DataPower-MQ Integration Deep Dive

Robin Wiley
(Robin Wiley Training)

Your Presenter: Robin Wiley

- **Senior Instructor, IBM Messaging Products**
  - MQ Administration & Application Development
  - DataPower Administration & Service Development
  - Integration Bus Administration & Message Flow Development

- **IBM Certified:**
  - MQ Administrator
  - MQ Solution Designer
  - DataPower Solution Implementer

- **Over 40 years IT Industry Experience**
  - Network Integration
  - Managing Software Development

- **Experience: 20 years with MQ; 10 with DataPower**
  - Technical Architecture, Project Management, Installation, Training

- **Effective Instructor**
  - Over 35 years experience in corporate training and adult education
  - Brings magic to the classroom (Member, Academy of Magical Arts)
Focus:
- MQ 8 & DP 7.5

Topics:
- Queue Manager Object
- MQ Front Side Handler
- MQ URL
- Message Headers
- Error Handling
- Transaction Management

Queue Manager Object
Queue Manager Object

Host Name (IPv4)
- address:port 192.168.57.1:1414
- address(port) 192.168.57.1(1414)
- address 192.168.57.1 default port 1414

Host Name (IPv6)
- [address]:port [2202::148:248]:1414
- address(port) 2202::148:248(1414)
- address 2202::148:248 default port 1414

Host Name (DNS)
- hostname:port myserver:1414
- hostname(port) myserver(1414)
- hostname myserver default port 1414
Queue Manager Object

- **Channel Name**
  - SVRCONN name as defined on the Queue Manager

- **Channel Heartbeat (seconds)**
  - Approximate time between heartbeat flows on the channel
  - 0 = no heartbeat flow exchanged
  - Does not set the heartbeat on the channel
  - Negotiates heartbeat value with channel definition -- greater is used

- **Cache Timeout (seconds)**
  - How long the appliance keeps alive a dynamic connection in the connection cache
  - Must be greater than the negotiated heartbeat interval but less than the Queue Manager keep alive interval (defined on the host)

Queue Manager Object

- **User Name**
  - Supplied to Queue Manager at connection
  - Maximum 12 characters

- **Alternate User**
  - Enables or disables MQOD.AlternateUserId
  - Off = use Message Descriptor User Identifier for queue authorization
  - On (default) = use Object Descriptor Alternate User Identifier for queue authorization (need to create Object Descriptor)

- **XML Manager**
  - Recommend using a dedicated XML Manager per QM object
Queue Manager Object

- **Maximum Message Size**
  - Limit the size of the MQ message payload

- **Units of Work**
  - Controls syncpoint processing (transaction management)
  - Affects MQ Front Side Handlers only
  - Two values: 0 or 1

- **Units of Work = 0**
  - No syncpoint control
  - Front Side Handler MQGET is immediate and irrevocable
  - If error, message integrity is responsibility of DataPower

- **Units of Work = 1**
  - Enables syncpoint control
  - Front Side Handler MQGET has an implied MQGMO_SYNCPOINT
  - If error, message is rolled back via implied MQBACK

---

Queue Manager – Connections Tab

**Open Connections**
- **Total Connection Limit**: 350
- **Initial Connections**: 1
- **Local Address**: 

**Retry Behavior**
- **Automatic Retry**: on
- **Retry Interval**: 1 seconds
- **Retry Attempts**: 0 attempts
- **Long Retry Interval**: 1000 seconds
- **Reporting Interval**: 1 seconds

**Conversation Sharing**
- **Sharing Conversations**: 0
Queue Manager – Connections Tab

- **Total Connection Limit:**
  - Connection pool size of the QM object
  - Default value is 250
  - Can tune the total connection limit for performance

- **Initial Connections:**
  - Maximum simultaneous initial connection requests
  - Default value is 1
  - If too high, may flood the Queue Manager

- **Local Address**
  - Local address for outbound connections
  - Specific local interface and port
  - For a range of ports, use (1414,1420) or x.x.x.x(1414,1420)

Queue Manager – Connections Tab

- **Automatic Retry**
  - On: Attempt to reconnect to the Queue Manager if connection dropped
  - Off: Disable and re-enable the Queue Manager object to reconnect

- **Retry Interval**
  - Time interval between attempts to retry failed connections
  - Recommend 10 to 15 seconds (default is 1 second)
  - Low value can spike CPU and memory usage

- **Retry Attempts**
  - Number of attempts to retry the failed connections
  - After threshold reached, the Long Retry Interval is used instead
  - Default value of 0 (zero) disables Long Retry Interval
  - Recommend non-zero value
Queue Manager – Connections Tab

