 资源管理器 与 XA 规范中使用的术语相同。

限制

数据库协调支持存在限制。

具有以下限制:

- 在 MQI 客户机应用程序中 **不** 支持在 IBM MQ 工作单元内协调数据库更新的功能。 在客户机应用程序中使用 MQBEGIN 失败。 调用 MQBEGIN 的程序必须在与队列管理器相同的机器上作为 服务器 应用程序运行。

  注: 服务器 应用程序是已与必需的 IBM MQ 服务器库链接的程序; 客户机 应用程序是已与必需的 IBM MQ 客户机库链接的程序。 请参阅 为 IBM MQ MQI clients 和 构建过程应用程序 构建应用程序，以获取有关编译和链接以过程语言编写的程序的详细信息。
- 数据库服务器可以驻留在与队列管理器服务器不同的机器上，只要数据库客户机与队列管理器安装在同一机器上，它就支持此功能。 请参阅数据库产品的文档以确定其客户机软件是否可用于两阶段落实系统。
- 虽然队列管理器充当资源管理器 (为了参与方案 2 全局工作单元)，但无法使一个队列管理器在其方案 1 全局工作单元中协调另一个队列管理器。

切换装入文件

交换机装入文件是由 IBM MQ 应用程序和队列管理器中的代码装入的共享库 (Windows 系统上的 DLL)。 其目的是简化数据库的客户机共享库的装入，并将指针返回到 XA 函数。

必须在启动队列管理器之前指定交换机装入文件的详细信息。 详细信息放在 Windows UNIX 和 Linux 系统上的 qm.ini 文件中。

- 在 Windows 和 Linux (x86 和 x86-64 平台) 系统上，使用 MQ Explorer 来更新 qm.ini 文件。
- 在所有其他系统上，直接编辑文件 qm.ini。

如果交换机装入文件的 C 源支持方案 1 全局工作单元，那么会随 IBM MQ 安装一起提供该文件的 C 源。 源包含名为 MQStart 的函数。 装入交换机装入文件时，队列管理器将调用此函数，这将返回称为 *XA* 交换机的结构的地址。

XA 切换结构存在于数据库客户机共享库中，并且包含许多函数指针，如 第 436 页的表 32 中所述:

表 *32: XA* 切换函数指针

函数指针名称	**XA 函数**	**用途**
xa_open_entry	xa_open	连接到数据库
xa_close_entry	xa_close	与数据库断开连接
xa_start_entry	xa_start	启动全局工作单元的分支
xa_end_entry	xa_end	暂挂全局工作单元的分支
xa_rollback_entry	XA_rollback	回滚全局工作单元的分支
xa_prepare_entry	xa_prepare	准备落实全局工作单元的分支
xa_commit_entry	xa_commit	落实全局工作单元的分支
xa_recover_entry	xa_recover	从数据库中发现它是否具有不确定的工作单元
xa_forget_entry	xa_算了	允许数据库忘记全局工作单元的分支
xa_complete_entry	xa_complete	完成全局工作单元的分支

在应用程序中的第一个 MQBEGIN 调用期间，作为 MQBEGIN 的一部分执行的 IBM MQ 代码将装入切换装入文件，并在数据库共享库中调用 xa_open 函数。 类似地，在队列管理器启动期间以及在其他后续场合，某些队列管理器进程装入切换装入文件并调用 xa_open。

您可以使用 动态注册来减少 xa_ * 调用数。 有关此优化方法的完整描述，请参阅 第 452 页的『XA 动态注册』。

配置系统以进行数据库协调

在数据库管理器可以参与由队列管理器协调的全局工作单元之前，必须执行若干任务。 这些描述如下:

- 第 436 页的『安装和配置数据库产品』
- 第 437 页的『创建交换机装入文件』
- 第 437 页的『向队列管理器添加配置信息』
- 第 439 页的『编写和修改应用程序』
- 第 439 页的『测试系统』

安装和配置数据库产品

要安装和配置数据库产品，请参阅产品自己的文档。 本部分中的本主题描述了常规配置问题以及它们如何与 IBM MQ 与数据库之间的互操作相关。

数据库连接

与队列管理器建立标准连接的应用程序与单独的本地队列管理器代理进程中的线程相关联。 (不是 *fastpath* 连接的连接是此上下文中的 标准 连接。 请参阅 使用 MQCONNX 调用连接到队列管理器。)

当应用程序发出 MQBEGIN 时，它和代理进程都会在数据库客户机库中调用 xa_open 函数。 为此，数据库客户机库代码 将 连接到 来自应用程序和队列管理器进程的工作单元中涉及的数据库。 只要应用程序保持与队列管理器的连接，就会维护这些数据库连接。

如果数据库仅支持有限数量的用户或连接，那么这是一个重要的注意事项，因为正在与数据库建立两个连接以支持一个应用程序。

客户机/服务器配置

装入到 IBM MQ 队列管理器和应用程序进程中的数据库客户机库 **必须** 能够发送到其服务器并从其服务器接收。 请确保:

- 数据库的客户机/服务器配置文件具有正确的详细信息
- 相关环境变量是在队列管理器 **和** 应用程序进程的环境中设置的

创建交换机装入文件

IBM MQ 随附样本 makefile ，用于为受支持的数据库管理器构建切换装入文件。

MQ_INSTALLATION_PATH 表示 IBM MQ 安装所在的高级目录。

样本 makefile 与构建交换机装入文件所需的所有关联 C 源文件一起安装在以下目录中:

- 对于 IBM MQ for Windows，在 *MQ_INSTALLATION_PATH*\tools\c\samples\xatm\ 目录中
- 对于 IBM MQ for UNIX 和 Linux 系统，在 *MQ_INSTALLATION_PATH*/samp/xatm/ 目录中

用于构建交换机装入文件的样本源模块包括:

- 对于 Db2， db2swit.c
- 对于 Oracle， oraswit.c
- 对于 Informix， infswit.c
- 对于 Sybase， sybswit.c

生成交换机装入文件时，请在 /var/mqm/exits 中安装 32 位交换机装入文件，并在 /var/mqm/exits64 中安装 64 位交换机装入文件。

注: 此切换装入文件由 IBM MQ C 应用程序使用。 对于 Java 应用程序，请参阅 使用 IBM MQ classes for Java 的 JTA/JDBC 协调。

如果您有 32 位队列管理器，那么样本 make 文件 xaswit.mak 将在 /var/mqm/exits 中安装 32 位交换机装入文件。

如果您有 64 位队列管理器，那么样本 make 文件 xaswit.mak 将在 /var/mqm/exits 中安装 32 位交换机装入文件，在 /var/mqm/exits64 中安装 64 位交换机装入文件。

文件安全性

由于 IBM MQ 无法控制的原因，操作系统可能无法通过 IBM MQ 装入交换机装入文件。 如果发生此情况，那么会将错误消息写入 IBM MQ 错误日志，并且可能导致 MQBEGIN 调用失败。 为了帮助确保操作系统不会使交换机装入文件的装入失败，您必须满足以下要求:

1. 切换装入文件必须在 qm.ini 文件中提供的位置中可用。
2. 交换机装入文件必须可供需要装入该文件的所有进程 (包括队列管理器进程和应用程序进程) 访问。
3. 交换机装入文件所依赖的所有库 (包括数据库产品提供的库) 都必须存在且可访问。

向队列管理器添加配置信息

为数据库管理器创建切换装入文件并将其放在安全位置时，必须向队列管理器指定该位置。

要指定位置，请执行以下步骤:

- 在 Windows 和 Linux (x86 和 x86-64 平台) 系统上，使用 IBM MQ Explorer。 在队列管理器属性面板中的 XA 资源管理器下指定切换装入文件的详细信息。

- 在所有其他系统上，在队列管理器的 qm.ini 文件的 XAResourceManager 节中指定交换机装入文件的详细信息。

为队列管理器要协调的数据库添加 XAResourceManager 节。 最常见的情况是只有一个数据库，因此只有一个 XAResourceManager 节。 有关涉及多个数据库的更复杂的配置的详细信息，请参阅 第 448 页的『多个数据库配置』。 XAResourceManager 节的属性如下所示:

名称 = 名称
用户选择的用于标识资源管理器的字符串。 实际上，它为 XAResourceManager 节提供了名称。 该名称是必需的，长度最多可以为 31 个字符。

您选择的名称必须唯一; 此 qm.ini 文件中必须只有一个具有此名称的 XAResourceManager 节。 该名称也应该有意义，因为在使用 `dspmqtrn` 命令时，队列管理器使用它来在队列管理器错误日志消息和输出中引用此资源管理器。 (请参阅 第 449 页的『使用 dspmqtrn 命令显示未完成的工作单元』 以获取更多信息。)

选择名称并启动队列管理器后，请勿更改 "名称" 属性。 有关更改配置信息的更多详细信息，请参阅 第 451 页的『更改配置信息』。

SwitchFile= 名称
这是先前构建的 XA 切换装入文件的名称。 这是必需属性。 队列管理器和 IBM MQ 应用程序进程中的代码两次尝试装入交换机装入文件:

1. 在队列管理器启动时
2. 在 IBM MQ 应用程序进程中首次调用 MQBEGIN 时

交换机装入文件的安全性和许可权属性必须允许这些进程执行此操作。

XAOpenString= 字符串
这是 IBM MQ 代码在其对数据库管理器的 xa_open 函数的调用中传递的数据字符串。 这是可选属性; 如果省略此属性，那么将假定字符串长度为零。

队列管理器和 IBM MQ 应用程序进程中的代码两次调用 xa_open 函数:

1. 在队列管理器启动时
2. 在 IBM MQ 应用程序进程中首次调用 MQBEGIN 时

此字符串的格式特定于每个数据库产品，将在该产品的文档中进行描述。 通常， xa_open 字符串包含认证信息 (用户名和密码) ，以允许连接到队列管理器和应用程序进程中的数据库。

V 8.0.0.4 从 IBM MQ 8.0.0 修订包 4 开始，当 XAOpenString 包含密码时，您可以获取 IBM MQ 以保护此信息，而不是使密码在 `qm.ini` 文件中以纯文本显示。 IBM MQ 将用户名和密码 (以加密形式) 存储在另一个文件中，并使用这些凭证连接到数据库。 有关详细信息，请参阅 保护数据库认证详细信息。

XACloseString= 字符串
这是 IBM MQ 代码在其对数据库管理器的 xa_close 函数的调用中传递的数据字符串。 这是可选属性; 如果省略此属性，那么将假定字符串长度为零。

队列管理器和 IBM MQ 应用程序进程中的代码两次调用 xa_close 函数:

1. 在队列管理器启动时
2. 在 IBM MQ 应用程序进程中调用 MQDISC 时，先前已调用 MQBEGIN

此字符串的格式特定于每个数据库产品，将在该产品的文档中进行描述。 通常，该字符串为空，并且通常会从 XAResourceManager 节中省略 XACloseString 属性。

ThreadOfControl=THREAD | PROCESS
ThreadOf 控制值可以是 THREAD 或 PROCESS。 队列管理器将其用于序列化目的。 这是可选属性; 如果省略此属性，那么将采用值 PROCESS。

如果数据库客户机代码允许线程在不序列化的情况下调用 XA 函数，那么 ThreadOfControl 的值可以是 THREAD。 队列管理器假定它可以在必要时同时从多个线程调用数据库客户机共享库中的 XA 函数。

如果数据库客户机代码不允许线程以此方式调用其 XA 函数，那么 ThreadOfControl 的值必须为 PROCESS。 在这种情况下，队列管理器会序列化对数据库客户机共享库的所有调用，以便一次仅从特定

进程中发出一个调用。 如果应用程序使用多个线程运行，那么您可能还需要确保应用程序执行类似的序列化。

请注意，数据库产品以这种方式处理多线程进程的能力是该产品的供应商面临的问题。 有关是否可以将 ThreadOf 控制属性设置为 THREAD 或 PROCESS 的详细信息，请参阅数据库产品的文档。 如果可以，建议将 ThreadOf 控件设置为 THREAD。 如果不确定，那么 更安全 选项将其设置为 PROCESS ，尽管您将失去使用 THREAD 的潜在性能优势。

编写和修改应用程序
如何实现全局工作单元。

第 433 页的『介绍工作单元』中描述了随 IBM MQ 安装一起提供的方案 1 全局工作单元的样本应用程序。

通常，全局工作单元通过以下方法在应用程序中实现 (伪代码):

```
MQBEGIN
MQGET
MQPUT
SQL INSERT
MQCMIT
```

MQBEGIN 的目的是表示全局工作单元的开始。 MQCMIT 的目的是表示全局工作单元的结束，并使用两阶段落实协议在所有参与的资源管理器中完成该工作单元。

在 MQBEGIN 和 MQCMIT 之间，队列管理器不会对数据库进行任何调用以更新其资源。 即，更改数据库表的唯一方式是由您的代码 (例如，伪代码中的 SQL INSERT) 进行更改。

就数据库而言，队列管理器的作用是告诉它全局工作单元何时开始，何时结束，以及全局工作单元是应该落实还是回滚。

就应用程序而言，队列管理器执行两个角色: 资源管理器 (其中资源是队列上的消息) 和全局工作单元的事务管理器。

从提供的样本程序开始，处理在这些程序中进行的各种 IBM MQ 和数据库 API 调用。 相关的 API 调用完全记录在 样本 IBM MQ 过程程序， MQI 中使用的数据类型以及 (对于数据库自己的 API) 数据库自己的文档中。

测试系统
您知道是否仅通过在测试期间运行应用程序和系统来正确配置这些应用程序和系统。 您可以通过构建和运行提供的其中一个样本程序来测试系统的配置 (队列管理器与数据库之间的成功通信)。

配置 Db2

Db2 支持和配置信息。

Db2 的受支持级别是在 WebSphere MQ 详细系统需求 页面上定义的。

注: 在队列管理器为 64 位的平台上不支持 Db2 的 32 位实例。

请执行以下操作:

1. 请检查环境变量设置。
2. 创建 Db2 交换机装入文件。
3. 添加资源管理器配置信息。
4. 必要时更改 Db2 配置参数。

请结合 第 436 页的『配置系统以进行数据库协调』中提供的一般信息阅读本信息。

警告: 如果在 UNIX 和 Linux 平台上运行 `db2profile` ，那么将设置环境变量 LIBPATH 和 LD_LIBRARY_PATH。 建议使用 `unset` 这些环境变量。 请参阅 crtmqenv 或 setmqenv 以获取更多信息。

检查 Db2 环境变量设置

确保为队列管理器进程 **以及应用程序进程在** 中设置了 Db2 环境变量。 特别是，必须始终设置启动队列管理器的 DB2INSTANCE 环境变量 **前** 。 DB2INSTANCE 环境变量标识包含要更新的 Db2 数据库的 Db2 实例。 例如：

- 在 UNIX 和 Linux 系统上，使用：

```
export DB2INSTANCE=db2inst1
```

- 在 Windows 系统上，使用：

```
set DB2INSTANCE=DB2
```

在具有 Db2 数据库的 Windows 上，必须将用户 MUSR_MQADMIN 添加到 DB2USERS 组以允许队列管理器启动。

创建 Db2 交换机装入文件

创建 Db2 切换装入文件的最简单方法是使用样本文件 xaswit.mak， IBM MQ 提供该样本文件来为各种数据库产品构建切换装入文件。

在 Windows 系统上，可以在目录 *MQ_INSTALLATION_PATH*\tools\c\samples\xatm 中找到 xaswit.mak 。 *MQ_INSTALLATION_PATH* 表示 IBM MQ 安装所在的高级目录。 要使用 Microsoft Visual C++ 创建 Db2 切换装入文件，请使用：

```
nmake /f xaswit.mak db2swit.dll
```

生成的交换机文件放置在 C:\Program Files\IBM\WebSphere MQ\exits 中。

您可以在目录 *MQ_INSTALLATION_PATH*/samp/xatm 中找到 xaswit.mak 。 *MQ_INSTALLATION_PATH* 表示 IBM MQ 安装所在的高级目录。

编辑 xaswit.mak 以 取消注释 与您正在使用的 Db2 版本相应的行。 然后使用以下命令执行 makefile：

```
make -f xaswit.mak db2swit
```

生成的 32 位交换机装入文件将放在 /var/mqm/exits 中。

生成的 64 位交换机装入文件放置在 /var/mqm/exits64 中。

添加 Db2 的资源管理器配置信息

您必须修改队列管理器的配置信息，以将 Db2 声明为全局工作单元中的参与者。 在 第 437 页的『向队列管理器添加配置信息』中的更多详细信息中描述了如何以此方式修改配置信息。

- 在 Windows 和 Linux (x86 和 x86-64 平台) 系统上，使用 IBM MQ Explorer。 在队列管理器属性面板中的 XA 资源管理器下指定切换装入文件的详细信息。
- 在所有其他系统上，在队列管理器的 qm.ini 文件的 XAResourceManager 节中指定交换机装入文件的详细信息。

第 441 页的图 86 是 UNIX 样本，显示一个 XAResourceManager 条目，其中要协调的数据库名为 mydbname，此名称在 XAOpenString 中指定：

```
XAResourceManager:
  Name=mydb2
  SwitchFile=db2swit
  XAOpenString=mydbname,myuser,mypasswd,toc=t
  ThreadOfControl=THREAD
```

图 *86: UNIX* 平台 上 *Db2* 的样本 *XAResourceManager* 条目

注:

1. `ThreadOfControl=THREAD` 不能与版本 8 之前的 Db2 版本配合使用。 将 `ThreadOfControl` 和 XAOpenString 参数 `toc` 设置为下列其中一个组合:
   - `ThreadOfControl=THREAD` 和 `toc=t`
   - `ThreadOfControl=PROCESS` 和 `toc=p`

   如果要使用 jdbcdb2 XA 切换装入文件来启用 JDBC/JTA 协调，那么必须使用 `ThreadOfControl=PROCESS` 和 `toc=p`。

更改 Db2 配置参数

对于队列管理器正在协调的每个 Db2 数据库，必须设置数据库特权，更改 tp_mon_name 参数，并重置 maxappls 参数。 要进行此操作，请执行以下步骤:

设置数据库特权

队列管理器进程在 UNIX 和 Linux 系统上以有效用户和组 mqm 运行。 在 Windows 系统上，它们以启动队列管理器的用户身份运行。 这可以是以下之一:

1. 发出 `strmqm` 命令的用户，或者
2. 运行 IBM MQ Service COM 服务器的用户

缺省情况下，此用户称为 MUSR_MQADMIN。

如果未在 xa_open 字符串上指定用户名和密码，那么 Db2 将使用 **运行队列管理器的用户** 来认证 xa_open 调用。 如果此用户 (例如， UNIX 和 Linux 系统上的用户 mqm) 在数据库中没有最低特权，那么数据库将拒绝认证 xa_open 调用。

