**WebSphere MQ > WebSphere MQ 8.0.0 > IBM MQ > Planning**

<http://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.helphome.v80.doc/WelcomePagev8r0.htm>

**What’s New in MQ 8.0.0**

<http://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.pro.doc/q113110_.htm>

Simpler and more flexible security administration

The ability to integrate with existing identity repositories, enabling consistency across a solution and reducing security administration for solutions with many users

More flexible administration of client connection information

Customizable publication-routing for more efficiency in large scale clusters

The ability to separate workloads by defining multiple cluster transmission queues on all supported platforms

[z/OS]Massively increased capacity of buffer pools in IBM MQ for z/OS®

[z/OS]Increased maximum addressable log range in IBM MQ for z/OS

Improved performance for database replication traffic using the QREP feature of DB2®

Improved efficiency and more linear scaling through refinements to locking and optimization of codepaths

Full 64-bit queue manager implementation on Windows and support for SSL in managed .NET applications

Support for the JMS 2.0 version of the JMS standard, including deferred message delivery, shared subscriptions and asynchronous send operation

Improved online documentation delivery through IBM Knowledge Center

**Set of Documentations**

<http://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.pro.doc/q001040_.htm>

**Code Scanning for Deprecated Function**

**What’s New in MQ 8.0.0**

<http://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.pro.doc/q113110_.htm>

**What’s changed in IBM MQ Version?**

<http://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.pro.doc/q001060_.htm>

**List of changes that affect migration**

<http://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.mig.doc/q007690_.htm>

**Changes from WebSphere MQ Version 7.0.1 to IBM MQ Version 8.0**

<http://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.mig.doc/q007810_.htm>

**Migrating and upgrading IBM MQ**

<http://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.mig.doc/q006180_.htm>

**Maintenance, migration and upgrading**

<http://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.mig.doc/q006180_.htm>

IBM MQ uses this terminology as follows:

Migration is the process of updating queue manager data to match a newer level of code. This occurs the first time a queue manager is started with the newer level of code.

Maintenance is the application of a fix pack, interim fix or program Temporary Fix (PTF). The installation can be restored to its previous level and queue managers or applications continue to work. Migration is not required after applying maintenance. However, you should test applications with the new level of IBM MQ code.

Upgrading is the process of taking an existing IBM MQ installation and upgrading to a new level of code. Unless the upgrade is applying a fix (and not enabling new function), an upgrade must be followed by migration.

Once migration has occurred, the queue manager can no longer be started by an earlier code level. On most platforms, queue manager migration is not reversible. [z/OS]The exception is z/OS®, where you can reverse queue manager migration, but only if you have not enabled new function.

**MQ Technical Overview**

<http://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.pro.doc/q002610_.htm>

**Styles of Message Queuing**

* Point-to-Point
* Publish/Subscribe

**Appendix**

**Message**

In message queuing, a message is a collection of data sent by one program and intended for another program. See [IBM MQ messages](http://www.ibm.com/support/knowledgecenter/en/SSFKSJ_8.0.0/com.ibm.mq.dev.doc/q022860_.htm?view=kc). For information about message types, see [Types of message](http://www.ibm.com/support/knowledgecenter/en/SSFKSJ_8.0.0/com.ibm.mq.dev.doc/q022870_.htm?view=kc).

**Message descriptor**

An IBM MQ message consists of control information and application data.

The control information is defined in a message descriptor structure (MQMD) and contains such things as:

* The type of the message
* An identifier for the message
* The priority for delivery of the message

The structure and content of the application data is determined by the participating programs, not by IBM MQ.

**Queue**

A named destination to which messages can be sent. Messages accumulate on queues until they are retrieved by programs that service those queues.

**Queue manager**

A queue manager is a system program that provides queuing services to applications.

It provides an application programming interface so that programs can put messages on, and get messages from, queues. A queue manager provides additional functions so that administrators can create new queues, alter the properties of existing queues, and control the operation of the queue manager.

For IBM MQ message queuing services to be available on a system, there must be a queue manager running. You can have more than one queue manager running on a single system (for example, to separate a test system from a live system). To an application, each queue manager is identified by a connection handle (*Hconn*).

Many different applications can use the services of the queue manager at the same time and these applications can be entirely unrelated. For a program to use the services of a queue manager, it must establish a connection to that queue manager.