- **Long Retry Interval**
  - Interval in seconds to retry connection after Retry Attempts
  - Recommend 600 seconds (default is 1800)
  - Must be greater than the Retry Interval (if less, ignored)

- **Reporting Interval**
  - How often to log retries (seconds)
  - Suppresses duplicate log entries
  - Recommend setting this the same as Retry Interval

Queue Manager – Connections Tab

- **Sharing Conversations**
  - Maximum conversations sharing single TCP/IP connection
  - Value is negotiated between SVRCONN SHARECNV setting and DataPower (lower value takes effect)
  - Value of 0 means **NO** Shared Conversations
    - Suppresses MQ V7+ features (Administrator stop-quiesce; Heartbeating; Read ahead; Client asynchronous consume)
  - Value of 1 means **NO** Shared Conversations
    - Allows MQ V7+ features (Administrator stop-quiesce; Heartbeating; Read ahead; Client asynchronous consume)
  - Value > 1 means Shared Conversations permitted
    - Allows MQ V7+ features (Administrator stop-quiesce; Heartbeating; Read ahead; Client asynchronous consume)
  - Can impact performance of clients (unless V7+ features used)
Queue Manager – Connections Tab

SSL Client Type: Client Profile
- Select the SSL Client Profile object to use from the pick list
- Must be used for connection to a z/OS host

SSL Client Type: Proxy Profile
- Deprecated – recommend using Client Profile instead
- Select the SSL Proxy Profile object to use from the pick list

SSL Key Repository
- Select the location of the key database file

SSL Version 3 Support
- Permit SSL v3 or not

SSL Cipher Specification
- Choose the Cipher Spec to use
Client Profile Object

Client Profile – Main Tab

- Protocols
  - Enable SSL version 3
  - Enable TLS version 1.0
  - Enable TLS version 1.1
  - Enable TLS version 1.2

- Ciphers
  - ECCM/ECDH_WITH_AES_256_GCM_SHA384
  - ECCM/ECDH_WITH_AES_256_GCM_SHA256
  - ECCM/ECDH_WITH_AES_256_CBC_SHA384
  - ECCM/ECDH_WITH_AES_256_CBC_SHA256

- Features
  - Use SME
  - Permit connections to insecure SSL servers
  - Enable compression

- Use custom SME Hostname

- Credential
  - Identification credentials
  - Validate server certificate: on
  - Validation credentials

Client Profile – Main Tab

- **Protocols:**
  - Choose the protocols to be supported

- **Ciphers:**
  - Choose the ciphers to be supported

- **Use SNI:**
  - Send the Server Name Indication (SNI) TLS extension in the client hello message

- **Permit connections to insecure SSL servers:**
  - Allow connection to potentially vulnerable servers

- **Enable compression:**
  - Allow SSL compression
  - Not recommended – can allow CRIME or BREACH attacks

- **Identification credentials:**
  - If mutual authentication requested by server

- **Validate server certificate:**
  - Check the credentials presented by the server (Default: On)

- **Validation credentials:**
  - Crypto Validation Credential object used for server certificate validation

Client Profile – Session Caching Tab

SSL Client Profile

Apply | Cancel

**Name**

**Enable session caching**

**Session cache timeout**

300 seconds

**Session cache size**

100 entries
Client Profile – Session Caching Tab

- **Enable session caching:**
  - Allow SSL session caching

- **Session Cache Timeout:**
  - How long before cache is flushed
  - Maximum: 86,400 seconds (24 hours)

- **Session Cache Size:**
  - How many entries to be cached
  - Maximum: 500,000

Client Profile – Advanced Tab

- **Elliptical Curves:**
  - Build a list of acceptable Elliptical Curve algorithms (RFC 4492)
  - Allows equivalent security to current cryptosystems like RSA but smaller key size
  - Favored for mobile technology
Back to the Queue Manager Object

Queue Manager – CCSI Tab

- **Coded Character Set ID**
  - Presented to the SVRCONN channel during connection
  - Same as setting MQCCSID Environment Variable

- **Convert Input**
  - On: Ask the Queue Manager to convert messages using the CCSID (default)
  - Off: No conversion
Queue Manager – MQCSP Tab

- Defines the MQCSP Data Structure
  - Simulates passing MQCSP using MQCONN

- MQCSP User ID
  - Sent via MQCSP if present
  - If blank (and Password Alias set to “none”), no MQCSP is sent