同样的注意事项也适用于您的应用程序进程。 如果未在 xa_open 字符串上指定用户名和密码，那么 Db2 将使用运行应用程序的用户来认证在第一个 MQBEGIN 期间发出的 xa_open 调用。 同样，此用户必须在数据库中具有最低特权才能工作。

例如，通过发出以下 Db2 命令，在 mydbname 数据库中授予 mqm 用户连接权限:

```
db2 connect to mydbname
db2 grant connect on database to user mqm
```

有关安全性的更多信息，请参阅第 449 页的『安全性注意事项』。

Windows **更改 TP_MON_NAME 参数**

仅对于 Windows 系统上的 Db2 ，请更改 TP_MON_NAME 配置参数，以命名 Db2 用于调用队列管理器以进行动态注册的 DLL。

使用命令 `db2 update dbm cfg using TP_MON_NAME mqmax` 来命名 MQMAX.DLL 作为 Db2 用于调用队列管理器的库。 这必须存在于 PATH 中的目录中。

重置 maxappls 参数

您可能需要查看 *maxappls* 参数的设置，这将限制可以连接到数据库的最大应用程序数。 请参阅第 436 页的『安装和配置数据库产品』。

配置 Oracle

Oracle 支持和配置信息。

完成以下步骤:

1. 检查环境变量设置。
2. 创建 Oracle 交换机装入文件。
3. 添加资源管理器配置信息。
4. 根据需要更改 Oracle 配置参数。

在 WebSphere MQ 详细系统需求 页面上提供了 IBM MQ 支持的 Oracle 级别的当前列表。

检查 Oracle 环境变量设置

确保为队列管理器进程以及应用程序进程设置了 Oracle 环境变量。 尤其是，在启动队列管理器之前，请始终设置以下环境变量:

ORACLE_HOME
Oracle 主目录。 例如，在 UNIX 和 Linux 系统上，使用:

```
export ORACLE_HOME=/opt/oracle/product/8.1.6
```

在 Windows 系统上，使用:

```
set ORACLE_HOME=c:\oracle\ora81
```

ORACLE_SID
正在使用的 Oracle SID。 如果要将 Net8 用于客户机/服务器连接，那么可能不必设置此环境变量。 请参阅 Oracle 文档。

以下示例是在 UNIX 和 Linux 系统上设置此环境变量的示例:

```
export ORACLE_SID=sid1
```

Windows 系统上的等效项为:

```
set ORACLE_SID=sid1
```

注: 必须将 PATH 环境变量设置为包含二进制文件目录 (例如 `ORACLE_INSTALL_DIR/VERSION/32BIT_NAME/bin` 或 `ORACLE_INSTALL_DIR/VERSION/64BIT_NAME/bin`)，否则您可能会看到一条消息，指出机器中缺少 oraclient 库。

如果在 Windows 64 位系统上运行队列管理器，那么必须仅安装 64 位 Oracle 客户机。 由 64 位队列管理器装入的交换机装入文件必须访问 Oracle 64 位客户机库。

创建 Oracle 交换机装入文件

要创建 Oracle 交换机装入文件，请使用样本文件 `xaswit.mak`， IBM MQ 提供此文件来为各种数据库产品构建交换机装入文件。 在 Windows 系统上，可以在目录 `C:\Program Files\IBM\WebSphere MQ\tools\c\samples\xatm` 中找到 `xaswit.mak` 。 要使用 Microsoft Visual C + + 创建 Oracle 切换装入文件，请使用: `nmake /f xaswit.mak oraswit.dll`

注: 这些交换机装入文件只能与 C 应用程序配合使用。 对于 Java 应用程序，请参阅 使用 IBM MQ classes for Java 的 JTA/JDBC 协调。

生成的交换机文件放置在 *MQ_INSTALLATION_PATH*`\exits` 中。 *MQ_INSTALLATION_PATH* 表示 IBM MQ 安装所在的高级目录。

您可以在目录 *MQ_INSTALLATION_PATH*`/samp/xatm` 中找到 `xaswit.mak` 。 *MQ_INSTALLATION_PATH* 表示 IBM MQ 安装所在的高级目录。

编辑 xaswit.mak 以取消注释与您正在使用的 Oracle 版本相应的行。 然后使用以下命令执行 makefile:

```
make -f xaswit.mak oraswit
```

安装 IBM MQ 时， MQ_INSTALLATION_PATH/samp/xatm 的内容是只读的，因此要编辑 xaswit.mak，将 samp/xatm 中的所有文件复制到另一个目录，修改 xaswit.mak，然后从该目录运行 make -f xaswit.makoraswit 。

生成的 32 位交换机装入文件放置在 /var/mqm/exits 中。

生成的 64 位交换机装入文件放置在 /var/mqm/exits64 中。

为 Oracle 添加资源管理器配置信息

您必须修改队列管理器的配置信息，以将 Oracle 声明为全局工作单元中的参与者。 在 第 437 页的『向队列管理器添加配置信息』中更详细地描述了如何以此方式修改队列管理器的配置信息。

- 在 Windows 和 Linux (x86 和 x86-64 平台) 系统上，使用 MQ Explorer。 在队列管理器属性面板中的 XA 资源管理器下指定切换装入文件的详细信息。
- 在所有其他系统上，在队列管理器的 qm.ini 文件的 XAResourceManager 节中指定交换机装入文件的详细信息。

第 443 页的图 87 是显示 XAResourceManager 条目的 UNIX 和 Linux 系统样本。 必须将 LogDir 添加到 XA 打开字符串，以便将所有错误和跟踪信息记录到同一位置。

```
XAResourceManager:
  Name=myoracle
  SwitchFile=oraswit
  XAOpenString=Oracle_XA+Acc=P/myuser/mypasswd+SesTm=35+LogDir=/tmp+threads=true
  ThreadOfControl=THREAD
```

图 *87: UNIX* 和 *Linux* 平台上 *Oracle* 的样本 *XAResourceManager* 条目

注:

1. 在 第 443 页的图 87 中， xa_open 字符串已与四个参数配合使用。 可以包括其他参数，如 Oracle 的文档中所述。
2. 使用 IBM MQ 参数 ThreadOfControl=THREAD 时，必须使用 XAResourceManager 节中的 Oracle 参数 +threads=true 。

有关 xa_open 字符串的更多信息，请参阅 *Oracle8 Server Application Developer 's Guide* 。

更改 Oracle 配置参数

对于队列管理器正在协调的每个 Oracle 数据库，必须查看最大会话数并设置数据库特权。 为此，请完成以下步骤:

查看最大会话数

您可能必须查看 LICENSE_MAX_SESSIONS 和 PROCESSES 设置以考虑属于队列管理器的进程所需的其他连接。 请参阅第 436 页的『安装和配置数据库产品』以获取更多详细信息。

设置数据库特权

xa_open 字符串中指定的 Oracle 用户名必须具有访问 DBA_PENDING_TRANSACTIONS 视图的特权，如 Oracle 文档中所述。

可以使用以下示例命令来授予必需的特权:

```
grant select on DBA_PENDING_TRANSACTIONS to myuser;
```

配置 Informix

Informix 支持和配置信息。

完成以下步骤:

1. 确保已安装相应的 Informix 客户端 SDK:
   - 32 位队列管理器和应用程序需要 32 位 Informix 客户机 SDK。
   - 64 位队列管理器和应用程序需要 64 位 Informix 客户机 SDK。
2. 确保正确创建 Informix 数据库。
3. 检查环境变量设置。
4. 构建 Informix 交换机装入文件。
5. 添加资源管理器配置信息。

在 WebSphere MQ 详细系统需求 页面上提供了 IBM MQ 支持的 Informix 级别的当前列表。

确保正确创建 Informix 数据库

必须创建要由 IBM MQ 队列管理器协调的每个 Informix 数据库，并指定 `log` 参数。 例如:

```
create database mydbname with log;
```

IBM MQ 队列管理器无法协调在创建时未指定 `log` 参数的 Informix 数据库。 如果队列管理器尝试协调在创建时未指定 `log` 参数的 Informix 数据库，那么对 Informix 的 xa_open 调用将失败，并生成大量 FFST 错误。

检查 Informix 环境变量设置

确保为队列管理器进程 **以及应用程序进程在** 中设置了 Informix 环境变量。 尤其是，在启动队列管理器 **之前** ，请始终设置以下环境变量:

INFORMIXDIR
Informix 产品安装的目录。

- 对于 32 位 UNIX 和 Linux 应用程序，请使用以下命令:

  ```
  export INFORMIXDIR=/opt/informix/32-bit
  ```

- 对于 64 位 UNIX 和 Linux 应用程序，请使用以下命令:

  ```
  export INFORMIXDIR=/opt/informix/64-bit
  ```

- 对于 Windows 应用程序，请使用以下命令:

  ```
  set INFORMIXDIR=c:\informix
  ```

对于具有必须同时支持 32 位和 64 位应用程序的 64 位队列管理器的系统，您需要同时安装 Informix 32 位和 64 位客户机 SDK。 用于创建切换装入文件的样本 makefile `xaswit.mak` 还会设置两个产品安装目录。

INFORMIXSERVER
Informix 服务器的名称。 例如，在 UNIX 和 Linux 系统上，使用:

```
export INFORMIXSERVER=hostname_1
```

在 Windows 系统上，使用:

```
set INFORMIXSERVER=hostname_1
```

ONCONFIG
Informix 服务器配置文件的名称。 例如，在 UNIX 和 Linux 系统上，使用:

```
export ONCONFIG=onconfig.hostname_1
```

在 Windows 系统上，使用:

```
set ONCONFIG=onconfig.hostname_1
```

创建 Informix 交换机装入文件

要创建 Informix 切换装入文件，请使用样本文件 xaswit.mak， IBM MQ 提供该样本文件来为各种数据库产品构建切换装入文件。 在 Windows 系统上，您可以在目录 *MQ_INSTALLATION_PATH*\tools\c\samples\xatm 中找到 xaswit.mak 。 *MQ_INSTALLATION_PATH* 表示 IBM MQ 安装所在的高级目录。 要使用 Microsoft Visual C++ 创建 Informix 切换装入文件，请使用:

```
nmake /f xaswit.mak infswit.dll
```

生成的交换机文件放置在 C:\Program Files\IBM\WebSphere MQ\exits 中。

您可以在目录 *MQ_INSTALLATION_PATH*/samp/xatm 中找到 xaswit.mak 。 *MQ_INSTALLATION_PATH* 表示 IBM MQ 安装所在的高级目录。

编辑 xaswit.mak 以 取消注释 与您正在使用的 Informix 版本相应的行。 然后使用以下命令执行 makefile:

```
make -f xaswit.mak infswit
```

生成的 32 位交换机装入文件放置在 /var/mqm/exits 中。

生成的 64 位交换机装入文件放置在 /var/mqm/exits64 中。

添加 Informix 的资源管理器配置信息

您必须修改队列管理器的配置信息，以将 Informix 声明为全局工作单元中的参与者。 在 第 437 页的『向队列管理器添加配置信息』中更详细地描述了如何以此方式修改队列管理器的配置信息。

- 在 Windows 和 Linux (x86 和 x86-64 平台) 系统上，使用 IBM MQ Explorer。 在队列管理器属性面板中的 XA 资源管理器下指定切换装入文件的详细信息。
- 在所有其他系统上，在队列管理器的 qm.ini 文件的 XAResourceManager 节中指定交换机装入文件的详细信息。

第 445 页的图 88 是 UNIX 样本，显示 qm.ini XAResourceManager 条目，其中要协调的数据库名为 mydbname，此名称在 XAOpenString 中指定:

```
XAResourceManager:
  Name=myinformix
  SwitchFile=infswit
  XAOpenString=DB=mydbname@myinformixserver\;USER=myuser\;PASSWD=mypasswd
  ThreadOfControl=THREAD
```

图 *88: UNIX* 平台 上 *Informix* 的样本 *XAResourceManager* 条目

注: 缺省情况下， UNIX 平台 上的样本 xaswit.mak 将创建一个使用线程 Informix 库的交换机装入文件。 使用这些 Informix 库时，必须确保 ThreadOf 控件设置为 THREAD。 在 第 445 页的图 88 中， qm.ini 文件 XAResourceManager 节属性 ThreadOf 控件设置为 THREAD。 指定 THREAD 时，必须使用 Informix 线程库和 IBM MQ 线程 API 库来构建应用程序。

XAOpenString 属性必须包含数据库名称，后跟 @ 符号，然后后跟 Informix 服务器名称。

要使用非线程 Informix 库，必须确保 qm.ini 文件 XAResourceManager 节属性 ThreadOfControl 设置为 PROCESS。 您还必须对样本 xaswit.mak:

1. 取消注释非线程交换机装入文件的生成。
2. 注释掉线程切换装入文件的生成。

Sybase 配置

Sybase 支持和配置信息。

完成以下步骤:

1. 确保已安装 Sybase XA 库，例如通过安装 XA DTM 选项。
2. 检查环境变量设置。
3. 启用 Sybase XA 支持。
4. 创建 Sybase 交换机装入文件。
5. 添加资源管理器配置信息。

在 WebSphere MQ 详细系统需求 页面上提供了 IBM MQ 支持的 Sybase 级别的当前列表。

检查 Sybase 环境变量设置

确保为队列管理器进程 **以及** 应用程序进程设置了 Sybase 环境变量。 尤其是，在启动队列管理器 **之前** ，请始终设置以下环境变量:

SYBASE
Sybase 产品安装的位置。 例如，在 UNIX 和 Linux 系统上，使用:

```
export SYBASE=/sybase
```

在 Windows 系统上，使用:

```
set SYBASE=c:\sybase
```

SYBASE_OCS
已安装 Sybase 客户机文件的 SYBASE 下的目录。 例如，在 UNIX 和 Linux 系统上，使用:

```
export SYBASE_OCS=OCS-12_0
```

在 Windows 系统上，使用:

```
set SYBASE_OCS=OCS-12_0
```

启用 Sybase XA 支持

在 Sybase XA 配置文件 $SYBASE/$SYBASE_OCS/xa_config 中，为要更新的 Sybase 服务器的每个连接定义逻辑 Resource Manager (LRM)。 第 447 页的图 89 中显示了 $SYBASE/$SYBASE_OCS/xa_config 内容的示例。

```
# The first line must always be a comment

[xa]

  LRM=lrmname
  server=servername
```

图 *89: $SYBASE/ $SYBASE_OCS/xa_config* 的示例内容

创建 Sybase 交换机装入文件

要创建 Sybase 切换装入文件，请使用 IBM MQ 随附的样本文件。 在 Windows 系统上，您可以在目录 `C:\Program Files\IBM\WebSphere MQ\tools\c\samples\xatm` 中找到 xaswit.mak 。 要使用 Microsoft Visual C++ 创建 Sybase 切换装入文件，请使用:

```
nmake /f xaswit.mak sybswit.dll
```

生成的交换机文件放置在 `C:\Program Files\IBM\WebSphere MQ\exits` 中。

您可以在目录 *MQ_INSTALLATION_PATH*`/samp/xatm` 中找到 xaswit.mak 。 *MQ_INSTALLATION_PATH* 表示 IBM MQ 安装所在的高级目录。

编辑 xaswit.mak 以 取消注释 适合于您正在使用的 Sybase 版本的行。 然后使用以下命令执行 makefile:

```
make -f xaswit.mak sybswit
```

生成的 32 位交换机装入文件将放在 `/var/mqm/exits` 中。

生成的 64 位交换机装入文件放置在 `/var/mqm/exits64` 中。

注: V 8.0.0.12 AIX 在 AIX 上，已修改样本 makefile ，如以下示例中所示，以便您可以选择不同的 SYBLINKFLAG64，具体取决于您是使用 Sybase 15 ESD#5 还是更低版本的 Sybase。

```
SYBLINKFLAGS32=-brtl
# The following line is for Sybase 15
#SYBLINKFLAGS64=-brtl
# The following line is for Sybase 16
SYBLINKFLAGS64=-bstatic -bdynamic
```

您需要对 makefile 进行的唯一更改是确保仅取消注释其中一个 SYBLINKFLAGS64 值。 缺省值为 Sybase 16 ，这是要用于 15 #ESD5 及更高版本的值。

生成的任何 XA 切换文件都将链接到 Sybase 的特定发行版，并且不得移动到其他平台。

如果更改了 Sybase 的级别，那么应该重建 XA 切换文件。

为 Sybase 添加资源管理器配置信息

您必须修改队列管理器的配置信息，以将 Sybase 声明为全局工作单元中的参与者。 第 437 页的『向队列管理器添加配置信息』中更详细地描述了如何修改配置信息。

- 在 Windows 和 Linux (x86 和 x86-64 平台) 系统上，使用 IBM MQ Explorer。 在队列管理器属性面板中的 XA 资源管理器下指定切换装入文件的详细信息。
- 在所有其他系统上，在队列管理器的 `qm.ini` 文件的 XAResourceManager 节中指定交换机装入文件的详细信息。

第 448 页的图 90 显示了 UNIX 和 Linux 样本，该样本使用与 Sybase XA 配置文件 $SYBASE/ $SYBASE_OCS/xa_config 中的 *lrmname* LRM 定义相关联的数据库。 如果要记录 XA 函数调用，请包含日志文件名:

```
XAResourceManager:
  Name=mysybase
  SwitchFile=sybswit
  XAOpenString=-Uuser -Ppassword -Nlrmname -L/tmp/sybase.log -Txa
  ThreadOfControl=THREAD
```

图 *90: UNIX* 和 *Linux* 平台上 *Sybase* 的样本 *XAResourceManager* 条目

将多线程程序与 Sybase 配合使用

如果要将多线程程序与包含 Sybase 更新的 IBM MQ 全局工作单元配合使用，那么必须对 ThreadOfControl 参数使用值 THREAD。 还要确保将程序 (和交换机装入文件) 与线程安全 Sybase 库 (_r 版本) 链接。 使用 ThreadOfControl 参数的值 THREAD 在上一个示例中显示。

多个数据库配置

如果要配置队列管理器以便可以在全局工作单元中包含对多个数据库的更新，请为每个数据库添加 XAResourceManager 节。

如果数据库全部由同一数据库管理器管理，那么每个节定义一个单独的数据库。 每个节指定相同的 *SwitchFile*，但 *XAOpenString* 的内容不同，因为它指定要更新的数据库的名称。 例如， 第 448 页的图 91 中显示的节通过 UNIX 和 Linux 系统上的 Db2 数据库 *MQBankDB* 和 *MQFeeDB* 来配置队列管理器。

要点: 不能有多个节指向同一数据库。 此配置在任何情况下都不起作用，如果您尝试此配置，那么它将失败。

您将收到格式为 `when the MQ code makes its second xa_open call in any process in this environment, the database software fails the second xa_open with a -5 error, XAER_INVAL` 的错误。

```
XAResourceManager:
Name=DB2 MQBankDB
SwitchFile=db2swit
XAOpenString=MQBankDB

XAResourceManager:
Name=DB2 MQFeeDB
SwitchFile=db2swit
XAOpenString=MQFeeDB
```