For applications to send messages to applications that are connected to other queue managers, the queue managers must be able to communicate among themselves. IBM MQ implements a store-and-forward protocol to ensure the safe delivery of messages between such applications.

**Channels**

Channels are objects that provide a communication path from one queue manager to another. Channels are used in distributed queuing to move messages from one queue manager to another and they shield applications from the underlying communications protocols. The queue managers might exist on the same, or different, platforms.

**Message channel agent**

A message channel agent is one end of a channel. A pair of message channel agents, one sending and one receiving, make up a channel and move messages from one queue manager to another.

**Cluster**

A cluster is a network of queue managers that are logically associated in some way.

In an IBM MQ network using distributed queuing without clustering, every queue manager is independent. If one queue manager needs to send messages to another, it must have defined a transmission queue and a channel to the remote queue manager.

There are two different reasons for using clusters: to reduce system administration and to improve availability and workload balancing.

As soon as you establish even the smallest cluster, you benefit from simplified system administration. Queue managers that are part of a cluster need fewer definitions and so the risk of making an error in your definitions is reduced.

For more information about clustering, see [Clusters](http://www.ibm.com/support/knowledgecenter/en/SSFKSJ_8.0.0/com.ibm.mq.pro.doc/q002660_.htm?view=kc#q002660___clusters).

**[z/OS]Shared queue**

[z/OS]

A shared queue is a type of local queue with messages that can be accessed by one or more queue managers that are in a sysplex. This is not the same as a queue being shared by more than one application, using the same queue manager. This applies only to IBM MQ for z/OS®.

**[z/OS]Queue-sharing group**

[z/OS]

The queue managers that can access the same set of shared queues form a group called a queue-sharing group (QSG). They communicate with each other with a coupling facility (CF) that stores the shared queues. This applies only to IBM MQ for z/OS. See [Shared queues and queue-sharing groups](http://www.ibm.com/support/knowledgecenter/en/SSFKSJ_8.0.0/com.ibm.mq.pro.doc/q003630_.htm?view=kc) for more information about queue-sharing groups.

**[z/OS]Intra-group queuing**

[z/OS]

Queue managers in a queue-sharing group can communicate using normal channels or you can use a technique called intra-group queuing (IGQ), which lets you perform fast message transfer without defining channels. This applies only to IBM MQ for z/OS.

**IBM MQ MQI client**

IBM MQ MQI clients are independently installable components of IBM MQ. An MQI client allows you to run IBM MQ applications with a communications protocol, to interact with one or more Message Queue Interface (MQI) servers on other platforms and to connect to their queue managers.

For full details on how to install and use IBM MQ MQI client components, see [Installing an IBM MQ MQI client](http://www.ibm.com/support/knowledgecenter/en/SSFKSJ_8.0.0/com.ibm.mq.ins.doc/q008960_.htm?view=kc)and [Configuring connections between the server and client](http://www.ibm.com/support/knowledgecenter/en/SSFKSJ_8.0.0/com.ibm.mq.con.doc/q016480_.htm?view=kc).

**Point-to-point messaging**

In point-to-point messaging, each message travels from one producing application to one consuming application. Messages are transferred through the producing application putting messages onto a queue and the consuming application gets them from that queue.

**Publish/subscribe messaging**

In publish/subscribe messaging, a copy of each message published by a publishing application is delivered to every interested application. There might be many, one or no interested applications. In publish/subscribe an interested application is known as a subscriber and the messages are queued on a queue identified by a subscription. For more information, see [Publish/subscribe messaging](http://www.ibm.com/support/knowledgecenter/en/SSFKSJ_8.0.0/com.ibm.mq.pro.doc/q004870_.htm?view=kc).

**Topic**

A topic is a character string that describes the subject of the information that is published in a publish/subscribe message.

Topics are key to the successful delivery of messages in a publish/subscribe system. Instead of including a specific destination address in each message, a publisher assigns a topic to each message. The queue manager matches the topic with a list of subscribers who have subscribed to that topic, and delivers the message to each of those subscribers.

**Subscription**

A publish/subscribe application can register an interest in messages about specific topics. When an application does this it is known as a subscriber and the term subscription defines how matching messages are queued for processing.

A subscription contains information about the identity of the subscriber and the identity of the destination queue on to which publications are to be placed. It also contains information about how a publication is to be placed on the destination queue.