- MQCSP Password Alias
  - Points to the encrypted password stored within DataPower
  - Password is sent in clear text in MQCSP after retrieval from the Alias
MQ Front Side Handler Object

MQ Front Side Handler

<table>
<thead>
<tr>
<th>General</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrative state</td>
<td>enabled</td>
</tr>
<tr>
<td>Comments</td>
<td></td>
</tr>
<tr>
<td>Queue Manager</td>
<td>DEVQMG1</td>
</tr>
<tr>
<td>Get Queue</td>
<td>OPEN</td>
</tr>
<tr>
<td>Put Queue</td>
<td>DROUT</td>
</tr>
<tr>
<td>The number of concurrent IBM MQ conversations</td>
<td>1</td>
</tr>
<tr>
<td>Get Message Options</td>
<td>1</td>
</tr>
<tr>
<td>Polling Interval</td>
<td>20 seconds</td>
</tr>
<tr>
<td>Retrieve Backout Settings</td>
<td>on</td>
</tr>
<tr>
<td>Use Queue Manager in URL</td>
<td>on</td>
</tr>
<tr>
<td>CQS2</td>
<td>0</td>
</tr>
</tbody>
</table>
MQ Front Side Handler

- **Get Queue**
  - Name of queue to get messages from
  - Mandatory, unless Pub/Sub being used

- **Put Queue**
  - Optional, because:
    - May be "one-way" messaging (fire and forget)
    - May be using Reply-To Queue
    - May be dynamically allocated by Response Rule code

- **The number of concurrent MQ conversations:**
  - Number of parallel active and pending MQGETs for the Get Queue
  - Recommend value of 1 (in high throughput situations, may use up to 5)
  - Regardless of this setting, multiple FSH threads will still use multiple connections
  - If greater than 1, monitor Queue Manager for workload
  - If greater than 1, Backout Threshold must be this value plus 1
  - If using message ordering (MQGMO_LOGICAL_ORDER), set it to 1

MQ Front Side Handler

- **Get Message Options**
  - Allows the use of any MQGMO_Options parameters
  - Overrides any specific parameters set elsewhere

- **Polling Interval**
  - How long to wait on an empty queue (seconds)
  - Equivalent to Wait Interval with conventional MQ applications
  - Low value increases network traffic
  - Recommend default of 30

- **Retrieve Backout Settings**
  - Get BOTHRESH and BOQUEUE from the Get Queue
  - Issues MQINQ before every MQGET – potential performance hit
  - Only relevant if queue parameters were set by MQ administrator
  - Recommend set "off" and use Queue Manager Object settings
Some MQGMO Options

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>MQGMO_NONE</td>
</tr>
<tr>
<td>1</td>
<td>MQGMO_WAIT</td>
</tr>
<tr>
<td>2</td>
<td>MQGMO_SYNCPOINT</td>
</tr>
<tr>
<td>4</td>
<td>MQGMO_NO_SYNCPOINT</td>
</tr>
<tr>
<td>8</td>
<td>MQGMO_SET_SIGNAL</td>
</tr>
<tr>
<td>16</td>
<td>MQGMO_BROWSE_FIRST</td>
</tr>
<tr>
<td>32</td>
<td>MQGMO_BROWSE_NEXT</td>
</tr>
<tr>
<td>64</td>
<td>MQGMO_ACCEPT_TRUNCATED_MSG</td>
</tr>
<tr>
<td>128</td>
<td>MQGMO_MARK_SKIP_BACKOUT</td>
</tr>
<tr>
<td>256</td>
<td>MQGMO_MSG_UNDER_CURSOR</td>
</tr>
<tr>
<td>512</td>
<td>MQGMO_LOCK</td>
</tr>
<tr>
<td>1024</td>
<td>MQGMO_UNLOCK</td>
</tr>
<tr>
<td>2048</td>
<td>MQGMO_BROWSE_MSG_UNDER_CURSOR</td>
</tr>
<tr>
<td>4096</td>
<td>MQGMO_SYNCPOINT_IF_PERSISTENT</td>
</tr>
<tr>
<td>8192</td>
<td>MQGMO_FAIL_IF_QUIESCING</td>
</tr>
<tr>
<td>16384</td>
<td>MQGMO_CONVERT</td>
</tr>
<tr>
<td>32768</td>
<td>MQGMO_LOGICAL_ORDER</td>
</tr>
<tr>
<td>65536</td>
<td>MQGMO_COMPLETE_MSG</td>
</tr>
<tr>
<td>131072</td>
<td>MQGMO_ALL_MSGS_AVAILABLE</td>
</tr>
<tr>
<td>262144</td>
<td>MQGMO_ALL_SEGMENTS_AVAILABLE</td>
</tr>
</tbody>
</table>