图 *91:* 多个 *Db2* 数据库的样本 *XAResourceManager* 条目

如果要更新的数据库由不同的数据库管理器管理，请为每个数据库管理器添加 XAResourceManager 节。 在这种情况下，每个节指定不同的 *SwitchFile*。 例如，如果 *MQFeeDB* 由 Oracle 而不是 Db2 管理，请在 UNIX 和 Linux 系统上使用以下节:

```
XAResourceManager:
Name=DB2 MQBankDB
SwitchFile=db2swit
XAOpenString=MQBankDB

XAResourceManager:
Name=Oracle MQFeeDB
SwitchFile=oraswit
XAOpenString=Oracle_XA+Acc=P/myuser/mypassword+SesTm=35+LogDir=/tmp/ora.log+DB=MQFeeDB
```

图 *92: Db2* 和 *Oracle* 数据库的样本 *XAResourceManager* 条目

原则上，可以使用单个队列管理器配置的数据库实例数没有限制。

注: 有关支持在全局工作单元中的多个数据库更新中包含 Informix 数据库的信息，请查看产品自述文件。

安全性注意事项

在 XA 模型下运行数据库的注意事项。

提供以下信息仅供参考。 在所有情况下，请参阅数据库管理器随附的文档，以确定在 XA 模型下运行数据库的安全性影响。

应用程序进程表示使用 MQBEGIN 动词启动全局工作单元。 第一个 MQBEGIN 调用是应用程序问题通过在 xa_open 入口点调用其客户机库代码来连接到所有参与数据库。 所有数据库管理器都提供了在其 XAOpenString 中提供用户标识和密码的机制。 这是认证信息流动的唯一时间。

请注意，在 UNIX 和 Linux 平台上，快速路径应用程序在进行 MQI 调用时必须使用有效用户标识 mqm 运行。

丢失与 XA 资源管理器的联系时的注意事项

队列管理器允许数据库管理器不可用。 这意味着您可以独立于数据库服务器启动和停止队列管理器。 恢复联系人后，队列管理器和数据库再同步。 您还可以使用 rsvmqtrn 命令来手动解析不确定的工作单元。

在正常操作中，完成配置步骤后仅需要最少量的管理。 由于队列管理器允许数据库管理器不可用，因此使管理作业更容易执行。 这尤其意味着:

- 队列管理器可以随时启动，而无需首先启动每个数据库管理器。
- 如果其中一个数据库管理器变为不可用，那么队列管理器不需要停止并重新启动。

这允许您独立于数据库服务器启动和停止队列管理器。

每当队列管理器与数据库之间失去联系时，它们都需要在两者再次变为可用时再同步。 再同步是完成涉及该数据库的任何不确定工作单元的过程。 通常，这会自动发生，而无需用户干预。 队列管理器要求数据库提供不确定的工作单元列表。 然后，它指示数据库落实或回滚其中每个不确定的工作单元。

当队列管理器启动时，它将与每个数据库再同步。 当单个数据库变为不可用时，仅当队列管理器通知该数据库再次可用时，才需要对该数据库进行再同步。

当使用 MQBEGIN 启动新的全局工作单元时，队列管理器将自动恢复与先前不可用的数据库的联系。 它通过在数据库客户机库中调用 xa_open 函数来执行此操作。 如果此 xa_open 调用失败，那么 MQBEGIN 将返回完成代码 MQCC_WARNING 和原因码 MQRC_PARTICIPANT_NOT_AVAILABLE。 您可以稍后重试 MQBEGIN 调用。

请勿继续尝试涉及在 MQBEGIN 期间指示失败的数据库更新的全局工作单元。 将不存在与可通过其进行更新的数据库的连接。 唯一的选项是结束程序，或者定期重试 MQBEGIN ，以希望数据库可能再次可用。

或者，可以使用 `rsvmqtrn` 命令来显式解析所有不确定的工作单元。

不确定的工作单元

如果在指示数据库管理器准备之后丢失了与队列管理器的联系，那么数据库可能具有不确定的工作单元。 在数据库服务器从队列管理器接收结果 (落实或回滚) 之前，它需要保留与更新关联的数据库锁定。

由于这些锁定会阻止其他应用程序更新或读取数据库记录，因此需要尽快进行再同步。

如果由于某种原因，您无法等待队列管理器自动与数据库再同步，那么可以使用数据库管理器提供的工具来手动落实或回滚数据库更新。 在 *X/Open* 分布式事务处理*: XA* 规范中，这称为作出 启发式 决策。 仅将其用作最后手段，因为可能会损害数据完整性; 例如，当所有其他参与者已落实其更新时，您可能会错误地回滚数据库更新。

最好重新启动队列管理器，或者在重新启动数据库后使用 `rsvmqtrn` 命令来启动自动再同步。

使用 *dspmqtrn* 命令显示未完成的工作单元

当数据库管理器不可用时，可以使用 **dspmqtrn** 命令来检查涉及该数据库的未完成全局工作单元的状态。

dspmqtrn 命令仅显示其中一个或多个参与者处于不确定状态的工作单元。 参与者正在等待队列管理器的决策，以落实或回滚已准备的更新。

对于其中每个全局工作单元，将在 **dspmqtrn** 的输出中显示每个参与者的状态。 如果工作单元未更新特定资源管理器的资源，那么不会显示该工作单元。

关于不确定的工作单元，据说资源管理器已执行下列其中一项操作:

准备
资源管理器已准备好落实其更新。

已落实
资源管理器已落实其更新。

已回滚
资源管理器已回滚其更新。

参与
资源管理器是参与者，但尚未准备，落实或回滚其更新。

重新启动队列管理器时，它会要求每个具有 XAResourceManager 节的数据库获取其不确定全局工作单元的列表。 如果数据库尚未重新启动，或者在其他情况下不可用，那么队列管理器无法向数据库交付这些工作单元的最终结果。 当数据库再次可用时，会将不确定工作单元的结果第一次传递到数据库。

在这种情况下，报告数据库管理器处于 *prepared* 状态，直到发生再同步为止。

每当 dspmqtrn 命令显示不确定的工作单元时，它都会首先列出可能参与的所有可能的资源管理器。 为这些资源分配了唯一标识 *RMId*，在报告其与不确定工作单元相关的状态时，将使用该标识来代替资源管理器的 名称 。

样本 dspmqtrn 输出 显示发出以下命令的结果:

```
dspmqtrn -m MY_QMGR
```

```
AMQ7107: Resource manager 0 is MQSeries.
AMQ7107: Resource manager 1 is Db2 MQBankDB.
AMQ7107: Resource manager 2 is Db2 MQFeeDB.

AMQ7056: Transaction number 0,1.
    XID: formatID 5067085, gtrid_length 12, bqual_length 4
         gtrid [3291A50600002013746573 74]
         bqual [00000001]
AMQ7105: Resource manager 0 has committed.
AMQ7104: Resource manager 1 has prepared.
AMQ7104: Resource manager 2 has prepared.
```

其中， 事务号 是可与 rsvmqtrn 命令配合使用的事务的标识。 请参阅 AMQ7xxx: IBM MQ 产品消息 以获取更多信息。 *XID* 变量是 *X/Open XA* 规范 的一部分; 有关此规范的最新信息，请参阅: https://publications.opengroup.org/c193。

图 *93:* 样本 *dspmqtrn* 输出

Sample dspmqtrn output 中的输出显示有三个资源管理器与队列管理器相关联。 第一个是资源管理器 0，它是队列管理器本身。 其他两个资源管理器实例是 MQBankDB 和 MQFeeDB Db2 数据库。

此示例仅显示单个不确定工作单元。 将针对所有三个资源管理器发出一条消息，这意味着已对工作单元中的队列管理器和两个 Db2 数据库进行了更新。

对队列管理器 **0** 进行的更新已 落实。 对 Db2 数据库的更新处于 *prepared* 状态，这意味着在调用 Db2 以落实对 *MQBankDB* 和 *MQFeeDB* 数据库的更新之前，必须已变为不可用。

不确定工作单元具有称为 XID (事务标识) 的外部标识。 这是队列管理器提供给 Db2 的数据片段，用于标识其在全局工作单元中的部分。

使用 *rsvmqtrn* 命令解析未完成的工作单元

队列管理器和 Db2 再同步时，未完成的工作单元完成。

第 450 页的图 93 中显示的输出显示尚未将落实决策传递到两个 Db2 数据库的单个不确定工作单元。

要完成此工作单元，队列管理器和 Db2 需要在下次 Db2 变为可用时再同步。 队列管理器使用新工作单元的启动作为重新与 Db2 联系的机会。 或者，您可以指示队列管理器使用 **rsvmqtrn** 命令显式再同步。

在重新启动 Db2 后立即执行此操作，以便尽快释放与不确定工作单元相关联的任何数据库锁定。 使用 -a 选项，它指示队列管理器解析所有不确定的工作单元。 在以下示例中， Db2 已重新启动，因此队列管理器可以解析不确定的工作单元:

```
> rsvmqtrn -m MY_QMGR -a
Any in-doubt transactions have been resolved.
```

混合结果和错误

虽然队列管理器使用两阶段落实协议，但这不会完全消除某些工作单元完成混合结果的可能性。 在此情况下，一些参与者会落实其更新，而一些参与者会回退其更新。

以混合结果完成的工作单元会产生严重影响，因为本应更新为单个工作单元的共享资源不再处于一致状态。

主要在对工作单元做出启发式决策时导致混合结果，而不是允许队列管理器解决不确定的工作单元本身。 此类决策不在队列管理器的控制范围内。

每当队列管理器检测到混合结果时，它都会生成 FFST 信息并在其错误日志中记录故障，并显示以下两条消息之一:

- 如果数据库管理器回滚而不是落实:

  ```
  AMQ7606 A transaction has been committed but one or more resource
          managers have rolled back.
  ```

- 如果数据库管理器落实而不是回滚:

  ```
  AMQ7607 A transaction has been rolled back but one or more resource
          managers have committed.
  ```

进一步的消息标识试探性损坏的数据库。 然后，您负责在本地恢复受影响数据库的一致性。 这是一个复杂的过程，您需要首先隔离错误落实或回滚的更新，然后手动撤销或重做数据库更改。

更改配置信息

在成功启动队列管理器以协调全局工作单元之后，请勿更改任何资源管理器配置信息。

如果需要更改配置信息，您可以随时执行此操作，但这些更改直到重新启动队列管理器之后才会生效。

如果除去数据库的资源管理器配置信息，那么将有效除去队列管理器与该数据库管理器联系的能力。

从不 更改任何资源管理器配置信息中的 名称 属性。 此属性向队列管理器唯一地标识该数据库管理器实例。 如果更改此唯一标识，那么队列管理器假定已除去数据库并添加了全新的实例。 队列管理器仍将未完成的工作单元与旧的 名称相关联，这可能使数据库处于不确定状态。

除去数据库管理器实例

如果需要从配置中永久除去数据库，请确保在重新启动队列管理器之前该数据库未处于不确定状态。

数据库产品提供用于列出不确定事务的命令。 如果存在任何不确定事务，请首先允许队列管理器与数据库再同步。 通过启动队列管理器来执行此操作。 您可以使用 **rsvmqtrn** 命令或数据库自己的命令来查看不确定的工作单元，以验证是否已执行再同步。 一旦您确信已进行再同步，请结束队列管理器并除去数据库的配置信息。

如果未能观察到此过程，那么队列管理器仍会记住涉及该数据库的所有不确定工作单元。 每次重新启动队列管理器时，都会发出一条警告消息 AMQ7623。 如果不再使用队列管理器配置此数据库，请使用 **rsvmqtrn** 命令的 -r 选项指示队列管理器忘记数据库参与其不确定事务。 仅当所有参与者都已完成不确定事务时，队列管理器才会忘记此类事务。

有时可能需要临时除去某些资源管理器配置信息。 在 UNIX 和 Linux 系统上，最好通过注释掉节来实现此目标，以便稍后可以轻松恢复此节。 如果队列管理器每次与特定数据库或数据库管理器联系时都发生错误，那么您可能决定执行此操作。 临时除去相关的资源管理器配置信息允许队列管理器启动涉及所有其他参与者的全局工作单元。 以下是已注释掉的 XAResourceManager 节的示例:

```
# This database has been temporarily removed
#XAResourceManager:
#   Name=mydb2
#   SwitchFile=db2swit
#   XAOpenString=mydbname,myuser,mypassword,toc=t
#   ThreadOfControl=THREAD
```

图 *94:* 已注释掉 *UNIX* 和 *Linux* 系统上的 *XAResourceManager* 节

在 Windows 系统上，使用 IBM MQ Explorer 来删除有关数据库管理器实例的信息。 在恢复时，请非常小心地在 名称 字段中输入正确的名称。 如果错误输入名称，那么可能会迁到不确定问题，如 第 451 页的『更改配置信息』 中所述。

XA 动态注册

XA 规范提供了一种减少事务管理器对资源管理器进行的 xa_* 调用数的方法。 此优化称为 动态注册。

Db2 支持动态注册。 其他数据库可能支持此操作; 请参阅数据库产品的文档以获取详细信息。

为什么动态注册优化有用? 在应用程序中，某些全局工作单元可能包含对数据库表的更新; 其他工作单元可能不包含此类更新。 如果未对数据库的表进行持久更新，那么无需将该数据库包含在 MQCMIT 期间发生的落实协议中。

无论您的数据库是否支持动态注册，您的应用程序都会在 IBM MQ 连接上的第一个 MQBEGIN 调用期间调用 xa_open 。 它还会在后续 MQDISC 调用上调用 xa_close 。 后续 XA 调用的模式取决于数据库是否支持动态注册:

如果您的数据库不支持动态注册 ...

每个全局工作单元都涉及 IBM MQ 代码对数据库客户机库进行的若干 XA 函数调用，而无论您是否在工作单元中对该数据库的表进行了持久更新。 其中包括:

- 应用程序进程中的 xa_start 和 xa_end 。 这些用于声明全局工作单元的开始和结束。
- 来自队列管理器代理进程 amqzlaa0 的 xa_prepare， xa_commit 和 xa_rollback 。 这些用于交付全局工作单元的结果: 落实或回滚决策。

此外，队列管理器代理进程还会在第一个 MQBEGIN 期间调用 xa_open 。

如果您的数据库支持动态注册 ...

IBM MQ 代码仅执行必需的那些 XA 函数调用。 对于 **未** 涉及数据库资源的持久更新的全局工作单元， **不会** 对数据库进行 XA 调用。 对于 **已** 涉及此类持久更新的全局工作单元，调用的目的是:

- 来自应用程序进程的 xa_end ，用于声明全局工作单元的结束。
- 来自队列管理器代理进程 amqzlaa0 的 xa_prepare， xa_commit 和 xa_rollback 。 这些用于交付全局工作单元的结果: 落实或回滚决策。

要使动态注册生效，数据库必须能够在执行持久更新时告知 IBM MQ 它希望包含在当前全局工作单元中。 IBM MQ 提供了用于此目的的 ax_reg 函数。

在应用程序进程中运行的数据库的客户机代码将找到 ax_reg 函数并对其进行调用，以 动态注册 它在当前全局工作单元中执行持久工作的事实。 作为对此 ax_reg 调用的响应， IBM MQ 记录了数据库已参与的记录。 如果这是此 IBM MQ 连接上的第一个 ax_reg 调用，那么队列管理器代理进程将调用 xa_open。

数据库客户机代码在进程中运行时 (例如，在 SQL UPDATE 调用期间或数据库的客户机 API 中的任何调用) 发出此 ax_reg 调用。

错误条件

在 XA 动态注册中，可能会在队列管理器中发生混淆故障。

一个常见示例是，如果在启动队列管理器之前忘记正确设置数据库环境变量，那么队列管理器对 xa_open 的调用将失败。 不能使用全局工作单元。

要避免此情况，请确保在启动队列管理器之前设置了相关环境变量。 查看数据库产品的文档以及 第 439 页的『配置 Db2』， 第 442 页的『配置 Oracle』和 第 446 页的『Sybase 配置』中提供的建议。

对于所有数据库产品，队列管理器在队列管理器启动时调用 xa_open 一次，作为恢复会话的一部分 (如 第 449 页的『丢失与 XA 资源管理器的联系时的注意事项』 中所述)。 如果未正确设置数据库环境变量，但这不会导致队列管理器无法启动，那么此 xa_open 调用将失败。 这是因为数据库客户机库使用相同的 xa_open 错误代码来指示数据库服务器不可用。 IBM MQ 不会将此视为严重错误，因为队列管理器必须能够开始在涉及该数据库的全局工作单元之外继续处理数据。

在 IBM MQ 连接上的第一个 MQBEGIN 期间 (如果未使用动态注册) 或数据库客户机代码对 IBM MQ-provided ax_reg 函数的调用期间 (如果正在使用动态注册)，将从队列管理器进行对 xa_open 的后续调用。

任何错误条件 (有时为 FFST 报告) 的 **计时** 取决于您是否正在使用动态注册:

- 如果您正在使用动态注册，那么 MQBEGIN 调用可能会成功，但 SQL UPDATE (或类似) 数据库调用将失败。
- 如果您未使用动态注册，那么 MQBEGIN 调用将失败。

确保在应用程序和队列管理器进程中正确设置环境变量。

汇总 *XA* 调用

以下是由于控制全局工作单元的各种 MQI 调用而对数据库客户机库中的 XA 函数进行的调用的列表。 这不是 XA 规范中描述的协议的完整描述; 它作为简要概述提供。

请注意， xa_start 和 xa_end 调用始终由应用程序进程中的 IBM MQ 代码调用，而 xa_prepare, xa_commit 和 xa_rollback 始终从队列管理器代理程序进程 amqzlaa0 调用。

此表中显示的 xa_open 和 xa_close 调用都是从应用程序进程进行的。 队列管理器代理进程在 第 452 页的『错误条件』中描述的情况下调用 xa_open 。

表 *33: XA* 函数调用摘要

MQI 调用	**使用动态注册进行的 XA 调用**	**在未进行动态注册的情况下进行的 XA 调用**
第一个 MQBEGIN	`xa_open`	`xa_open` `xa_start`
后续 MQBEGIN	无 XA 调用	`xa_start`

MQCMIT (在当前全局工作单元期间调用 **而不调用** ax_reg)	无 XA 调用	`xa_end`
		`xa_prepare`
		`xa_commit`
		`xa_rollback`

MQCMIT (**，在当前全局工作单元期间调用** ax_reg)	`xa_end`
	`xa_prepare`
	`xa_commit`
	`xa_rollback`
		不适用。 不会以非动态方式调用 ax_reg。

MQBACK (在当前全局工作单元期间调用 **而不调用** ax_reg)	无 XA 调用	`xa_end` `xa_rollback`

表 *33: XA* 函数调用摘要 *(继续)*		
MQI 调用	**使用动态注册进行的 XA 调用**	**在未进行动态注册的情况下进行的 XA 调用**
MQBACK（，**在当前全局工作单元期间调用** ax_reg）	xa_end xa_rollback	不适用。 不会以非动态方式调用 ax_reg。
MQDISC，其中首先调用了 MQCMIT 或 MQBACK。 如果不是，那么首先在 MQDISC 期间执行 MQCMIT 处理。	xa_close	xa_close

