Using IBM MQ Pub/Sub with Unmanaged Subscriptions

https://www.ibm.com/support/pages/node/154105

Date last updated: 14-Feb-2023

IBM MQ Support
https://www.ibm.com/products/mq/support
Find all the support you need for IBM MQ

+++ Problem +++

You have a legacy configuration that uses the IBM MQ V6 style for Publish/Subscribe, which is called "queued Pub/Sub". This old style was replaced by a totally new way of doing Pub/Sub, starting with IBM MQ V7.

You are exploiting the old feature of "unmanaged subscription", which does not have a direct counterpart with the new implementation and you would like to know if there is a way to force recent versions IBM MQ to use predefined queue for subscription without manually creating a subscription with new MQ verbs or MQ Explorer?

++ Symptom ++

Customer is working on a project that will migrate a V6-style message broker pub/sub infrastructure to use the new pub/sub. One major difference that was encountered is that using JMS program the customer can only create a "managed queue subscription" which means that MQ creates a dynamic temporary queue to store the subscribed messages.
The preferred way for the customer is to use the predefined queue in JMS topic for the messages received for the durable subscription.

The customer thinks that this way it is easier to manage and monitor. But so far the customer has not been able to achieve that using JMS program and JMS object definition like Topic Connection Factory and Topic Destination. I posted my JMSAdmin sample script below. You can see I defined brokersubq in topic definition.

The issue is about making "unmanaged JMS subscription" under recent versions of MQ. The unmanaged subscription means using a predefined durable subscription queue in JMS topic object instead of dynamic queue used which is called managed subscription.
But with this setting, the broker-side message selector will not work. With broker version set to V1, client-side message selector will work but it will significantly impact application performance hence it is not a feasible solution. The capability of using predefined subscription via the V6-style is important to the customer because the customer monitors the queue depth of the subscription queue for application health checking. With the new dynamic queue design, it will be difficult for the customer to link message build-up in a subscription queue to a specific application. It will force the customer to completely change the way they monitor their pub/sub applications.

++ Resolving The Problem ++

When the publish/subscribe API was added to the IBM MQ server code in V7, the concept of managed and unmanaged subscriptions was intended to be transparent to the JMS code, as this is a concept that does not exist in the JMS specification; it was intended that JMS applications would generally use "managed subscriptions" automatically. As a result, it seems that extending the use of the IBM MQ JMS client code to make use of unmanaged subscriptions was not covered in the documentation.

Here is a summary on the use of unmanaged subscription in JMS:
It is possible to administratively create an unmanaged subscription using the IBM MQ tooling and then access this pre-existing subscription from within a JMS application, but it is not possible to directly create an unmanaged subscriptions from within the JMS application itself.

The BrokerDurSubQueue property available on the MQTopic object was used in V6 of the IBM MQ classes for JMS and earlier to define the queue to which messages for a subscriber to that topic would be sent. There was a restriction on the naming of such a queue that it had to have the prefix "SYSTEM.JMS.D..." in order to provide a scope for possible names that could be used by the internal clean-up code in the JMS client.

With the release of IBM MQ V7 and higher, new publish/subscribe verbs were added to the MQI, and the queue manager became able to internally hold a subscription's messages. This removed the need for an externally controlled subscriber queue and made administration of the queue manager simpler.

However, it also removed the option for an application to perform work directly on a subscriber queue, outside the strict publish/subscribe operation defined by the JMS specification.

IBM MQ therefore introduced the concept of managed and unmanaged subscriptions:

- In the default managed subscription, a subscription is created and the storage and distribution of messages sent to that subscription is controlled internally by the queue manager itself.

- In the unmanaged case, messages are sent to an externally visible, named, message queue, and can therefore be viewed and operated upon by application code.

The JMS specification contains no such concept: creating a subscriber results in the opening or creation of a subscription that is internally controlled, that is, the equivalent of a managed subscription. The JMS specification distinguishes between queues and topics as completely separate and independent destination types. **It is not possible to programmatically create an unmanaged subscription from within a JMS application.**

It is possible, however, to use an existing unmanaged subscription from within a JMS application. To do so requires the subscription to be created outside the JMS application itself, and so this process is only usefully applicable to durable subscribers.

If a durable subscription is created administratively, such as through the IBM MQ Explorer, then this subscription can be re-opened when a durable Topic Subscriber is created.