MQ Front Side Handler

- **Use Queue Manager in URL**
  - Defines the behavior of var://service/URL-in when a QM Group is specified
  - If on, the variable returns the name of the chosen Queue Manager
  - If off, the variable returns the name of the Queue Manager Group
  - Default off

- **CCSI**
  - Sets the CCSID in the MQ Message Descriptor
  - If blank or zero, default is ISO-8859-1 (latin-1)
  - For MQCCSI_EMBEDDED enter 4294967295
  - For MQCCSI_INHERIT enter 4294967294
MQ Front Side Handler

Publish and Subscribe

Subscribe Topic String

Subscription Name

Publish Topic String

Properties and Headers

Parse Properties

Selector

Exclude Message Headers

CICS Bridge Header (MQCH1)
Dead Letter Header (MQDLH)
MQS Information Header (MQQTH)
Queues and Formatting Header (MQRFH)
Rules and Formatting Header (MQRFH2)
Work Information Header (MQWH)

Header to extract Content-Type

Advanced

Async Put

Batch Size

Subscribe Topic String

Pub/Sub topic string for subscription

If Get Queue also defined, this is ignored

Subscription Name

Used to establish or resume a Durable Subscription

Publish Topic String

Pub/Sub topic string for response publication

If Put Queue also defined, this is ignored

Parse Properties

Extracts MQ V7 (and above) Message Properties into Node Set

Minor overhead, so leave off unless needed
MQ Front Side Handler

- **Selector**
  - Allows selective retrieval of messages based on properties
  - Forces sequential search of queue so may be inefficient

- **Exclude Message Headers**
  - Strip off selected MQ header types

- **Header to extract Content-Type**
  - Can obtain Content-Type from
    - MQMD
    - RFH
    - RFH2

MQ Front Side Handler

- **Async Put**
  - Put message to queue without waiting for a response
  - Do not use when Queue Manager units-of-work is 1
  - Recommend only use where performance is an issue

- **Batch Size**
  - Number of messages to handled as a single commit or rollback operation
  - Recommend leave this as zero – each message is a separate transaction
MQ Back-End URL

- **General Syntax:**
  - `dpmq://mqQueueManagerObject/URI?<parameters>`

- **RequestQueue**=`requestQueueName`
  - Name of the backend MQ request queue

- **ReplyQueue**=`replyQueueName`
  - Name of the backend MQ reply queue

- **Sync**=`true`
  - Issues a Commit call when a message is put on Request Queue

- **GMO**=`optionsValue`
  - MQGMO_Options parameter value when getting from Reply Queue

- **PMO**=`optionsValue`
  - MQPMO_Options parameter value when putting to Request Queue
### MQ Back-End URL

- **ParseHeaders={true|false}**
  - Specifies whether to parse and strip headers from message

- **SetReplyTo={true|false}**
  - Specifies whether to set MD ReplyToQ during Put

- **AsyncPut={true|false}**
  - Specifies whether to use Asynchronous Put
  - Only valid when using MQ V7 (and above)

- **Browse={first|next|current}**
  - Controls non-destructive retrieval of messages

---

### MQ Back-End URL

- **ContentTypeHeader=** *header*
  - Which MQ header identifies the content type of the message

- **ContentTypeXPath=** *expression*
  - XPath expression to extract the content type of message

- **ParseProperties={on|off}**
  - Parse message properties

- **PublishTopicString=** *string* and **SubscribeTopicString=** *string*
  - Specifies topic to use with Pub/Sub (MQ V7 and above)

- **SubscriptionName=** *string*
  - Specifies name for a durable subscription (MQ V7 and above)

- **Selector=** *expression*
  - SQL92 style query filtering on message properties
  - Performance hit
Multi-Protocol Gateway Parameter Settings

MPGW Headers Tab – MQ Headers

<table>
<thead>
<tr>
<th>Direction</th>
<th>Header Name</th>
<th>Header Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Direction</th>
<th>Header Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td></td>
</tr>
</tbody>
</table>
Using Header Injection (Header Tab)

Example setting Format and Persistence:
- Direction: Front (for FSH MQPUT)
- Direction: Back (for Backend MQPUT)
- Header Name: MQMD
- Header Value:
  