注意:

1. 对于 MQCMIT，如果 xa_prepare 成功，那么将调用 xa_commit。 否则，将调用 xa_rollback。

方案 2: 其他软件提供协调

在方案 2 中，外部事务管理器协调全局工作单元，在事务管理器的 API 控制下启动并落实这些工作单元。 MQBEGIN、MQCMIT 和 MQBACK 动词均不可用。

本部分描述了此场景，包括:

- 第 454 页的『外部同步点协调』
- 第 456 页的『使用 CICS』
- 第 460 页的『使用 Microsoft Transaction Server (COM +)』

HP Integrity NonStop Server 的 IBM MQ 客户机可以使用 HP NonStop 事务管理设施 (TMF) 来协调全局工作单元。 有关更多信息，请参阅 使用 HP NonStop TMF。

外部同步点协调

全局工作单元还可以由符合 X/Open XA 的外部事务管理器进行协调。 在此， IBM MQ 队列管理器参与但不协调工作单元。

由外部事务管理器协调的全局工作单元中的控制流如下所示:

1. 应用程序告知外部同步点协调程序 (例如， TXSeries) 它想要启动事务。
2. 同步点协调程序告知已知资源管理器 (例如， IBM MQ) 有关当前事务的信息。
3. 应用程序向与当前事务关联的资源管理器发出调用。 例如，应用程序可以向 IBM MQ 发出 MQGET 调用。
4. 应用程序向外部同步点协调程序发出落实或回退请求。
5. 同步点协调程序通过向每个资源管理器 (通常使用两阶段落实协议) 发出相应的调用来完成事务。

在 WebSphere MQ 详细系统需求上定义了可以为 IBM MQ 参与的事务提供两阶段落实过程的受支持外部同步点协调程序级别。

本节的其余部分描述了如何启用外部工作单元。

IBM MQ XA 开关结构

参与外部协调工作单元的每个资源管理器都必须提供 XA 切换结构。 此结构定义资源管理器的功能以及要由同步点协调程序调用的功能。

IBM MQ 提供了此结构的两个版本:

- 用于静态 XA 资源管理的 *MQRMIXASwitch*
- *MQRMIXASwitchDynamic* 用于动态 XA 资源管理

请参阅事务管理器文档以确定是使用静态还是动态资源管理接口。 只要事务管理器支持，我们就建议您使用动态 XA 资源管理。

某些 64 位事务管理器将 XA 规范中的 *long* 类型视为 64 位，而某些事务管理器将其视为 32 位。 IBM MQ 支持两种模型:

- 如果事务管理器为 32 位，或者事务管理器为 64 位，但将 *long* 类型视为 32 位，请使用 第 455 页的表 34 中列出的交换机装入文件。
- 如果事务管理器是 64 位的，并且将 *long* 类型视为 64 位，请使用 第 455 页的表 35 中列出的交换机装入文件。

第 455 页的表 36 中提供了将 *long* 类型视为 64 位的已知 64 位事务管理器的列表。 如果您不确定事务管理器使用的模型，请查阅事务管理器文档。

表 *34: XA* 切换装入文件名

平台	**交换机装入文件名 (服务器)**	**交换机装入文件名 (扩展事务客户机)**
Windows	*mqmxa.dll*	*mqcxa.dll*
AIX (非线程)	*libmqmxa.a*	*libmqcxa.a*
AIX (线程化)	*libmqmxa_r.a*	*libmqcxa_r.a*
HP-UX (非线程)	*libmqmxa.so*	*libmqcxa.so*
HP-UX (线程化)	*libmqmxa_r.so*	*libmqcxa_r.so*
Linux (非线程)	*libmqmxa.so*	*libmqcxa.so*
Linux (线程化)	*libmqmxa_r.so*	*libmqcxa_r.so*
Solaris	*libmqmxa.so*	*libmqcxa.so*

表 *35:* 备用 *64* 位 *XA* 切换装入文件名

平台	**交换机装入文件名 (服务器)**	**交换机装入文件名 (扩展事务客户机)**
AIX (非线程)	*libmqmxa64.a*	*libmqcxa64.a*
AIX (线程化)	*libmqmxa64_r.a*	*libmqcxa64_r.a*
HP-UX (非线程)	*libmqmxa64.so*	*libmqcxa64.so*
HP-UX (线程化)	*libmqmxa64_r.so*	*libmqcxa64_r.so*
Linux (非线程)	*libmqmxa64.so*	*libmqcxa64.so*
Linux (线程化)	*libmqmxa64_r.so*	*libmqcxa64_r.so*
Solaris	*libmqmxa64.so*	*libmqcxa64.so*

表 *36:* 需要备用 *64* 位交换机装入文件的 *64* 位事务管理器

事务管理器
Tuxedo

某些外部同步点协调程序 (非 CICS) 要求参与工作单元的每个资源管理器在 XA 开关结构的名称字段中提供其名称。 IBM MQ 资源管理器名称为 MQSeries_XA_RMI。

同步点协调程序定义 IBM MQ XA 切换结构如何链接到同步点协调程序。 第 456 页的『使用 CICS』 中提供了有关将 IBM MQ XA 交换机结构与 CICS 链接的信息。 有关将 IBM MQ XA 开关结构与其他符合 XA 的同步点协调程序链接的信息，请参阅随这些产品提供的文档。

以下注意事项适用于将 IBM MQ 与所有符合 XA 的同步点协调程序配合使用:

- 期望事务管理器库代码 (作为应用程序员已调用的 API 的一部分运行) 在调用 MQCONN 之前的某个时间点将 **xa_open** 调用到 IBM MQ 中。

  必须在执行 MQCONN 调用的同一线程上执行 **xa_open** 调用。 此需求的原因是 XA 规范要求使用线程来暗示上下文。

  请注意，这是样本程序 `amqstxsx.c` 中采用的方法。 此样本程序假定 **xa_open** 调用在其函数 `tpopen` 中从事务管理器的库代码进入 IBM MQ。

  如果在同一线程上没有 **xa_open** 调用，那么在 MQCONN 调用之前， IBM MQ 队列管理器连接将不会与 XA 上下文相关联。

  有关更多信息，请参阅 MQCTL。
- 同步点协调程序在任何 xa_open 调用上传递的 xa_info 结构包含 IBM MQ 队列管理器的名称。 该名称采用与传递到 `MQCONN` 调用的队列管理器名称相同的格式。 如果在 xa_open 调用上传递的名称为空，那么将使用缺省队列管理器。

  或者， xa_info 结构可以包含 *TPM* 和 *AXLIB* 参数的值。 *TPM* 参数指定正在使用的事务管理器。 有效值为 CICS， TUXEDO 和 ENCINA。 *AXLIB* 参数指定包含事务管理器的 ax_reg 和 ax_unreg 函数的库的名称。 有关这些参数的更多信息，请参阅 配置扩展事务客户机。 如果 xa_info 结构包含其中任一参数，那么除非正在使用缺省队列管理器，否则将在 *QMNAME* 参数中指定队列管理器名称。
- 一次只能有一个队列管理器参与由外部同步点协调程序实例协调的事务。 同步点协调程序有效连接到队列管理器，并且遵循一次仅支持一个连接的规则。
- 所有包含对外部同步点协调程序的调用的应用程序都只能连接到参与由外部协调程序管理的事务的队列管理器 (因为它们已有效地连接到该队列管理器)。 但是，此类应用程序必须发出 `MQCONN` 调用以获取连接句柄，并在它们退出之前发出 `MQDISC` 调用。
- 具有由外部同步点协调程序协调的资源更新的队列管理器必须在外部同步点协调程序之前启动。 同样，同步点协调程序必须在队列管理器之前结束。
- 如果外部同步点协调程序异常终止，请停止并重新启动队列管理器 **前** ，重新启动同步点协调程序，以确保正确解决在发生故障时未落实的任何消息传递操作。

使用 CICS

CICS 是 TXSeries 的其中一个元素。

在以下位置定义了符合 XA 的 TXSeries 版本 (并使用两阶段落实过程): WebSphere MQ 详细系统需求

IBM MQ 还支持其他事务管理器。 请参阅 WebSphere MQ 详细系统需求 以获取受支持软件的当前列表。

两阶段落实过程的需求

将 CICS 两阶段落实过程与 IBM MQ 配合使用时，两阶段落实过程的需求。 这些要求不适用于 z/OS。

请注意以下需求:

- IBM MQ 和 CICS 必须位于同一物理机器上。
- IBM MQ 在 IBM MQ MQI client 上不支持 CICS 。
- 必须先启动队列管理器，并在 XAD 资源定义节中指定其名称 **，然后再** 尝试启动 CICS。 如果将 IBM MQ 的 XAD 资源定义节添加到 CICS 区域，那么未能执行此操作将阻止您启动 CICS 。
- 一次只能从单个 CICS 区域访问一个 IBM MQ 队列管理器。
- CICS 事务必须先发出 `MQCONN` 请求，然后才能访问 IBM MQ 资源。 `MQCONN` 调用必须指定在 CICS 区域的 XAD 资源定义节的 XAOpen 条目上指定的 IBM MQ 队列管理器的名称。 如果此条目为空，那么 `MQCONN` 请求必须指定缺省队列管理器。
- 访问 IBM MQ 资源的 CICS 事务必须在返回到 CICS 之前从该事务发出 `MQDISC` 调用。 未能执行此操作可能意味着 CICS 应用程序服务器仍处于连接状态，从而使队列处于打开状态。 此外，如果未安装任务终止出口 (请参阅 第 459 页的『样本任务终止出口』)，那么 CICS 应用程序服务器稍后可能会异常结束，可能在后续事务期间异常结束。
- 必须确保 CICS 用户标识 (cics) 是 mqm 组的成员，以便 CICS 代码具有调用 IBM MQ 的权限。

对于在 CICS 环境中运行的事务，队列管理器会调整其授权和确定上下文的方法，如下所示:

- 队列管理器将查询 CICS 用于运行事务的用户标识。 这是对象权限管理器检查的用户标识，用于上下文信息。
- 在消息上下文中，应用程序类型为 MQAT_CICS。
- 将从 CICS 事务名称复制上下文中的应用程序名称。

常规 *XA* 支持

常规 XA 在 IBM i 上不受支持。 提供了 XA 切换装入模块，使您能够将 CICS 与 UNIX 和 Linux 系统上的 IBM MQ 链接。 此外，还提供了样本源代码文件，使您能够为其他事务消息开发 XA 交换机。

提供的交换机装入模块的名称为:

表 *37: CICS* 应用程序的基本代码 *:XA* 初始化例程

C（源代码）	**C (exec)-将下列其中一项添加到您的 XAD.Stanza**
amqzscix.c	amqzsc - TXSeries for AIX， V 5.1, amqzsc - TXSeries for HP-UX V5.1 amqzsc - TXSeries for Sun SolarisV 5.1
amqzscin.c	mqmc4swi - TXSeries for WindowsV 5.1

构建用于 *TXSeries for Multiplatforms* 的库
构建用于 TXSeries for Multiplatforms 的库时，请使用此信息。

预构建的交换机装入文件 是可与 CICS 程序配合使用的共享库 (在 Windows 系统上称为 *DLL*) ，这些程序需要使用 XA 协议的 2 阶段落实事务。 这些预构建库的名称在表 CICS 应用程序的基本代码 :XA 初始化例程 中。 以下目录中还提供了样本源代码:

表 *38: Windows* 上的安装目录， *UNIX* 和 *Linux* 操作系统

平台	**目录**	**源文件**
UNIX 和 Linux	*MQ_INSTALLATION_PATH*/samp/	amqzscix.c
Windows	*MQ_INSTALLATION_PATH*\Tools\C\Samples	amqzscin.c

其中 *MQ_INSTALLATION_PATH* 是 IBM MQ 的安装目录。

要从样本源构建交换机装入文件，请遵循适合于您的操作系统的指示信息:

AIX

发出以下命令:

```
export MQM_HOME=/usr/mqm
echo "amqzscix" > tmp.exp
xlc_r $MQM_HOME/samp/amqzscix.c -I/usr/lpp/cics/include -I$MQM_HOME/inc -e amqzscix -bE:tmp.exp -bM:SRE
    -o amqzsc /usr/lpp/cics/lib/regxa_swxa.o -L$MQM_HOME/lib -L/usr/lpp/cics/lib -lcicsrt -lEncina
    -lEncServer -lpthreads -lsarpc -lmqmcics_r -lmqmxa_r -lmqzi_r -lmqmcs_r
rm tmp.exp
```

Solaris
发出以下命令:

```
/opt/SUNWspro/bin/cc -s -l/opt/encina/include amqzscix.c -G -o amqzscix -e
    CICS_XA_Init -L MQ_INSTALLATION_PATH/lib -L/opt/encina/lib
    -L/opt/dcelocal/lib /opt/cics/lib/reqxa_swxa.o
    -lmqmcics -lmqmxa -lmqzi -lmqmcs -lmqmzse -lcicsrt -lEncina -lEncSfs -ldce
```

HP-UX
发出以下命令:

```
cc -c -s -I/opt/encina/include MQ_INSTALLATION_PATH/samp/amqzscix.c -Aa +z -o amqzscix.o ld -b
    -o amqzscix amqzscix.o /opt/cics/lib/regxa_swxa.o +e CICS_XA_Init \
    -L MQ_INSTALLATION_PATH/lib -L/opt/encina/lib -L/opt/cics/lib
    -lmqmxa_r -lmqzi_r -lmqmcs_r -lmqmzse -ldbm -lc -lm
```

Linux 平台
发出以下命令:

```
gcc -m32 -shared -fPIC -o amqzscix amqzscix.c
    \-IMQ_INSTALLATION_PATH/inc -I CICS_INSTALLATION_PATH/include
    \-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib
    \-Wl,-rpath=/usr/lib -Wl,-rpath-link,/usr/lib -Wl,--no-undefined
    -Wl,--allow-shlib-undefined \-L CICS_LIB_PATH/regxa_swxa.o \-lpthread -ldl -lc
    -shared -lmqzi_r -lmqmxa_r -lmqmcics_r -ldl -lc
```

Windows
请按照以下步骤操作:

1. 使用 cl 命令通过至少编译以下变量来构建 amqzscin.obj :

   ```
   cl.exe -c -I EncinaPath\include -I MQ_INSTALLATION_PATH\include -Gz -LD amqzscin.c
   ```

2. 创建名为 `mqmc1415.def` 的模块定义文件，其中包含以下行:

   ```
   LIBRARY MQMC4SWI
   EXPORTS
   CICS_XA_Init
   ```

3. 使用 **lib** 命令通过至少使用以下选项来构建导出文件和导入库:

   ```
   lib -def:mqmc4swi.def -out:mqmc4swi.lib
   ```

   如果 lib 命令成功，那么还会构建 `mqmc4swi.exp` 文件。

4. 使用链接命令通过至少使用以下选项来构建 `mqmc4swi.dll` :

   ```
   link.exe -dll -nod -out:mqmc4swi.dll
       amqzscin.obj CicsPath\lib\regxa_swxa.obj
       mqmc4swi.exp mqmcics4.lib
       CicsPath\lib\libcicsrt.lib
       DcePath\lib\libdce.lib DcePath\lib\pthreads.lib
       EncinaPath\lib\libEncina.lib
       EncinaPath\lib\libEncServer.lib
       msvcrt.lib kernel32.lib
   ```

IBM MQ XA 支持和 *Tuxedo*
IBM MQ 在 Windows 上， UNIX 和 Linux 系统可以在 xa_start 中无限期地阻止 Tuxedo 协调的 XA 应用程序。

仅当 Tuxedo 在单个全局事务中协调的两个或多个进程尝试使用同一事务分支标识 (XID) 访问 IBM MQ 时，才会发生此情况。 如果 Tuxedo 为全局事务中的每个进程提供不同的 XID 以用于 IBM MQ，那么不会发生此情况。

为避免此问题，请在 Tuxedo 服务器组中的单个全局事务标识 (gtrid) 下配置 Tuxedo 中访问 IBM MQ 的每个应用程序。 同一服务器组中的进程在代表单个 gtrid 访问资源管理器时使用相同的 XID，因此容易在 IBM MQ 中的 xa_start 中受到阻止。 不同服务器组中的进程在访问资源管理器时使用单独的 XID，因此不必在 IBM MQ 中序列化其事务工作。

启用 *CICS* 两阶段落实过程

要使 CICS 能够使用两阶段落实过程来协调包含 MQI 调用的事务，请将 CICS XAD 资源定义节条目添加到 CICS 区域。 请注意，本主题不适用于 z/OS。

以下是为 IBM MQ for Windows 添加 XAD 节条目的示例，其中 <Drive> 是安装了 IBM MQ 的驱动器 (例如 D:)。

```
cicsadd -cxad -r<cics_region> \
     ResourceDescription="MQM XA Product Description" \
     SwitchLoadFile="<Drive>:\Program Files\IBM\WebSphere MQ\bin\mqmc4swi.dll" \
     XAOpen=<queue_manager_name>
```

对于扩展事务客户机，请使用切换装入文件 mqcc4swi.dll。

以下是为 IBM MQ for UNIX 和 Linux 系统添加 XAD 节条目的示例，其中 *MQ_INSTALLATION_PATH* 表示安装了 IBM MQ 的高级目录:

```
cicsadd -cxad -r<cics_region> \
      ResourceDescription="MQM XA Product Description" \
      SwitchLoadFile=" MQ_INSTALLATION_PATH/lib/amqzsc" \
      XAOpen=<queue_manager_name>
```

对于扩展事务客户机，请使用切换装入文件 amqczsc。

有关使用 **cicsadd** 命令的信息，请参阅适用于您的平台的 *CICS Administration Reference* 或 *CICS Administration Guide* 。

对 IBM MQ 的调用可以包含在 CICS 事务中，并且将根据 CICS 的指示落实或回滚 IBM MQ 资源。 此支持不可用于客户机应用程序。

您 **必须** 从 CICS 事务发出 MQCONN 以访问 IBM MQ 资源，然后在退出时发出相应的 MQDISC 。

启用 *CICS* 用户出口

CICS 用户出口 点 (通常称为 用户出口) 是 CICS 模块中的一个位置，在此位置 CICS 可以将控制权转移到您已编写的程序 (用户出口 程序)，在此位置 CICS 可以在出口程序完成其工作时恢复控制权。

在使用 CICS 用户出口之前，请阅读适用于您平台的 *CICS Administration Guide* 。

样本任务终止出口

IBM MQ 为 CICS 任务终止出口提供样本源代码。

样本源代码位于以下目录中:

表 *39: CICS* 任务终止出口

平台	目录	源文件
UNIX 和 Linux 系统	*MQ_INSTALLATION_PATH*/samp	amqzscgx.c
Windows	*MQ_INSTALLATION_PATH*\Tools\C\Samples	amqzscgn.c

MQ_INSTALLATION_PATH 表示 IBM MQ 安装所在的高级目录。

样本任务终止出口的构建指示信息包含在每个源文件中的注释中。

此出口由 CICS 在正常和异常任务终止时 (在执行任何同步点之后) 调用。 在出口程序中不允许可恢复的工作。

这些函数仅在 CICS 版本支持 XA 接口的 IBM MQ 和 CICS 上下文中使用。 CICS 将这些库称为 `programs` 或 `user exits`。

CICS 具有多个用户出口，并且 amqzscgx (如果使用) 在 CICS 上定义并启用为 `Task termination user exit (UE014015)`，即出口号 15。

当 CICS 调用任务终止出口时， CICS 已通知 IBM MQ 该任务的终止状态，并且 IBM MQ 已执行相应的操作 (落实或回滚)。 出口执行的所有操作都是发出 MQDISC 以进行清除。

安装和配置 CICS 系统以使用任务终止出口的一个目的是保护系统免受发生故障的应用程序代码的某些后果的影响。 例如，如果 CICS 事务在未首先调用 MQDISC 的情况下异常结束，并且未安装任务终止出口，那么您可能会看到 (大约 10 秒内) CICS 区域的后续不可恢复故障。 这是因为将不会发布在 cicsas 进程中运行的 IBM MQ 运行状况线程，也不会为其提供清除和返回的时间。 症状可能是 cicsas 进程立即结束，已将 FFST 报告写入 /var/mqm/errors 或 Windows 上的等效位置。

使用 Microsoft Transaction Server (COM +)

COM + (Microsoft Transaction Server) 旨在帮助用户在典型的中间层服务器中运行业务逻辑应用程序。

请参阅 只能与 窗口 上的主安装配合使用的功能部件 以获取重要信息。

COM + 将工作划分为 活动，这些活动通常是业务逻辑的简短独立块，例如 将资金从帐户 *A* 转移到帐户 *B*。 COM + 在很大程度上依赖于对象方向，尤其是 COM; 松散地， COM + 活动由 COM (业务) 对象表示。

COM + 是操作系统的集成部分。

COM + 为业务对象管理员提供三种服务，消除了业务对象程序员的许多烦恼:

- 事务管理
- 安全性
- 资源池

通常，将 COM + 与前端代码一起使用，而前端代码是 COM + 中保存的对象和后端服务 (例如数据库) 的 COM + ，并在 COM + 业务对象和后端之间进行 IBM MQ 桥接。

前端代码可以是独立程序，也可以是由 Microsoft Internet Information Server (IIS) 托管的活动服务器页面 (ASP)。 前端代码可以与 COM + 及其业务对象在同一台计算机上，通过 COM 进行连接。 或者，前端代码可以在不同的计算机上，通过 DCOM 进行连接。 您可以使用不同的客户机在不同的情况下访问同一个 COM + 业务对象。

后端代码可以与 COM + 及其业务对象位于同一台计算机上，也可以位于通过任何 IBM MQ 支持的协议进行连接的另一台计算机上。

即将到期的全球工作单元

可以将队列管理器配置为在预先配置的不活动时间间隔后使全局工作单元到期。

要启用此行为，请设置以下环境变量:

- *AMQ_TRANSACTION_EXPIRY_RESCAN* =<rescan interval in milliseconds>
- *AMQ_XA_TRANSACTION_*到期 =<timeout interval in milliseconds>

注意: 环境变量仅影响 XA 规范的表 6-4 中处于 空闲 状态的事务，该规范可从位于 OPEN 组的 出版物获取。

即，未在任何应用程序线程上关联但外部事务管理器软件尚未调用 **xa_prepare** 函数调用的事务。

外部事务管理器仅保留已准备，已落实或已回滚的事务的日志。 如果外部事务管理器由于任何原因而关闭，那么在返回时，它会将已准备，已落实和已回滚的事务驱动至完成，但尚未准备的任何活动事务将成为孤立事务。 要避免此情况，请设置 *AMQ_XA_TRANSACTION_*到期 ，以允许应用程序执行 MQI 事务性 API 调用与完成事务之间的预期时间间隔，在其他资源管理器上执行事务性工作。

要确保在 *AMQ_XA_TRANSACTION_*到期 到期后及时清除，请将 *AMQ_TRANSACTION_EXPIRY_RESCAN* 值设置为低于 *AMQ_XA_TRANSACTION_*到期 时间间隔的值，以便在 *AMQ_XA_TRANSACTION_*到期 时间间隔内多次发生重新扫描。

恢复单元处置

IBM MQ for z/OS 提供了恢复处置单元。 此功能部件允许您配置在连接到同一队列共享组 (QSG) 中的另一个队列管理器时，是否可以驱动 2 阶段落实事务的第二阶段 (例如，在恢复期间)。

IBM MQ for z/OS V7.0.1 和更高版本支持恢复单元处置。

恢复单元处置

恢复处置单元与应用程序的连接以及随后启动的任何事务相关。 有两个可能的恢复单元。

- GROUP 恢复单元处置标识事务应用程序在逻辑上已连接到队列共享组，并且与任何特定队列管理器都没有亲缘关系。 在连接到 QSG 中的任何队列管理器时，可以查询并解决它启动的任何 2 阶段落实事务，这些事务已完成落实进程的 phase-1 (即，它们处于不确定状态)。 在恢复方案中，这意味着事务协调程序不必重新连接到可能不可用的同一队列管理器。
- QMGR 恢复处置单元标识应用程序与它所连接的队列管理器具有直接亲缘关系，并且它启动的任何事务也具有此处置。

  在恢复方案中，事务协调程序必须重新连接到同一队列管理器以查询和解决任何不确定事务，而不考虑该队列管理器是否属于队列共享组。

z/OS 有关如何实现此功能的详细信息，请参阅 恢复单元处置 。

安全方案

演示将安全性应用于不同配置的一组方案。

以下子主题中描述了可用的安全方案:

相关信息

z/OS 在 z/OS 上设置安全性

安全方案: z/OS 上的两个队列管理器

在此场景中，应用程序使用 **MQPUT1** 调用将消息放入队列管理器 QM1 上的队列。 然后，使用 TCP 和 LU 6.2 通道将某些消息转发到 QM2 上的队列。 TCP 通道可以使用 SSL，也可以不使用 SSL。 应用程序可以是批处理应用程序或 CICS 应用程序，并且使用 MQPMO_SET_ALL_CONTEXT 选项放置消息。

第 462 页的图 95 对此作了说明。

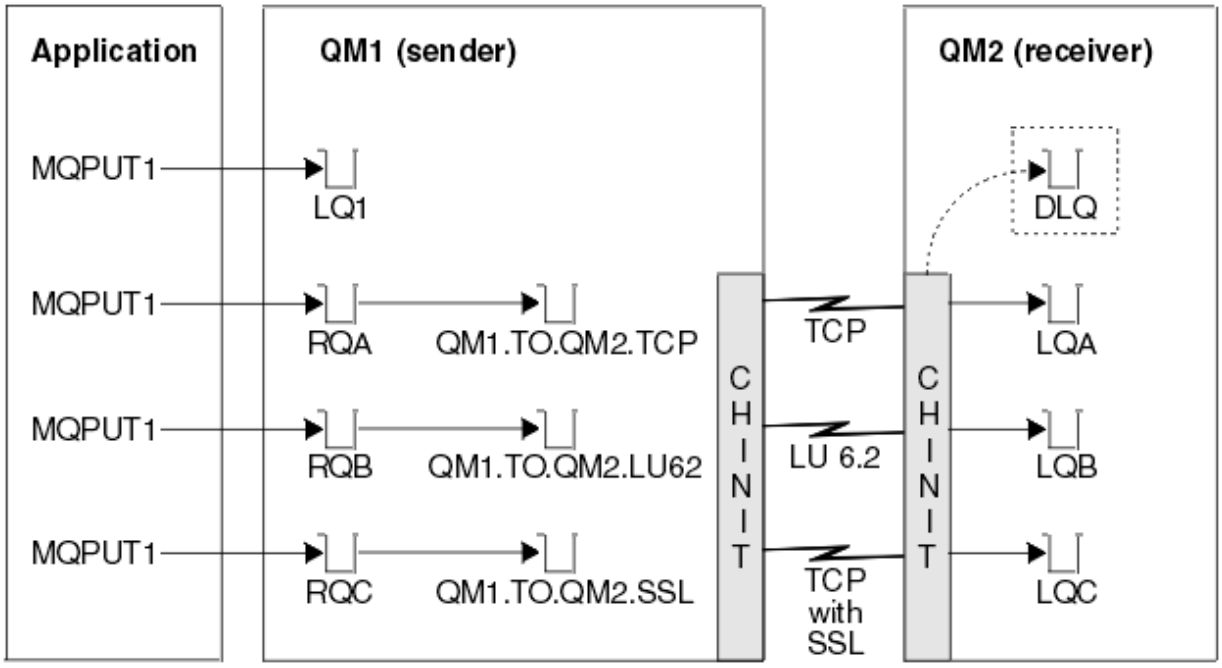


图 *95:* 示例安全方案

对队列管理器进行了以下假定:

- 所有必需的 IBM MQ 定义都已预定义或已通过队列管理器启动时处理的 CSQINP2 数据集进行。

  如果没有这些对象，那么您需要对定义这些对象所需的命令具有相应的访问权限。
- 在启动队列管理器和通道启动程序之前，已定义所需的所有 RACF 概要文件，并且已授予相应的访问权限。

  如果没有这些概要文件，那么您需要适当的权限来发出定义所有所需概要文件所需的 RACF 命令，并授予这些概要文件相应的访问权限。 您还需要适当的权限来发出 MQSC 安全命令，以开始使用新的安全概要文件。
- 已创建所需的所有数字证书并将其连接到密钥环。 作为 SSL 握手的一部分由 QM1 发送的数字证书由 QM2 系统上的 RACF 识别，因为它也安装在该 RACF 概要文件中，或者因为存在匹配的证书名称文件 (CNF) 过滤器。

相关信息

z/OS 在 z/OS 上设置安全性

双队列管理器方案的安全开关设置

切换设置和 RACF 概要文件。

为这两个队列管理器设置了以下安全开关:

- 子系统安全性开启
- 队列安全性开启
- 备用用户安全性
- 上下文安全性开启
- 流程安全性关闭
- 名称列表安全性关闭
- 主题安全性关闭
- 连接安全性开启

- 命令安全性开启
- 命令资源安全性开启

在 MQADMIN 类中定义了以下概要文件以关闭进程，名称列表和主题安全性:

```
QM1.NO.PROCESS.CHECKS
QM1.NO.NLIST.CHECKS
QM1.NO.TOPIC.CHECKS
QM2.NO.PROCESS.CHECKS
QM2.NO.NLIST.CHECKS
QM2.NO.TOPIC.CHECKS
```

双队列管理器方案中的队列管理器 QM1

QM1 的队列和通道。

在队列管理器 QM1:

LQ1
本地队列。

RQA
具有以下属性的远程队列定义:

- RNAME (LQA)
- RQMNAME (QM2)
- XMITQ (QM1.TO.QM2.TCP)

RQB
具有以下属性的远程队列定义:

- RNAME (LQB)
- RQMNAME (QM2)
- XMITQ (QM1.TO.QM2.LU62)

RQC
具有以下属性的远程队列定义:

- RNAME (LQC)
- RQMNAME (QM2)
- XMITQ (QM1.TO.QM2.SSL)

QM1.TO.QM2.TCP
传输队列。

QM1.TO.QM2.LU62
传输队列。

QM1.TO.QM2.SSL
传输队列。

在 QM1:

QM1.TO.QM2.TCP
具有以下属性的发送方通道定义:

- CHLTYPE (SDR)
- TRPTYPE (TCP)
- XMITQ (QM1.TO.QM2.TCP)
- CONNAME (QM2TCP)

QM1.TO.QM2.LU62
具有以下属性的发送方通道定义:

- CHLTYPE (SDR)
- TRPTYPE (LU62)
- XMITQ (QM1.TO.QM2.LU62)
- CONNAME (QM2LU62)

(请参阅 z/OS 上通道启动程序的安全性注意事项，以获取有关设置 APPC 安全性的信息。)

QM1.TO.QM2.SSL
具有以下属性的发送方通道定义:

- CHLTYPE (SDR)
- TRPTYPE (TCP)
- XMITQ (QM1.TO.QM2.SSL)
- CONNAME (QM2TCP)
- SSLCIPH (TLS_RSA_WITH_AES_128_CBC_SHA256)

双队列管理器方案中的队列管理器 QM2

QM2 的队列和通道。

已在队列管理器 QM2:

LQA
本地队列。

LQB
本地队列。

LQC
本地队列。

DLQ
用作死信队列的本地队列。

已在 QM2:

QM1.TO.QM2.TCP
具有以下属性的接收方通道定义:

- CHLTYPE (RCVR)
- TRPTYPE (TCP)
- PUTAUT (CTX)
- MCAUSER (MCATCP)

QM1.TO.QM2.LU62
具有以下属性的接收方通道定义:

- CHLTYPE (RCVR)
- TRPTYPE (LU62)
- PUTAUT (CTX)
- MCAUSER (MCALU62)

(请参阅 z/OS 上通道启动程序的安全性注意事项，以获取有关设置 APPC 安全性的信息。)

QM1.TO.QM2.SSL
具有以下属性的接收方通道定义:

- CHLTYPE (RCVR)
- TRPTYPE (TCP)
- PUTAUT (CTX)
- MCAUSER (MCASSL)

- SSLCIPH (TLS_RSA_WITH_AES_128_CBC_SHA256)

双队列管理器方案中使用的用户标识

方案中用户标识的说明。

使用以下用户标识:

BatchId
批处理应用程序 (作业或 TSO 标识)

MSGUSR
MQMD 中的 *UserIdentifier* (上下文用户标识)

MOVER1
QM1 通道启动程序地址空间用户标识

MOVER2
QM2 通道启动程序地址空间用户标识

MCATCP
在没有 SSL 接收方通道定义的 TCP/IP 上指定了 MCAUSER

MCALU62
在 LU 6.2 接收方通道定义上指定的 MCAUSER

MCASSL
在具有 SSL 接收方通道定义的 TCP/IP 上指定的 MCAUSER

CICSAD1
CICS 地址空间标识

CICSTX1
CICS 任务用户标识

CERTID
RACF 与流证书关联的用户标识。

双队列管理器方案所需的安全概要文件和访问权

双队列管理器方案的批处理或 CICS 实现的安全概要文件和访问权。

下表显示了使双队列管理器方案工作所需的安全概要文件。 还需要其他安全概要文件，具体取决于您是在执行方案的批处理还是 CICS 实现。 有关更多信息，请参阅 第 466 页的『批处理应用程序所需的安全概要文件』 和 第 468 页的『CICS 应用程序所需的安全概要文件』。

表 *40:* 示例方案的安全概要文件.

此表中的四列显示了双队列管理器方案的类，概要文件，用户标识和访问权。

类	概要文件	用户标识	访问
MQCONN	QM1.CHIN	MOVER1	读取
MQADMIN	QM1.RESLEVEL	BATCHID CICSAD1 MOVER1	无
MQADMIN	QM1.CONTEXT.**	MOVER1	控制
MQQUEUE	QM1.SYSTEM.COMMAND.INPUT	MOVER1	更新
MQQUEUE	QM1.SYSTEM.CHANNEL.SYNCQ	MOVER1	更新
MQQUEUE	QM1.SYSTEM.CHANNEL.INITQ	MOVER1	更新
MQQUEUE	QM1.SYSTEM.COMMAND.REPLY.MODEL	MOVER1	更新
MQQUEUE	QM1.SYSTEM.ADMIN.CHANNEL.EVENT	MOVER1	更新

表 *40:* 示例方案的安全概要文件.

此表中的四列显示了双队列管理器方案的类，概要文件，用户标识和访问权。

(继续)

类	概要文件	用户标识	访问
MQQUEUE	QM1.QM1.TO.QM2.TCP	MOVER1	更改
MQQUEUE	QM1.QM1.TO.QM2.LU62	MOVER1	更改
MQQUEUE	QM1.QM1.TO.QM2.SSL	MOVER1	更改
MQCONN	QM2.CHIN	MOVER2	读取
MQADMIN	QM2.RESLEVEL	MOVER2	无
MQADMIN	QM2.CONTEXT.**	MOVER2	控制
MQQUEUE	QM2.SYSTEM.COMMAND.INPUT	MOVER2	更新
MQQUEUE	QM2.SYSTEM.CHANNEL.SYNCQ	MOVER2	更新
MQQUEUE	QM2.SYSTEM.CHANNEL.INITQ	MOVER2	更新
MQQUEUE	QM2.SYSTEM.COMMAND.REPLY.MODEL	MOVER2	更新
MQQUEUE	QM2.SYSTEM.ADMIN.CHANNEL.EVENT	MOVER2	更新
MQQUEUE	QM2.DLQ	MOVER2	更新

批处理应用程序所需的安全概要文件

双队列管理器方案的批处理实现所需的其他安全概要文件。

批处理应用程序在 QM1 上的用户标识 BATCHID 下运行。 它连接到队列管理器 QM1 并将消息放入以下队列:

- LQ1
- RQA
- RQB
- RQC

它使用 MQPMO_SET_ALL_CONTEXT 选项。 在消息描述符 (MQMD) 的 *UserIdentifier* 字段中找到的备用用户标识为 MSGUSR。

队列管理器 QM1:

表 *41:* 队列管理器 *QM1* 上批处理应用程序的样本安全概要文件

类	概要文件	用户标识	访问
MQCONN	QM1.BATCH	BatchId	读取
MQADMIN	QM1.CONTEXT.**	BatchId	控制
MQQUEUE	QM1.LQ1	BatchId	更新
MQQUEUE	QM1.RQA	BatchId	更新
MQQUEUE	QM1.RQB	BatchId	更新
MQQUEUE	QM1.RQC	BatchId	更新

对于放入队列管理器 QM1 上队列 RQA 的消息 (对于不使用 SSL 的 TCP/IP 通道) , 队列管理器 QM2 上需要以下概要文件:

表 *42:* 使用 *TCP/IP* 而不是 *SSL* 的队列管理器 *QM2* 的样本安全概要文件			
类	**概要文件**	**用户标识**	**访问**
MQADMIN	QM2.ALTERNATE.USER.MSGUSR	MCATCP MOVER2	更新
MQADMIN	QM2.CONTEXT.**	MCATCP MOVER2	控制
MQQUEUE	QM2.LQA	MOVER2 MSGUSR	更新
MQQUEUE	QM2.DLQ	MOVER2 MSGUSR	更新

注意:

1. 在消息的 MQMD 中传递的用户标识将用作队列管理器 QM2 上 MQPUT1 的用户标识，因为接收方通道是使用 PUTAUT (CTX) 和 MCAUSER (MCATCP) 定义的。
2. 接收方通道定义的 MCAUSER 字段设置为 MCATCP; 除了通道启动程序地址空间用户标识外，还将使用此用户标识对备用用户标识和上下文概要文件执行的检查。
3. 消息描述符 (MQMD) 中的 MOVER2 用户标识和 *UserIdentifier* 用于针对队列进行资源检查。
4. MOVER2 和 MSGUSR 用户标识都需要访问死信队列，以便无法放入目标队列的消息可以在该队列中发送。
5. 由于 RESLEVEL 设置为 NONE ，因此将在执行的所有三项检查中检查两个用户标识。

对于放入队列管理器 QM1 (对于 LU 6.2 通道) 上的队列 RQB 的消息，队列管理器 QM2 上需要以下概要文件:

表 *43:* 使用 *LU 6.2* 的队列管理器 *QM2* 的样本安全概要文件			
类	**概要文件**	**用户标识**	**访问**
MQADMIN	QM2.ALTERNATE.USER.MSGUSR	MCALU62 MOVER1	更新
MQADMIN	QM2.CONTEXT.**	MCALU62 MOVER1	控制
MQQUEUE	QM2.LQB	MOVER1 MSGUSR	更新
MQQUEUE	QM2.DLQ	MOVER1 MSGUSR	更新

注意:

1. 在消息的 MQMD 中传递的用户标识将用作队列管理器 QM2 上 MQPUT1 的用户标识，因为接收方通道是使用 PUTAUT (CTX) 和 MCAUSER (MCALU62) 定义的。
2. MCA 用户标识设置为接收方通道定义 (MCALU62) 的 MCAUSER 字段的值。
3. 由于 LU 6.2 支持通道的通信系统上的安全性，因此从网络接收的用户标识将用作通道用户标识 (MOVER1)。
4. 由于 RESLEVEL 设置为 NONE ，因此将在执行的所有三项检查中检查两个用户标识。
5. MCALU62 和 MOVER1 用于对备用用户标识和上下文概要文件执行的检查， MSGUSR 和 MOVER1 用于对队列概要文件执行的检查。
6. MOVER1 和 MSGUSR 用户标识都需要访问死信队列，以便无法放入目标队列的消息可以在那里发送。

对于放入队列管理器 QM1 (对于使用 SSL 的 TCP/IP 通道) 上的队列 RQC 的消息，队列管理器 QM2 上需要以下概要文件:

表 *44:* 使用 *TCP/IP* 和 *SSL* 的队列管理器 *QM2* 的样本安全概要文件			
类	**概要文件**	**用户标识**	**访问**
MQADMIN	QM2.ALTERNATE.USER.MSGUSR	MCASSL CERTID	更新
MQADMIN	QM2.CONTEXT.**	MCASSL CERTID	控制

表 *44:* 使用 *TCP/IP* 和 *SSL* 的队列管理器 *QM2* 的样本安全概要文件 *(*继续*)*			
类	**概要文件**	**用户标识**	**访问**
MQQUEUE	QM2.LQC	CERTID MSGUSR	更新
MQQUEUE	QM2.DLQ	CERTID MSGUSR	更新

注意:

1. 在消息的 MQMD 中传递的用户标识将用作队列管理器 QM2 上 MQPUT1 的用户标识，因为接收方通道是使用 PUTAUT (CTX) 和 MCAUSER (MCASSL) 定义的。
2. MCA 用户标识设置为接收方通道定义 (MCASSL) 的 MCAUSER 字段的值。
3. 由于通道作为 SSL 握手的一部分从 QM1 流出的证书可能安装在 QM2 的系统上，或者可能与 QM2 的系统上的证书名称过滤器匹配，因此在该匹配期间找到的用户标识将用作通道用户标识 (CERTID)。
4. 由于 RESLEVEL 设置为 NONE ，因此将在执行的所有三项检查中检查两个用户标识。
5. MCASSL 和 CERTID 用于对备用用户标识和上下文概要文件执行的检查， MSGUSR 和 MOVER1 用于对队列概要文件执行的检查。
6. CERTID 和 MSGUSR 用户标识都需要访问死信队列，以便无法放入目标队列的消息可以在那里发送。

CICS 应用程序所需的安全概要文件

双队列管理器方案的 CICS 实现所需的其他安全概要文件。

CICS 应用程序使用 CICS 地址空间用户标识 CICSAD1 和 CICS 任务用户标识 CICSTX1。 队列管理器 QM1 上所需的安全概要文件与批处理应用程序所需的安全概要文件不同。 队列管理器 QM2 上所需的概要文件与批处理应用程序的概要文件相同。

队列管理器 QM1:

表 *45:* 队列管理器 *QM1* 上 *CICS* 应用程序的样本安全概要文件			
类	**概要文件**	**用户标识**	**访问**
MQCONN	QM1.CICS	CICSAD1	读取
MQADMIN	QM1.CONTEXT.**	CICSAD1 CICSTX1	控制
MQQUEUE	QM1.LQ1	CICSAD1 CICSTX1	更新
MQQUEUE	QM1.RQA	CICSAD1 CICSTX1	更新
MQQUEUE	QM1.RQB	CICSAD1 CICSTX1	更新

安全方案: z/OS 上的队列共享组

在此场景中，应用程序使用 **MQPUT1** 调用将消息放入队列管理器 QM1 上的队列。 然后，使用 TCP 和 LU 6.2 通道将某些消息转发到 QM2 上的队列。 应用程序是批处理应用程序，使用 MQPMO_SET_ALL_CONTEXT 选项放置消息。

第 462 页的图 95 对此作了说明。

对队列管理器进行了以下假定:

- 所有必需的 IBM MQ 定义都已预定义或已通过队列管理器启动时处理的 CSQINP2 数据集进行。

  如果没有这些对象，那么您需要对定义这些对象所需的命令具有相应的访问权限。
- 在启动队列管理器和通道启动程序之前，已定义所需的所有 RACF 概要文件，并且已授予相应的访问权限。

如果没有这些概要文件，那么您需要适当的权限来发出定义所有所需概要文件所需的 RACF 命令，并授予这些概要文件相应的访问权限。 您还需要适当的权限来发出 MQSC 安全命令，以开始使用新的安全概要文件。

相关信息

z/OS 在 z/OS 上设置安全性

队列共享组方案的安全开关设置

切换设置和 RACF 概要文件。

为队列共享组设置了以下安全开关:

- 子系统安全性开启
- 队列共享组安全性开启
- 队列管理器安全性关闭
- 队列安全性开启
- 备用用户安全性
- 上下文安全性开启
- 流程安全性关闭
- 名称列表安全性关闭
- 主题安全性关闭
- 连接安全性开启
- 命令安全性开启
- 命令资源安全性开启

在 MQADMIN 类中定义了以下概要文件以关闭进程，名称列表，主题和队列管理器级别的安全性:

```
QSGA.NO.PROCESS.CHECKS
QSGA.NO.NLIST.CHECKS
QSGA.NO.TOPIC.CHECKS
QSGA.NO.QMGR.CHECKS
```

队列共享组方案中的队列管理器 QM1

QM1 的队列和通道。

在队列管理器 QM1:

LQ1
本地队列。

RQA
具有以下属性的远程队列定义:

- RNAME (LQA)
- RQMNAME (QM2)
- XMITQ (QM1.TO.QM2.TCP)

RQB
具有以下属性的远程队列定义:

- RNAME (LQB)
- RQMNAME (QM2)
- XMITQ (QM1.TO.QM2.LU62)

QM1.TO.QM2.TCP
传输队列。

QM1.TO.QM2.LU62
传输队列。

在 QM1:

QM1.TO.QM2.TCP
具有以下属性的发送方通道定义:

- CHLTYPE (SDR)
- TRPTYPE (TCP)
- XMITQ (QM1.TO.QM2.TCP)
- CONNAME (QM2TCP)

QM1.TO.QM2.LU62
具有以下属性的发送方通道定义:

- CHLTYPE (SDR)
- TRPTYPE (LU62)
- XMITQ (QM1.TO.QM2.LU62)
- CONNAME (QM2LU62)

(请参阅 z/OS 上通道启动程序的安全性注意事项，以获取有关设置 APPC 安全性的信息。)

队列共享组方案中的队列管理器 QM2

QM2 的队列和通道。

已在队列管理器 QM2:

LQA
本地队列。

LQB
本地队列。

DLQ
用作死信队列的本地队列。

已在 QM2:

QM1.TO.QM2.TCP
具有以下属性的接收方通道定义:

- CHLTYPE (RCVR)
- TRPTYPE (TCP)
- PUTAUT (CTX)
- MCAUSER (MCATCP)

QM1.TO.QM2.LU62
具有以下属性的接收方通道定义:

- CHLTYPE (RCVR)
- TRPTYPE (LU62)
- PUTAUT (CTX)
- MCAUSER (MCALU62)

(请参阅 z/OS 上通道启动程序的安全性注意事项，以获取有关设置 APPC 安全性的信息。)

队列共享组方案中使用的用户标识

方案中用户标识的说明。

使用以下用户标识:

BatchId
批处理应用程序 (作业或 TSO 标识)

MSGUSR
MQMD 中的 *UserIdentifier* (上下文用户标识)

MOVER1
QM1 通道启动程序地址空间用户标识

MOVER2
QM2 通道启动程序地址空间用户标识

MCATCP
在 TCP/IP 接收方通道定义上指定了 MCAUSER

MCALU62
在 LU 6.2 接收方通道定义上指定的 MCAUSER

队列共享组方案所需的安全概要文件和访问权

队列共享组方案的批处理或 CICS 实现的安全概要文件和访问权。

下表显示了使队列共享组方案工作所需的安全概要文件。 此场景的批处理实现还需要 第 472 页的『批处理应用程序所需的安全概要文件』中描述的其他安全概要文件。

表 *46:* 示例方案的安全概要文件.

此表中的四列显示队列共享组方案的类，概要文件，用户标识和访问权。

类	概要文件	用户标识	访问
MQCONN	QSGA.CHIN	MOVER1 MOVER2	读取
MQADMIN	QSGA.RESLEVEL	BATCHID MOVER1 MOVER2	无
MQADMIN	QSGA.CONTEXT.**	MOVER1 MOVER2	控制
MQQUEUE	QSGA.SYSTEM.COMMAND.INPUT	MOVER1 MOVER2	更新
MQQUEUE	QSGA.SYSTEM.CHANNEL.SYNCQ	MOVER1 移动者	更新
MQQUEUE	QSGA.SYSTEM.CHANNEL.INITQ	MOVER1 MOVER2	更新
MQQUEUE	QSGA.SYSTEM.COMMAND.REPLY.MODEL	MOVER1 MOVER2	更新
MQQUEUE	QSGA.SYSTEM.ADMIN.CHANNEL.EVENT	MOVER1 MOVER2	更新
MQQUEUE	QSGA.SYSTEM.QSG.CHANNEL.SYNCQ	MOVER1 MOVER2	更新
MQQUEUE	QSGA.SYSTEM.QSG.TRANSMIT.QUEUE	MOVER1 MOVER2	更新

表 *46:* 示例方案的安全概要文件.

此表中的四列显示队列共享组方案的类，概要文件，用户标识和访问权。

(继续)

类	概要文件	用户标识	访问
MQQUEUE	QSGA.QM1.TO.QM2.TCP	MOVER1	更改
MQQUEUE	QSGA.QM1.TO.QM2.LU62	MOVER1	更改
MQQUEUE	QSGA.DLQ	MOVER2	更新

批处理应用程序所需的安全概要文件

队列共享组方案的批处理实现所需的其他安全概要文件。

批处理应用程序在 QM1 上的用户标识 BATCHID 下运行。 它连接到队列管理器 QM1 并将消息放入以下队列:

- LQ1
- RQA
- RQB

它使用 MQPMO_SET_ALL_CONTEXT 选项。 在消息描述符 (MQMD) 的 *UserIdentifier* 字段中找到的用户标识为 MSGUSR。

队列管理器 QM1:

表 *47:* 队列管理器 *QM1* 上批处理应用程序的样本安全概要文件

类	概要文件	用户标识	访问
MQCONN	QSGA.BATCH	BatchId	读取
MQADMIN	QSGA.CONTEXT.**	BatchId	控制
MQQUEUE	QSGA.LQ1	BatchId	更新
MQQUEUE	QSGA.RQA	BatchId	更新
MQQUEUE	QSGA.RQB	BatchId	更新

对于放入队列管理器 QM1 (对于 TCP/IP 通道) 上队列 RQA 的消息，队列管理器 QM2 上需要以下概要文件:

表 *48:* 使用 *TCP/IP* 的队列管理器 *QM2* 的样本安全概要文件

类	概要文件	用户标识	访问
MQADMIN	QSGA.ALTERNATE.USER.MSGUSR	MCATCP MOVER2	更新
MQADMIN	QSGA.CONTEXT.**	MCATCP MOVER2	控制
MQQUEUE	QSGA.LQA	MOVER2 MSGUSR	更新
MQQUEUE	QSGA.DLQ	MOVER2 MSGUSR	更新

注意:

1. 在消息的 MQMD 中传递的用户标识将用作队列管理器 QM2 上 MQPUT1 的用户标识，因为接收方通道是使用 PUTAUT (CTX) 和 MCAUSER (MCATCP) 定义的。
2. 接收方通道定义的 MCAUSER 字段设置为 MCATCP; 除了通道启动程序地址空间用户标识外，还将使用此用户标识对备用用户标识和上下文概要文件执行的检查。
3. 消息描述符 (MQMD) 中的 MOVER2 用户标识和 *UserIdentifier* 用于针对队列进行资源检查。

4. MOVER2 和 MSGUSR 用户标识都需要访问死信队列，以便无法放入目标队列的消息可以在该队列中发送。
5. 由于 RESLEVEL 设置为 NONE，因此将在执行的所有三项检查中检查两个用户标识。

对于放入队列管理器 QM1 (对于 LU 6.2 通道) 上的队列 RQB 的消息，队列管理器 QM2 上需要以下概要文件:

表 *49:* 使用 *LU 6.2* 的队列管理器 *QM2* 的样本安全概要文件

类	概要文件	用户标识	访问
MQADMIN	QSGA.ALTERNATE.USER.MSGUSR	MCALU62 MOVER1	更新
MQADMIN	QSGA.CONTEXT.**	MCALU62 MOVER1	控制
MQQUEUE	QSGA.LQB	MOVER1 MSGUSR	更新
MQQUEUE	QSGA.DLQ	MOVER1 MSGUSR	更新

注意:

1. 在消息的 MQMD 中传递的用户标识将用作队列管理器 QM2 上 MQPUT1 的用户标识，因为接收方通道是使用 PUTAUT (CTX) 和 MCAUSER (MCALU62) 定义的。
2. MCA 用户标识设置为接收方通道定义 (MCALU62) 的 MCAUSER 字段的值。
3. 由于 LU 6.2 支持通道的通信系统上的安全性，因此从网络接收的用户标识将用作通道用户标识 (MOVER1)。
4. 由于 RESLEVEL 设置为 NONE，因此将在执行的所有三项检查中检查两个用户标识。
5. MCALU62 和 MOVER1 用于对备用用户标识和上下文概要文件执行的检查， MSGUSR 和 MOVER1 用于对队列概要文件执行的检查。
6. MOVER1 和 MSGUSR 用户标识都需要访问死信队列，以便无法放入目标队列的消息可以在那里发送。

使用 SSL 或 TLS 连接两个队列管理器

使用 SSL 或 TLS 加密安全协议的安全通信涉及设置通信通道和管理将用于认证的数字证书。

要设置 SSL 或 TLS 安装，必须定义通道以使用 SSL 或 TLS。 您还必须获取和管理数字证书。 在测试系统上，可以使用自签名证书或本地认证中心 (CA) 颁发的证书。 在生产系统上，请勿使用自签名证书。

有关创建和管理证书的完整信息，请参阅以下主题:

- IBM i 在 IBM i 上使用 SSL 或 TLS
- distributed 在 UNIX， Linux 和 Windows 系统上使用 SSL 或 TLS
- z/OS 在 z/OS 上使用 SSL 或 TLS

此主题集合介绍了设置 SSL 通信所涉及的任务，并提供了有关完成这些任务的逐步指导。

您可能还想要测试 SSL 或 TLS 客户机认证，这是协议的可选部分。 在 SSL 或 TLS 握手期间， SSL 或 TLS 客户机始终从服务器获取并验证数字证书。 通过 IBM MQ 实现， SSL 或 TLS 服务器始终从客户机请求证书。

注意:

1. 在此上下文中， SSL 客户机是指启动握手的连接。
2. z/OS 当 z/OS 队列管理器充当 SSL 客户机的角色时，队列管理器仅发送证书。

仅当 SSL 或 TLS 客户机可以找到具有匹配标签的证书时，才会发送证书。 请参阅 数字证书标签 以获取详细信息。

SSL 或 TLS 服务器始终验证客户机证书 (如果发送了客户机证书)。 如果客户机未发送证书，那么仅当使用设置为 REQUIRED 的 **SSLCAUTH** 参数或设置为 **SSLPEER** 参数的值来定义充当 SSL 或 TLS 服务器的通道结束时，认证才会失败。 有关匿名连接队列管理器的更多信息，即，当 SSL 或 TLS 客户机未发送证书时，请参阅 第 478 页的『使用单向认证连接两个队列管理器』。

使用自签名证书对两个队列管理器进行相互认证

遵循以下样本指示信息，以使用自签名 SSL 或 TLS 证书在两个队列管理器之间实现相互认证。

关于此任务

方案:

- 您有两个需要安全通信的队列管理器 QM1 和 QM2。 您需要在 QM1 和 QM2 之间执行相互认证。
- 您已决定使用自签名证书测试安全通信。

生成的配置如下所示:

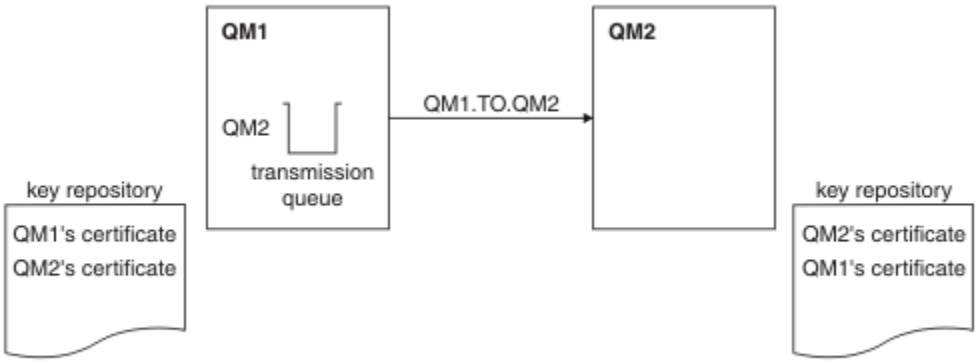


图 *96:* 由此任务生成的配置

在 第 474 页的图 96 中， QM1 的密钥存储库包含 QM1 的证书和来自 QM2 的公用证书。 QM2 的密钥存储库包含 QM2 的证书以及来自 QM1 的公用证书。