++ Procedure ++

Creating an unmanaged subscription for use in JMS In order to use an unmanaged subscription from a JMS application using the IBM MQ classes for JMS:

1. Create the subscriber queue, that is, the local IBM MQ queue to which the subscriptions messages will be sent. Note that the JMS client code will still apply the same restriction to the queue name as in V6, that is, this queue's name must begin with "SYSTEM.JMS.D..."

2. Create the subscription using the IBM MQ Explorer (or runmqsc).
Enter the details of the subscriptions as:

* Subscription Name
The full subscription name as used by the JMS client code.
This is constructed according to the following rule:

```
"JMS:" + <queue manager name> + ":" + <Client ID> + ":" + <JMS subscription
name as passed to the createDurableSubscriber call in the application>
```

* Topic name
"SYSTEM.BROKER.DEFAULT.STREAM" or the name of the stream on which the subscriber will be subscribed.

* Topic string
The name value for the Topic destination on which the subscriber will be subscribed.

* Destination class: **provided**

* Destination queue manager
The name of the queue manager on which the subscription is to be created. This is the same as <queue manager name> in the Subscription name.

* Destination name
The name of the subscriber queue created in the previous step.

3. Configure the JMS ConnectionFactory object, either in JNDI or programmatically within the JMS application:

* BROKERPUBQ
The same value as used for the Topic name when creating the subscription

* CLIENTID
The same value as used for <client ID> in the full subscription name

4. Configure the JMS Topic object, either in JNDI or programmatically within the JMS application:

* BROKERDURSUBQ
The name of the local subscription queue as defined in step 1.

* QMANAGER
The same value as used for <queue manager name> in the full subscription name, and for Destination queue manager.

5. Within the JMS application itself, in the createDurableSubscriber() call, use the same name as used for the JMS subscription name when constructing the full subscription name in the Explorer Now, when the JMS application runs, the createDurableSubscriber() call will cause the existing, unmanaged, subscription to be opened.

++ Updating unmanaged subscriptions ++

A consequence of using an unmanaged subscription from JMS is that the maintenance of the subscriber queue is no longer performed automatically by the queue manager.
This is of particular relevance if a durable subscriber is updated.

According to the JMS specification, "changing a durable subscriber is equivalent to unsubscribing (deleting) the old one and creating a new one."

- For a managed subscription, this would include deleting any previously published messages that had not been received by the subscriber before it was changed.

- For an unmanaged subscription, changing the subscriber results in a change to the correlationID used by the JMS client code to identify the subscriber's messages, but previously published messages are not deleted from the queue. These previously published messages will therefore not be received by the changed subscriber, but they will remain on the queue until explicitly removed.

Subscriptions created using the IBM MQ classes for JMS are always created as managed subscriptions. The JMS specification defines Queues and Topics as independent destination types, and the internals of how the messages for a subscription are stored is not defined in the specification. The management and storage of these messages is therefore delegated to be performed by the queue manager on a managed queue.

JMS durable subscriptions will persist between instances of a particular application, and so it is possible, when using durable subscriptions, to re-open an existing subscription on the queue manager. When the JMS createDurableSubscriber() call is made, the IBM MQ classes for JMS will request access to a named subscription from the queue manager and, if this subscription already exists, it will be reopened.

This means that if an unmanaged subscription has previously been created on the queue manager with the appropriate name, then the IBM MQ classes for JMS will open that subscription and use the unmanaged queue to retrieve subscription messages. However, because the IBM MQ classes for JMS is only aware of managed subscriptions, any attempt to update the subscription (such as changing the Topic or selector) will cause the unmanaged subscription to be deleted and replaced by a new managed subscription.

The full subscription name used by the IBM MQ classes for JMS is constructed as follows:
`JMS:<QMName>:<clientID>:<subscriptionName><StreamName>`

Where:
QMName is the name of the queue manager on which the subscription has been created.

clientID is the client ID set on the ConnectionFactory in the JMS application

subscriptionName is the subscription name provided as a parameter in the createDurableSubscriber() call

StreamName is the value of the BrokerPubQueue property on the Connection Factory, but only if this has been set to a non-default value, otherwise this part is not present.

Each part of the full subscription name uses unicode escape values to replace any instances of the following characters:
`; : " \`

This means that it is possible to use JMS to access unmanaged subscriptions, and therefore to send subscription messages to a named queue. However, to do this would be to use the IBM MQ classes for JMS in a manner for which it was not originally designed, and it would not be possible to change details of such a subscription from within a JMS application. Full JMS support for unmanaged subscriptions is outside the scope of a service fix and would constitute new function.

+++ end +++