AnypointMQ Connector

Anypoint MQ is an enterprise, multi-tenant, cloud messaging service that performs advanced asynchronous messaging scenarios between applications. Anypoint MQ is fully integrated with Anypoint Platform, offering role-based access control, client app management, and connectors.

Provides fully hosted and managed queues and message exchanges in the cloud. A queue is a temporary storage area. A message exchange binds one or more queues so that a message sent to a message exchange appears in all its bound queues simultaneously.

Enables you to easily connect to non-Mule applications using the REST API.

Provide REST API to deploy to On-premise

* **Horizontal scalability**

By putting a message on a queue, the work to process that message can be distributed over multiple subscribers.

* **Reliability**

Queues ensure that messages are always processed, even in the case of failure.

In Anypoint MQ, reliability is provided with queues using a lock and [acknowledge mechanism](https://docs.mulesoft.com/mq/mq-ack-mode). This ensures that a message is always processed, even in the case of failure.

* **Decoupling**

A queue can be used as a communication mechanism between different applications and devices, allowing them to have different development life cycles or maintenance windows, without affecting other applications.

* **Aggregation**

Multiple message producers can publish to a single queue, which enables easy consumption by consumers.

Client

In Anypoint MQ, each consumer must be registered as a client app with a name, ID, and a secret. The client app uses the ID and secret to authenticate and get access to queues and message exchanges within a single environment. Client apps cannot be used across multiple environments or business groups.

Queue

Message queues provide temporary storage that enables applications to communicate in a scalable and reliable way. Publishers can publish messages into queues, and subscribers can receive messages from queues.

3 types of queue

1. Standardise Queue
2. FIFO Queue
3. Exchange Queue

FIFO queues are used when ordering of the message is considered, avoid duplication, it will be bit slow

Standardised queues are used when order of message is not involved.

With message exchanges you can distribute a single message to multiple consumers.

For example, if a publisher publishes changes in the weather, and you have multiple apps that want to receive that data, you can use a message exchange to publish a single message to multiple queues for each app. A *binding* is a relationship between the queue and message exchange, which tells the message exchange where to send messages. A queue can be bound to multiple message exchanges.

**In Queue and In flight**

In Queue: messages that are being sent. No limit on the number of messages

In flight: messages that are received by the queue, but not deleted. Waiting for ACK/NACK

Limit- 120,000 queue

10 FIFO

Dead letter queue

A queue that stores undeliverable messages from other queues.

Type of queue and dead letter queue should be same and should be in same region and environment

JMS is also used for interapp communication,

JMS uses the Java Messaging Service protocol and requires an external JMS broker such as ActiveMQ. Can be used by any application that supports JMS connectivity.