  ```xml
  <MQMD><Format>MQSTR</Format><Persistence>1</Persistence></MQMD>
  ```
Using the Stylesheet method (page 1 of 2)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:dp="http://www.datapower.com/extensions"
    extension-element-prefixes="dp"
    exclude-result-prefixes="dp">
    <xsl:output method="xml"/>
    <xsl:template match="/">
        <xsl:variable name="newMQMDStr"/>
        <MQMD>
            <Format>MQSTR</Format><Persistence>1</Persistence>
        </MQMD>
        <xsl:variable name="mqmdStr"/>
        <dp:serialize select="$newMQMDStr" omit-xml-decl="yes"/>
        <xsl:variable name="message" dp:prioriy="debug">
            <xsl:value-of select="concat('The New MQMD : ', $mqmdStr')"/>
        </xsl:variable>
    </xsl:template>
</xsl:stylesheet>
```

Using the Stylesheet method (page 2 of 2)

```xml
<xsl:message>
<!-- for request rule -->
<dp:set-request-header name="MQMD" value="$mqmdStr"/>
<!-- for response rule -->
<!-- adding MQ header when MQ URL open call is used for MQPUT -->
<!--
    <xsl:variable name="mqHeaders">
        <header name="MQMD">$mqmdStr</header>
    </xsl:variable>
    <xsl:variable name="sendmessage">
        <dp:post-to open
            target="dpmq://DF4/?RequestQueue=QUEUE6"
            http.headers="$mqHeaders"
            response="responsecode-ignore">
            <xsl:copy-of select="."/>
        </dp:post-to>
    </xsl:variable>
-->
</xsl:message>
```
MQ Headers – Programmatic Manipulation

- Context variable method to inject the MQMD header
  - For the following code to work:
  - Set Transform Action's OUTPUT context to “EVENTS”
  - Set Result Action's INPUT context to “EVENTS”

```xml
<xsl:variable name="MQMDStr"></xsl:variable>
<xsl:variable name="MQMDStr2"></xsl:variable>
<xsl:variable name="MQMDStr" select="$MQMDStr"/>
<xsl:variable name="MQMDStr2" select="$MQMDStr2"/>
```
MQ Headers – Programmatic Manipulation

- **JMS Headers as RFH2**
  - Must set FSH “Parse Properties” to be “off”
  - Must set “Exclude RFH2” to be “on”
  - Message Properties appear as “MQRFH2” headers

```xml
<X-EnterpriseMQRFH2-Data-0>
  <jms:MessageData>
    <TFID>1474922592342</TFID>
  </jms:MessageData>
</X-EnterpriseMQRFH2-Data-0>
```

MQ Error Handling

- **MQ error handling example:**

```xml
<xsl:template match="/">
  <xsl:variable name="error" select="dp:response-header('x-dp-response-code')"/>
  <xsl:variable name="errMsg" select="concat('** The Response Code ** : ', $error, ' and ** Error Code ** : ', $error)"/>
  
  <xsl:choose>
    <xsl:when test="starts-with($error, '2') and (string-length(normalize-space($error))=4) or ($error != '4040000000')">
      <xsl:message dp:priority="debug">
        <xsl:value-of select="$errorMsg"/>
      </xsl:message>
      <xsl:set-variable name="error:context\/@ERROR/err-msg" select="$errorMsg"/>
      <xsl:reject override="true" xsl:value-of select="$errorMsg"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:message dp:priority="debug">
        <xsl:value-of select="$errorMsg"/>
      </xsl:message>
      <xsl:accept/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>
```
MQ Conversational Processing

- Backend application must copy MsgId to CorrelId
  - DataPower Back-End retrieves reply using CorrelId

- MQPUT1 Simulation
  - Create MQ Object Descriptor header with Queue Manager name in it
  - Request Rule issues MQOPEN/MQPUT/MQCLOSE to back end Queue Manager

- ReplyToQ Usage
  - If set, Response Rule sends message there

- ReplyToQmgr Usage
  - Can be set to send to a different Queue Manager
  - If destination is a Cluster, no need to supply ReplyToQmgr

---

MQ Conversational Processing

- XSL code snippet to set ReplyToQ and ReplyToQmgr in a Request Rule:

```xml
<xsl:variable name="rule-type" select="dp:variable('var://service/transaction-rule-type')"/>
<xsl:choose>
  <xsl:when test="rule-type = 'request'">
    <xsl:variable name="entries" select="dp:request-header('MOMD')"/>
    <xsl:variable name="header" select="dp:parse($entries)"/>
    <xsl:variable name="ReplyToQ" select="$header//ReplyToQ"/>
    <xsl:variable name="ReplyToQmgr" select="$header//ReplyToQmgr"/>
    <xsl:message dp:priority="debug">
      <xsl:value-of select="concat('ReplyToQ : ', $ReplyToQ)"/>
      <xsl:value-of select="concat('ReplyToQmgr : ', $ReplyToQmgr)"/>
    </xsl:message>
  </xsl:when>
</xsl:choose>
```
MQ Conversational Processing

- XSL code snippet to set ReplyToQ and ReplyToQmqr in a Response Rule:

```xml
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" indent="yes" encoding="UTF-8"/>

  <xsl:template match="/">
    <xsl:variable name="replyToQ" select="/ReplyToQ"/>
    <xsl:variable name="replyToQmgr" select="/ReplyToQMgr"/>

    <xsl:for-each select="/Response">
      <xsl:if test="@rule-type = 'response'">
        <xsl:choose>
          <xsl:when test="@rule-type = 'response'">
            <xsl:variable name="customMQODStr">
              <![CDATA[<![CDATA["$replyToQ","$replyToQmgr"]]>]]>
            </xsl:variable>

            <xsl:value-of select="concat("Response MQOD : ", $customMQODStr)="/">
          </xsl:when>
        </xsl:choose>
      </xsl:if>
    </xsl:for-each>
  </xsl:template>
</xsl:stylesheet>
```

Transactional Processing
### MQ Transactional Processing

- **DataPower is a standard MQ Client**
  - It does **NOT** offer Extended Transactional Client functionality
  - **NO XA** two-phase commit
- **DataPower is considered an application by MQ**
  - Therefore, no inherent message integrity
- **If the same Queue Manager at front and back:**
  - True message integrity
  - Once and once-only delivery
- **If different Queue Managers at front and back**
  - No possibility of two-phase commit
  - Message integrity assured if DataPower configured properly
  - Possibility of messages sent more than once

---

### Scenario: Two Different Queue Managers

**No Transactional Control**

![Diagram showing message flow between two different queue managers](image.png)
Scenario: Two Different Queue Managers

Front Side Transactional Control only

App 1

Units of Work = 1

FS.OUT  FS.IN

Queue Manager QM1

DataPower

Request Rule

Response Rule

Sync=false; Transactional=false

App 2

Queue Manager QM2

BE.OUT

BE.IN

Scenario: Two Different Queue Managers

Back End Transactional Control only

App 1

Units of Work = 0

FS.OUT  FS.IN

Queue Manager QM1

DataPower

Request Rule

Response Rule

Sync=false; Transactional=true

App 2

Queue Manager QM2

BE.OUT

BE.IN
Scenario: Two Different Queue Managers

Front Side and Back End Transactional Control

- App 1: Units of Work = 1, Sync=false; Transactional=true
- App 2: Sync=false; Transactional=true

Scenario: One Queue Manager (Front & Back)

No Transactional Control

- App 1: Units of Work = 0, Sync=false; Transactional=false
- App 2: Sync=false; Transactional=false
Capitalware's MQ Technical Conference v2.0.1.6

Scenario: One Queue Manager (Front & Back)

Front Side Transactional Control only

- App 1
  - FS.OUT
  - FS.IN
  - Units of Work = 1
- Queue Manager QM1
- DataPower
  - Request Rule
  - Response Rule
- App 2
  - BE.OUT
  - BE.IN

Sync=false; Transactional=false

Scenario: One Queue Manager (Front & Back)

Front Side Transactional Control only

- App 1
  - FS.OUT
  - FS.IN
  - Units of Work = 1
- Queue Manager QM1
- DataPower
  - Request Rule
  - Response Rule
- App 2
  - BE.OUT
  - BE.IN

Sync=true; Transactional=false
Scenario: One Queue Manager (Front & Back)

Front Side and Back End Transactional Control

Queue Manager QM1

FS.OUT  FS.IN  BE.OUT  BE.IN

DataPower
Request Rule
Response Rule

Sync=true; Transactional=true
Units of Work = 1

App 1
App 2

Poison Message

Queue Manager QM1

FS.ERR  FS.IN

DataPower
Request Rule

FSH

BOTHRESH = 3
BOQUEUE = FS.ERR
Units of Work = 1
Poison Message

**DataPower Knowledge Center Documentation**


“If there are no backout settings, the backout function is disabled.”
Questions & Answers

End of Session

Thank You!

Contact: Robin@RobinWileyTraining.com