过程

1. 根据操作系统，在每个队列管理器上准备密钥存储库:
   - 在 UNIX， Linux 和 窗口 系统上。
   - z/OS 在 z/OS 系统上。
2. 为每个队列管理器创建自签名证书:
   - 在 UNIX， Linux 和 窗口 系统上。
   - z/OS 在 z/OS 系统上。
3. 抽取每个证书的副本:
   - 在 UNIX， Linux 和 窗口 系统上。
   - z/OS 在 z/OS 系统上。
4. 使用诸如 FTP z/OS 之类的实用程序将 QM1 证书的公用部分传输到 QM2 系统，反之亦然，如 ../com.ibm.mq.sec.doc/q013210_.dita 中所述。
5. 将合作伙伴证书添加到每个队列管理器的密钥存储库:
   - 在 UNIX， Linux 和 窗口 系统上。
   - z/OS 在 z/OS 系统上。

6. 在 QM1 上，通过发出类似以下示例的命令来定义发送方通道和关联的传输队列:

```
DEFINE CHANNEL(QM1.TO.QM2) CHLTYPE(SDR) TRPTYPE(TCP) CONNAME(QM1.MACH.COM) XMITQ(QM2)
SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA) DESCR('Sender channel using SSL from QM1 to QM2')

DEFINE QLOCAL(QM2) USAGE(XMITQ)
```

   此示例使用 CipherSpec TLS_RSA。 通道两端的 CipherSpecs 必须相同。
7. 在 QM2 上，通过发出类似于以下示例的命令来定义接收方通道:

```
DEFINE CHANNEL(QM1.TO.QM2) CHLTYPE(RCVR) TRPTYPE(TCP) SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA)
SSLCAUTH(REQUIRED) DESCR('Receiver channel using SSL from QM1 to QM2')
```

   通道必须与您在步骤 6 中定义的发送方通道同名，并使用相同的 CipherSpec。
8. 启动通道 z/OS，如 启动发送方通道 中所述。

结果

创建密钥存储库和通道，如 第 474 页的图 96 中所述

下一步做什么

使用 DISPLAY 命令检查任务是否已成功完成。 如果任务成功，那么生成的输出类似于以下示例中显示的输出。

从队列管理器 QM1，输入以下命令:

```
DISPLAY CHS(QM1.TO.QM2) SSLPEER SSLCERTI
```

生成的输出类似于以下示例:

```
DISPLAY CHSTATUS(QM1.TO.QM2) SSLPEER SSLCERTI
     4 : DISPLAY CHSTATUS(QM1.TO.QM2) SSLPEER SSLCERTI
AMQ8417: Display Channel Status details.
   CHANNEL(QM1.TO.QM2)                     CHLTYPE(SDR)
   CONNAME(9.20.25.40)                     CURRENT
   RQMNAME(QM2)
   SSLCERTI("CN=QM2,OU=WebSphere MQ Development,O=IBM,ST=Hampshire,C=UK")
   SSLPEER("SERIALNUMBER=4C:D0:49:D5:02:5E:02,CN=QM2,OU=WebSphere MQ
Development,O=IBM,ST=Hampshire,C=UK")
   STATUS(RUNNING)                         SUBSTATE(MQGET)
   XMITQ(QM2)
```

从队列管理器 QM2 中，输入以下命令:

```
DISPLAY CHS(QM1.TO.QM2) SSLPEER SSLCERTI
```

生成的输出类似于以下示例:

```
DISPLAY CHSTATUS(QM1.TO.QM2) SSLPEER SSLCERTI
     5 : DISPLAY CHSTATUS(QM1.TO.QM2) SSLPEER SSLCERTI
AMQ8417: Display Channel Status details.
   CHANNEL(QM2.TO.QM1)                     CHLTYPE(RCVR)
   CONNAME(9.20.35.92)                     CURRENT
   RQMNAME(QM1)
   SSLCERTI("CN=QM1,OU=WebSphere MQ Development,O=IBM,ST=Hampshire,C=UK")
   SSLPEER("SERIALNUMBER=4C:D0:49:D5:02:5F:38,CN=QM1,OU=WebSphere MQ
Development,O=IBM,ST=Hampshire,C=UK")
   STATUS(RUNNING)                         SUBSTATE(RECEIVE)
   XMITQ( )
```

在每种情况下， SSLPEER 的值必须与步骤 2 中创建的伙伴证书中的 DN 的值匹配。 颁发者名称与同级名称匹配，因为证书是自签名的。

SSLPEER 是可选的。 如果指定了此值，那么必须设置其值，以便允许使用伙伴证书 (在步骤 2 中创建) 中的 DN。 有关使用 SSLPEER 的更多信息，请参阅 IBM MQ SSLPEER 值的规则。

使用 CA 签名的证书对两个队列管理器进行相互认证

遵循以下样本指示信息以使用 CA 签署的 SSL 或 TLS 证书在两个队列管理器之间实现相互认证。

关于此任务

方案:

- 您有两个名为 QMA 和 QMB 的队列管理器，它们需要安全地进行通信。 您需要在 QMA 和 QMB 之间执行相互认证。
- 将来您计划在生产环境中使用此网络，因此您决定从一开始就使用 CA 签署的证书。

生成的配置如下所示:

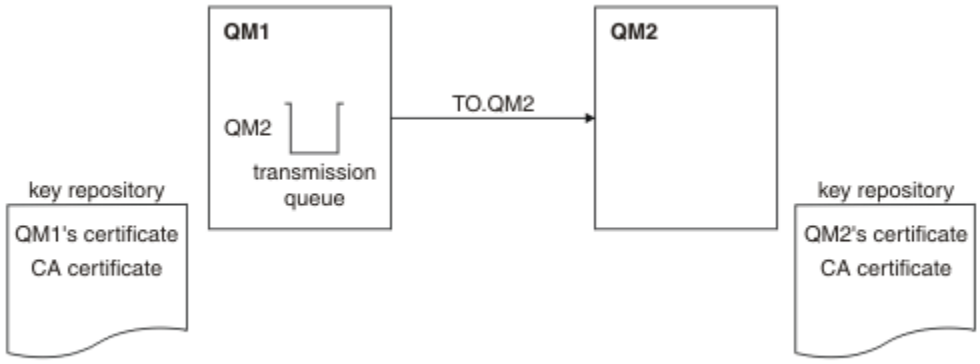


图 97: 由此任务生成的配置

在 第 476 页的图 97 中， QMA 的密钥存储库包含 QMA 的证书和 CA 证书。 QMB 的密钥存储库包含 QMB 的证书和 CA 证书。 在此示例中， QMA 的证书和 QMB 的证书都由同一 CA 发放。 如果 QMA 的证书和 QMB 的证书是由不同的 CA 发放的，那么 QMA 和 QMB 的密钥存储库必须同时包含两个 CA 证书。

过程

1. 根据操作系统，在每个队列管理器上准备密钥存储库:
   - IBM i 在 IBM i 系统上。
   - 在 UNIX， Linux 和 窗口 系统上。
   - z/OS 在 z/OS 系统上。
2. 请求每个队列管理器的 CA 签名证书。
   您可以对两个队列管理器使用不同的 CA。
   - IBM i 在 IBM i 系统上。
   - 在 UNIX， Linux 和 窗口 系统上。
   - z/OS 在 z/OS 系统上。
3. 将认证中心证书添加到每个队列管理器的密钥存储库:
   如果队列管理器使用不同的认证中心，那么必须将每个认证中心的 CA 证书添加到两个密钥存储库。
   - IBM i 请勿在 IBM i 系统上执行此步骤。
   - 在 UNIX， Linux 和 窗口 系统上。

- z/OS 在 z/OS 系统上。

4. 将 CA 签名的证书接收到每个队列管理器的密钥存储库中:
   - IBM i 在 IBM i 系统上。
   - 在 UNIX， Linux 和 窗口 系统上。
   - z/OS 在 z/OS 系统上。
5. 在 QMA 上，通过发出类似于以下示例的命令来定义发送方通道和关联的传输队列:

```
DEFINE CHANNEL(TO.QMB) CHLTYPE(SDR) TRPTYPE(TCP)
CONNAME(QMB.MACH.COM) XMITQ(QMB) SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA)
DESCR('Sender channel using TLS from QMA to QMB')

DEFINE QLOCAL(QMB) USAGE(XMITQ)
```

   此示例使用 CipherSpec RC4_MD5。 通道两端的 CipherSpecs 必须相同。
6. 在 QMB 上，通过发出类似于以下示例的命令来定义接收方通道:

```
DEFINE CHANNEL(TO.QMB) CHLTYPE(RCVR) TRPTYPE(TCP)
SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA) SSLCAUTH(REQUIRED)
DESCR('Receiver channel using TLS to QMB')
```

   通道必须与您在步骤 6 中定义的发送方通道同名，并使用相同的 CipherSpec。
7. 启动通道:
   - IBM i 在 IBM i 系统上。
   - 在 UNIX， Linux 和 窗口 系统上。
   - z/OS 在 z/OS 系统上。

结果

将创建密钥存储库和通道，如 第 476 页的图 97 中所述。

下一步做什么

使用 DISPLAY 命令检查任务是否已成功完成。 如果任务成功，那么生成的输出类似于以下示例中显示的输出。

从队列管理器 QMA ，输入以下命令:

```
DISPLAY CHS(TO.QMB) SSLPEER SSLCERTI
```

生成的输出类似于以下示例:

```
DISPLAY CHSTATUS(TO.QMB) SSLPEER SSLCERTI
     4 : DISPLAY CHSTATUS(TO.QMB) SSLPEER SSLCERTI
AMQ8417: Display Channel Status details.
   CHANNEL(TO.QMB)                         CHLTYPE(SDR)
   CONNAME(9.20.25.40)                     CURRENT
   RQMNAME(QMB)
   SSLCERTI("CN=WebSphere MQ CA,OU=WebSphere MQ Devt,O=IBM,ST=Hampshire,C=UK")
   SSLPEER("SERIALNUMBER=4C:D0:49:D5:02:5F:38,CN=QMB,OU=WebSphere MQ
Development,O=IBM,ST=Hampshire,C=UK")
   STATUS(RUNNING)                         SUBSTATE(MQGET)
   XMITQ(QMB)
```

从队列管理器 QMB 中，输入以下命令:

```
DISPLAY CHS(TO.QMB) SSLPEER SSLCERTI
```

生成的输出类似于以下示例:

```
DISPLAY CHSTATUS(TO.QMB) SSLPEER SSLCERTI
     5 : DISPLAY CHSTATUS(TO.QMB) SSLPEER SSLCERTI
AMQ8417: Display Channel Status details.
   CHANNEL(TO.QMB)                         CHLTYPE(RCVR)
   CONNAME(9.20.35.92)                     CURRENT
   RQMNAME(QMA)
   SSLCERTI("CN=WebSphere MQ CA,OU=WebSphere MQ Devt,O=IBM,ST=Hampshire,C=UK")
   SSLPEER("SERIALNUMBER=4C:D0:49:D5:02:5F:38,CN=QMA,OU=WebSphere MQ
Development,O=IBM,ST=Hampshire,C=UK")
   STATUS(RUNNING)                         SUBSTATE(RECEIVE)
   XMITQ( )
```

在每种情况下， SSLPEER 的值必须与步骤 2 中创建的伙伴证书中的专有名称 (DN) 的值匹配。 签发者名称与对步骤 4 中添加的个人证书进行签名的 CA 证书的主体集 DN 相匹配。

使用单向认证连接两个队列管理器

遵循以下样本指示信息来修改具有相互认证的系统，以允许队列管理器使用单向认证连接到另一个系统; 即，当 SSL 或 TLS 客户机未发送证书时。

关于此任务

方案:

- 您的两个队列管理器 (QM1 和 QM2) 已设置为 第 476 页的『使用 CA 签名的证书对两个队列管理器进行相互认证』。
- 您希望更改 QM1 ， 以便它使用单向认证连接到 QM2。

生成的配置如下所示:

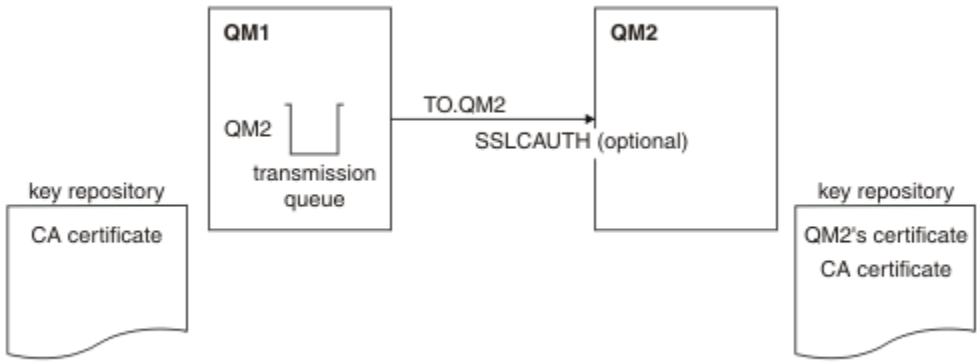


图 *98:* 允许单向认证的队列管理器

过程

1. 从其密钥存储库中除去 QM1 的个人证书:
   - IBM i 在 IBM i 系统上除去证书。
   - Windows Linux UNIX 在 UNIX， Linux 和 窗口 系统上除去证书。
   - z/OS 在 z/OS 系统上除去证书。 执行此步骤两次，以除去 QMA 的个人证书和缺省证书。

   有关如何标注 cetificates 的详细信息，请参阅 数字证书标签。
2. 可选： 在 QM1 上，如果先前已运行任何 SSL 或 TLS 通道，请刷新 SSL 或 TLS 环境 ， 如 刷新 SSL 或 TLS 环境 中所述。
3. 允许接收方上的匿名连接 ， 如 允许接收方通道上的匿名连接 中所述。

密钥存储库和通道已更改，如 第 478 页的图 98 中所示

4. 如果发送方通道未运行，请将其启动。

   注: 如果发送方通道正在运行，并且您在步骤 2 中发出了 REFRESH SECURITY TYPE (SSL) 命令，那么通道将自动重新启动。

   在通道的服务器端，通道状态屏幕上存在对等名称参数值指示客户机证书已流动。

5. 通过发出一些 DISPLAY 命令验证任务是否已成功完成。

   如果任务成功，那么生成的输出类似于以下示例中显示的输出:

   - 从 QM1 队列管理器中，输入以下命令:

     ```
     DISPLAY CHS(TO.QM2) SSLPEER SSLCERTI
     ```

     生成的输出将类似于以下示例:

     ```
     DISPLAY CHSTATUS(TO.QMB) SSLPEER SSLCERTI
          4 : DISPLAY CHSTATUS(TO.QMB) SSLPEER
     AMQ8417: Display Channel Status details.
        CHANNEL(TO.QM2)                         CHLTYPE(SDR)
        CONNAME(9.20.25.40)                     CURRENT
        RQMNAME(QM2)
        SSLCERTI("CN=WebSphere MQ CA,OU=WebSphere MQ Devt,O=IBM,ST=Hampshire,C=UK")
        SSLPEER("SERIALNUMBER=4C:D0:49:D5:02:5F:38,CN=QMB,OU=WebSphere MQ
     Development,O=IBM,ST=Hampshire,C=UK")
        STATUS(RUNNING)                         SUBSTATE(MQGET)
        XMITQ(QM2)
     ```

   - 从 QM2 队列管理器中，输入以下命令:

     ```
     DISPLAY CHS(TO.QM2) SSLPEER SSLCERTI
     ```

     生成的输出将类似于以下示例:

     ```
     DISPLAY CHSTATUS(TO.QM2) SSLPEER SSLCERTI
          5 : DISPLAY CHSTATUS(TO.QM2) SSLPEER SSLCERTI
     AMQ8417: Display Channel Status details.
        CHANNEL(TO.QM2)                         CHLTYPE(RCVR)
        CONNAME(9.20.35.92)                     CURRENT
        RQMNAME(QMA)                            SSLCERTI( )
        SSLPEER( )                              STATUS(RUNNING)
        SUBSTATE(RECEIVE)                       XMITQ( )
     ```

     在 QM2 上， SSLPEER 字段为空，显示 QM1 未发送证书。 在 QM1 上， SSLPEER 的值与 QM2 个人证书中的 DN 的值相匹配。

安全地将客户机连接到队列管理器

使用 SSL 或 TLS 加密安全协议的安全通信涉及设置通信通道和管理将用于认证的数字证书。

要设置 SSL 或 TLS 安装，必须定义通道以使用 SSL 或 TLS。 您还必须获取和管理数字证书。 在测试系统上，可以使用自签名证书或本地认证中心 (CA) 颁发的证书。 在生产系统上，请勿使用自签名证书。

有关创建和管理证书的完整信息，请参阅以下主题:

- IBM i 在 IBM i 上使用 SSL 或 TLS
- distributed 在 UNIX， Linux 和 Windows 系统上使用 SSL 或 TLS
- z/OS 在 z/OS 上使用 SSL 或 TLS

此主题集合介绍了设置 SSL 通信所涉及的任务，并提供了有关完成这些任务的逐步指导。

您可能还想要测试 SSL 或 TLS 客户机认证，这是协议的可选部分。 在 SSL 或 TLS 握手期间， SSL 或 TLS 客户机始终从服务器获取并验证数字证书。 通过 IBM MQ 实现， SSL 或 TLS 服务器始终从客户机请求证书。

在 IBM i IBM i， UNIX， Linux 和 窗口 系统上，仅当 SSL 或 TLS 客户机具有以正确的 IBM MQ 格式标注的证书 (`ibmwebspheremq` 后跟小写的登录用户标识) 或 **CERTLABL** 属性的值时， SSL 或 TLS 客户机才会发送证书。 请参阅 数字证书标签 (Digital certificate labels)。

SSL 或 TLS 服务器始终验证客户机证书 (如果发送了客户机证书)。 如果客户机未发送证书，那么仅当充当 SSL 或 TLS 服务器的通道的末尾定义了 SSLCAUTH 参数设置为 REQUIRED 或 SSLPEER 参数值时，认证才会失败。 有关以匿名方式连接队列管理器的更多信息，请参阅 第 483 页的『以匿名方式将客户机连接到队列管理器』。

相关信息

IBM MQ classes for Java 中的 SSL/TLS CipherSpec 和 CipherSuite
IBM MQ classes for JMS 中的 SSL/TLS CipherSpec 和 CipherSuite
对受管 .NET 客户机使用证书

使用自签名证书对客户机和队列管理器进行相互认证

遵循以下样本指示信息，通过使用自签名 SSL 或 TLS 证书来实现客户机与队列管理器之间的相互认证。

关于此任务

方案:

- 您具有需要安全通信的客户机 C1 和队列管理器 QM1。 您需要在 C1 和 QM1 之间执行相互认证。
- 您已决定使用自签名证书来测试安全通信。

IBM i 上的 DCM 不支持自签名证书，因此此任务不适用于 IBM i 系统。

生成的配置如下所示:

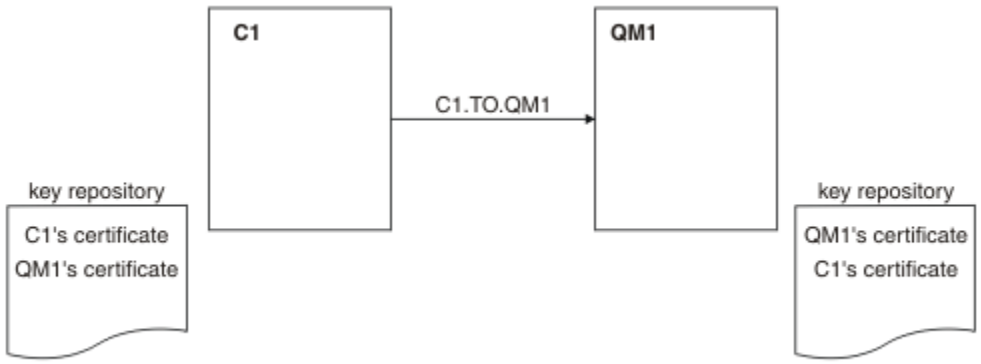


图 99: 由此任务生成的配置

在 第 480 页的图 99 中， QM1 的密钥存储库包含 QM1 的证书和来自 C1 的公用证书。 C1 的密钥存储库包含 C1 的证书以及来自 QM1 的公用证书。

过程

1. 根据操作系统，在客户机和队列管理器上准备密钥存储库:
   - 在 UNIX， Linux 和 窗口 系统上。
   - z/OS 在 z/OS 系统上 (仅限队列管理器)。
2. 为客户机和队列管理器创建自签名证书:
   - 在 UNIX， Linux 和 窗口 系统上。
   - z/OS 在 z/OS 系统上 (仅限队列管理器)。
3. 抽取每个证书的副本:

- 在 UNIX， Linux 和 窗口 系统上。
- z/OS 在 z/OS 系统上。

4. 使用诸如 FTP z/OS 之类的实用程序将 C1 证书的公用部分传输到 QM1 系统，反之亦然，如 交换自签名证书 中所述。
5. 将伙伴证书添加到客户机和队列管理器的密钥存储库:
   - 在 UNIX， Linux 和 窗口 系统上。
   - z/OS 在 z/OS 系统上。
6. 在队列管理器上发出命令 REFRESH SECURITY TYPE (SSL)。
7. 通过以下任一方式定义客户机连接通道:
   - 将 MQCONNX 调用与 C1 上的 MQSCO 结构配合使用，如 在 IBM MQ MQI client 上创建客户机连接通道中所述。
   - 使用客户机通道定义表，如 在服务器上创建服务器连接和客户机连接定义中所述。
8. 在 QM1 上，通过发出类似于以下示例的命令来定义服务器连接通道:

```
DEFINE CHANNEL(C1.TO.QM1) CHLTYPE(SVRCONN) TRPTYPE(TCP) SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA)
SSLCAUTH(REQUIRED) DESCR('Receiver channel using SSL from C1 to QM1')
```

   通道必须与您在步骤 6 中定义的客户机连接通道同名，并使用相同的 CipherSpec。

结果

将创建密钥存储库和通道，如 第 480 页的图 99 中所述。

下一步做什么

使用 DISPLAY 命令检查任务是否已成功完成。 如果任务成功，那么生成的输出与以下示例中显示的输出相似。

从队列管理器 QM1，输入以下命令:

```
DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
```

生成的输出类似于以下示例:

```
DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
     5 : DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
AMQ8417: Display Channel Status details.
   CHANNEL(C1.TO.QM1)                      CHLTYPE(SVRCONN)
   CONNAME(9.20.35.92)                       CURRENT
   SSLCERTI("CN=QM1,OU=WebSphere MQ Development,O=IBM,ST=Hampshire,C=UK")
   SSLPEER("SERIALNUMBER=4C:D0:49:D5:02:5E:02,CN=QM2,OU=WebSphere MQ
Development,O=IBM,ST=Hampshire,C=UK")
   STATUS(RUNNING)                           SUBSTATE(RECEIVE)
```

可选择设置通道定义的 SSLPEER 过滤器属性。 如果设置了通道定义 SSLPEER ，那么其值必须与步骤 2 中创建的伙伴证书中的主体集 DN 相匹配。 成功连接后， DISPLAY CHSTATUS 输出中的 SSLPEER 字段将显示远程客户机证书的主题 DN。

使用 CA 签名的证书对客户机和队列管理器进行相互认证

遵循以下样本指示信息，通过使用 CA 签署的 SSL 或 TLS 证书来实现客户机与队列管理器之间的相互认证。

关于此任务

方案:

- 您具有需要安全通信的客户机 C1 和队列管理器 QM1。 您需要在 C1 和 QM1 之间执行相互认证。

- 将来您计划在生产环境中使用此网络，因此您决定从一开始就使用 CA 签署的证书。

生成的配置如下所示:

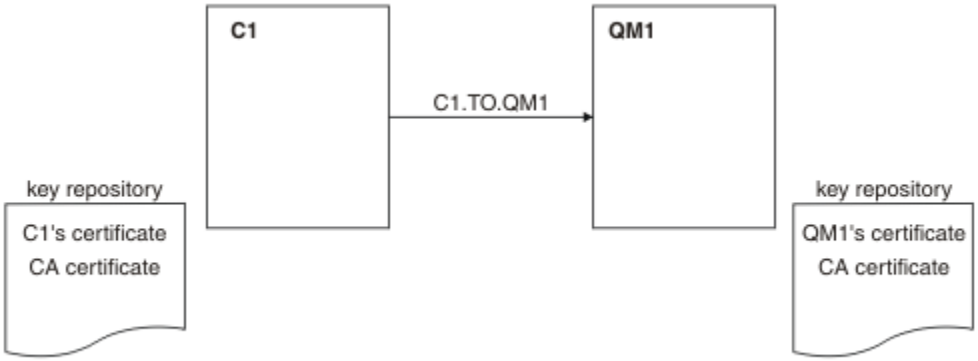


图 *100:* 由此任务生成的配置

在 第 482 页的图 100 中， C1 的密钥存储库包含 C1 的证书和 CA 证书。 QM1 的密钥存储库包含 QM1 的证书和 CA 证书。 在此示例中， C1 的证书和 QM1 的证书都由同一 CA 发放。 如果 C1 的证书和 QM1 的证书是由不同的 CA 发放的，那么 C1 和 QM1 的密钥存储库必须同时包含两个 CA 证书。

过程

1. 根据操作系统，在客户机和队列管理器上准备密钥存储库:
   - IBM i 在 IBM i 系统上。
   - 在 UNIX， Linux 和 窗口 系统上。
   - z/OS 在 z/OS 系统上 (仅限队列管理器)。
2. 请求客户机和队列管理器的 CA 签名证书。
   您可以对客户机和队列管理器使用不同的 CA。
   - IBM i 在 IBM i 系统上。
   - 在 UNIX， Linux 和 窗口 系统上。
   - z/OS 在 z/OS 系统上 (仅限队列管理器)。
3. 将认证中心证书添加到客户机和队列管理器的密钥存储库。
   如果客户机和队列管理器使用不同的认证中心，那么必须将每个认证中心的 CA 证书添加到两个密钥存储库。
   - IBM i 请勿在 IBM i 系统上执行此步骤。
   - 在 UNIX， Linux 和 窗口 系统上。
   - z/OS 在 z/OS 系统上 (仅限队列管理器)。
4. 将 CA 签名的证书接收到客户机和队列管理器的密钥存储库:
   - IBM i 在 IBM i 系统上。
   - 在 UNIX， Linux 和 窗口 系统上。
   - z/OS 在 z/OS 系统上 (仅限队列管理器)。
5. 通过以下任一方式定义客户机连接通道:
   - 将 MQCONNX 调用与 C1 上的 MQSCO 结构配合使用，如 在 IBM MQ MQI client 上创建客户机连接通道中所述。

- 使用客户机通道定义表，如 在服务器上创建服务器连接和客户机连接定义中所述。

6. 在 QM1 上，通过发出类似于以下示例的命令来定义服务器连接通道:

```
DEFINE CHANNEL(C1.TO.QM1) CHLTYPE(SVRCONN) TRPTYPE(TCP)
SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA) SSLCAUTH(REQUIRED)
DESCR('Receiver channel using SSL from C1 to QM1')
```

通道必须与您在步骤 6 中定义的客户机连接通道同名，并使用相同的 CipherSpec。

结果

将创建密钥存储库和通道，如 第 482 页的图 100 中所述。

下一步做什么

使用 DISPLAY 命令检查任务是否已成功完成。 如果任务成功，那么生成的输出类似于以下示例中显示的输出。

从队列管理器 QM1 中，输入以下命令:

```
DISPLAY CHSTATUS(TO.QMB) SSLPEER SSLCERTI
```

生成的输出类似于以下示例:

```
DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
     5 : DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
AMQ8417: Display Channel Status details.
   CHANNEL(C1.TO.QM1)                      CHLTYPE(SVRCONN)
   CONNAME(9.20.35.92)                     CURRENT
   SSLCERTI("CN=WebSphere MQ CA,OU=WebSphere MQ Devt,O=IBM,ST=Hampshire,C=UK")
   SSLPEER("SERIALNUMBER=4C:D0:49:D5:02:5F:38,CN=QMA,OU=WebSphere MQ
Development,O=IBM,ST=Hampshire,C=UK")
   STATUS(RUNNING)                         SUBSTATE(RECEIVE)
```

DISPLAY CHSTATUS 输出中的 SSLPEER 字段显示在步骤 2 中创建的远程客户机证书的主题 DN。 签发者名称与对步骤 4 中添加的个人证书进行签名的 CA 证书的主体集 DN 相匹配。

以匿名方式将客户机连接到队列管理器

遵循以下样本指示信息来修改具有相互认证的系统，以允许队列管理器以匿名方式连接到另一个队列管理器。

关于此任务

方案:

- 您的队列管理器和客户机 (QM1 和 C1) 已设置为与 第 481 页的『使用 CA 签名的证书对客户机和队列管理器进行相互认证』中一样。
- 您希望更改 C1 ，以便它以匿名方式连接到 QM1。

生成的配置如下所示:

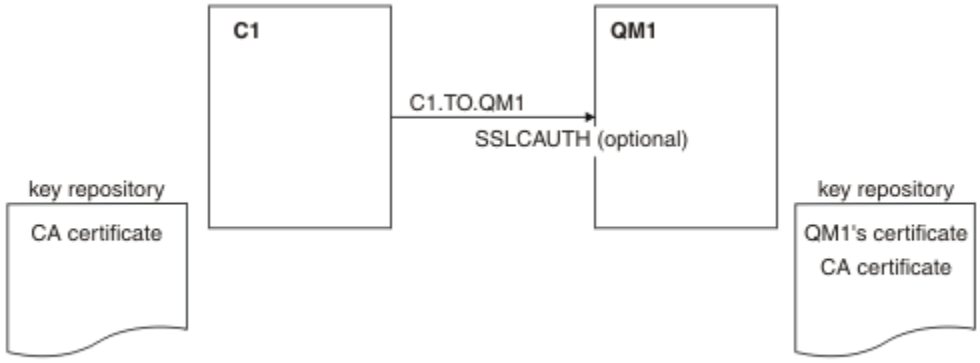


图 *101:* 允许匿名连接的客户机和队列管理器

过程

1. 根据操作系统，从 C1 的密钥存储库中除去个人证书:
   - IBM i IBM i 系统。
   - Windows Linux UNIX UNIX， Linux 和 窗口 系统。

   证书标签是 `ibmwebspheremq` 后跟小写的登录用户标识或 **CERTLABL** 属性的值。 请参阅 数字证书标签 (Digital certificate labels)。
2. 重新启动客户机应用程序，或者使客户机应用程序关闭并重新打开所有 SSL 或 TLS 连接。
3. 通过发出以下命令，允许队列管理器上的匿名连接:

```
ALTER CHANNEL(C1.TO.QM1) CHLTYPE(SVRCONN) SSLCAUTH(OPTIONAL)
```

结果

密钥存储库和通道已更改，如 第 484 页的图 101 中所示

下一步做什么

在通道的服务器端，通道状态屏幕上存在对等名称参数值指示客户机证书已流动。

通过发出一些 DISPLAY 命令验证任务是否已成功完成。 如果任务成功，那么生成的输出类似于以下示例中显示的输出:

从队列管理器 QM1，输入以下命令:

```
DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
```

生成的输出将类似于以下示例:

```
DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
     5 : DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
AMQ8417: Display Channel Status details.
   CHANNEL(C1.TO.QM1)                      CHLTYPE(SVRCONN)
   CONNAME(9.20.35.92)                     CURRENT
   SSLCERTI( )                             SSLPEER( )
   STATUS(RUNNING)                         SUBSTATE(RECEIVE)
```

SSLCERTI 和 SSLPEER 字段为空，显示 C1 未发送证书。

文件传输方案

演示如何使用 Managed File Transfer 功能的场景。

IBM MQ Managed File Transfer 文档中描述了可用的文件传输方案，从 方案概述开始。

声明

本信息是为在美国提供的产品和服务编写的。

IBM 可能在其他国家或地区不提供本文档中讨论的产品、服务或功能。 有关您当前所在区域的产品和服务的信息，请向您当地的 IBM 代表咨询。 任何对 IBM 产品、程序或服务的引用并非意在明示或默示只能使用 IBM 的产品、程序或服务。 只要不侵犯 IBM 的知识产权，任何同等功能的产品、程序或服务都可以代替 IBM 产品、程序或服务。 但是， 评估和验证任何非 IBM 产品、程序或服务的操作，由用户自行负责。

IBM 公司可能已拥有或正在申请与本文档内容有关的各项专利。 提供本文档并未授予用户使用这些专利的任何许可。 您可以以书面形式将许可查询寄往：

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

有关双字节（DBCS）信息的许可查询，请与您所在国家或地区的 IBM 知识产权部门联系，或用书面方式将查询寄往：

知识产权许可
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 063-8506 Japan

本条款不适用英国或任何这样的条款与当地法律不一致的国家或地区: International Business Machines Corporation “按现状” 提供本出版物，不附有任何种类的 (无论是明示的还是暗含的) 保证，包括但不限于暗含的有关非侵权，适销和适用于某种特定用途的保证。 某些国家或地区在某些交易中不允许免除明示或暗含的保证。 因此本条款可能不适用于您。

本信息中可能包含技术方面不够准确的地方或印刷错误。 此处的信息将定期更改；这些更改将编入本资料的新版本中。 IBM 可以随时对本出版物中描述的产品和/或程序进行改进和/或更改，而不另行通知。

本信息中对非 IBM Web 站点的任何引用都只是为了方便起见才提供的，不以任何方式 充当对那些 Web 站点的保证。 那些 Web 站点中的资料不是 IBM 产品资料的一部分，使用那些 Web 站点带来的风险将由您自行承担。

IBM 可以按它认为适当的任何方式使用或分发您所提供的任何信息而无须对您承担任何责任。

本程序的被许可方如果要了解有关程序的信息以达到如下目的：(i) 允许在独立创建的程序和其他程序（包括本程序）之间进行信息交换，以及 (ii) 允许对已经交换的信息进行相互使用，请与下列地址联系：

IBM Corporation
软件互操作性协调员，部门 49XA
北纬 3605 号公路
罗切斯特，明尼苏达州 55901
U.S.A.

只要遵守适当的条件和条款，包括某些情形下的一定数量的付费，都可获得这方面的信息。

本资料中描述的许可程序及其所有可用的许可资料均由 IBM 依据 IBM 客户协议、IBM 国际软件许可协议或任何同等协议中的条款提供。

此处包含的任何性能数据都是在受控环境中测得的。 因此，在其他操作环境中获得的数据可能会有明显的不同。 有些测量可能是在开发级的系统上进行的，因此不保证与一般可用系统上进行的测量结果相同。 此外，有些测量是通过推算而估计的， 实际结果可能会有差异。 本文档的用户应当验证其特定环境的适用数据。

© Copyright IBM Corp. 2007, 2023

涉及非 IBM 产品的信息可从这些产品的供应商、其出版说明或其他可公开获得的资料中 获取。 IBM 没有对这些产品进行测试，也无法确认其性能的精确性、兼容性或任何其他关于非 IBM 产品的声明。 有关非 IBM 产品性能的问题应当向这些产品的供应商提出。

所有关于 IBM 未来方向或意向的声明都可随时更改或收回，而不另行通知，它们仅仅表示了目标和意愿而已。

本信息包含日常商业运作所使用的数据和报表的示例。 为了尽可能全面地说明这些数据和报表，这些示例包括个人、公司、品牌和产品的名称。 所有这些名称都是虚构的，如与实际商业企业所使用的名称和地址有任何雷同，纯属巧合。

版权许可：

本信息包含源语言形式的样本应用程序，用以阐明在不同操作平台上的编程技术。 如果是为按照在编写样本程序的操作平台上的应用程序编程接口（API）进行应用程序的开发、使用、经销或分发为目的，您可以任何形式对这些样本程序进行复制、修改、分发，而无须向 IBM 付费。 这些示例并未在所有条件下作全面测试。 因此，IBM 不能担保或默示这些程序的可靠性、可维护性或功能。

如果您正在查看本信息的软拷贝，图片和彩色图例可能无法显示。

编程接口信息

编程接口信息 (如果提供) 旨在帮助您创建用于此程序的应用软件。

本书包含有关允许客户编写程序以获取 WebSphere MQ 服务的预期编程接口的信息。

但是，该信息还可能包含诊断、修改和调优信息。 提供诊断、修改和调优信息是为了帮助您调试您的应用程序软件。

要点: 请勿将此诊断，修改和调整信息用作编程接口，因为它可能会发生更改。

商标

IBM IBM 徽标 ibm.com 是 IBM Corporation 在全球许多管辖区域的商标。 当前的 IBM 商标列表可从 Web 上的 “Copyright and trademark information”www.ibm.com/legal/copytrade.shtml 获取。 其他产品和服务名称可能是 IBM 或其他公司的商标。

Microsoft 和 Windows 是 Microsoft Corporation 在美国和/或其他国家或地区的商标。

UNIX 是 Open Group 在美国和其他国家或地区的注册商标。

Linux 是 Linus Torvalds 在美国和/或其他国家或地区的商标。

此产品包含由 Eclipse 项目 (http://www.eclipse.org/) 开发的软件。

Java 和所有基于 Java 的商标和徽标是 Oracle 和/或其附属公司的商标或注册商标。

部件号:

(1P) P/N: