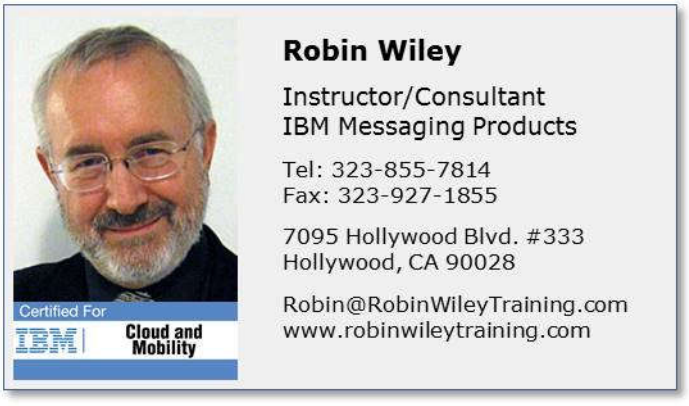
DataPower-MQ Connectivity Lab Instructions

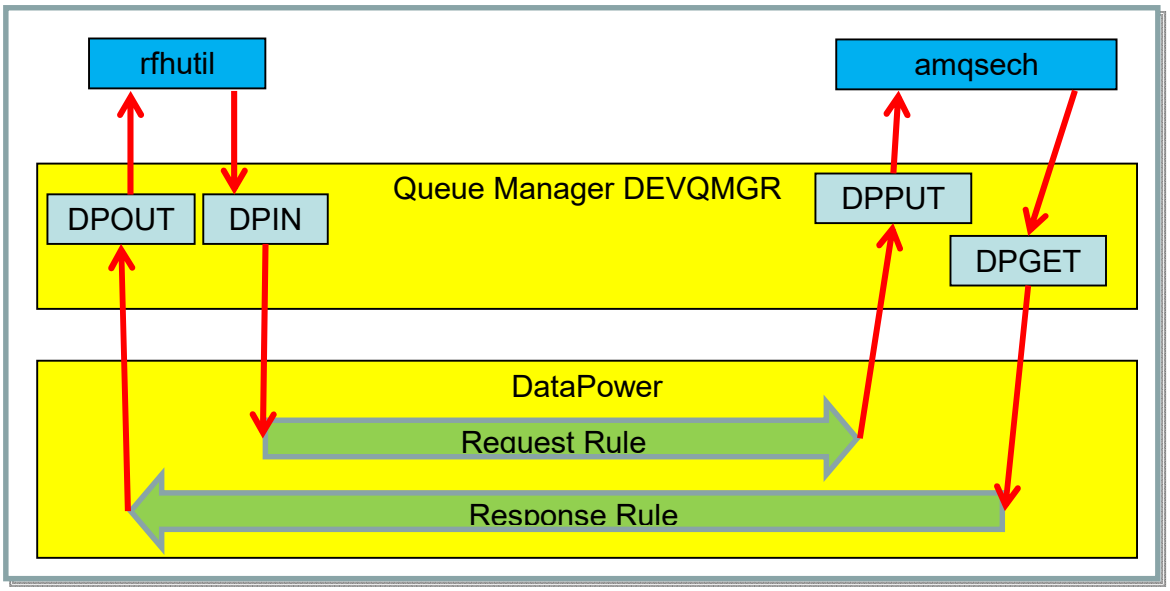
Labs provided courtesy of:



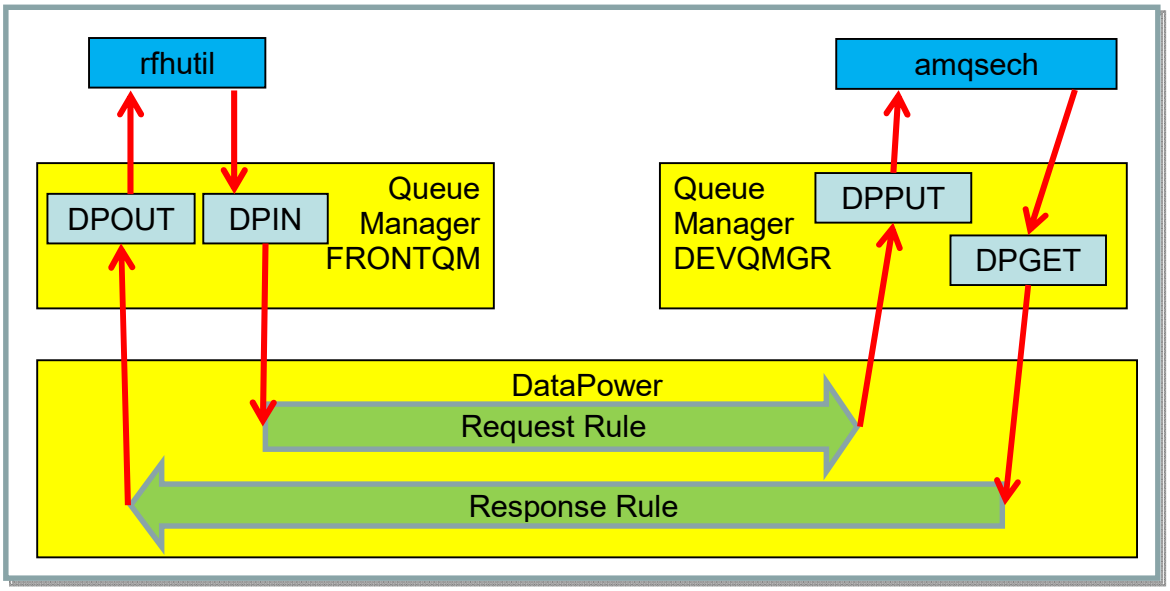
1 Exercise 1: Introduction and Verification

1.1 Infrastructure

The infrastructure for labs 2, 3, and 5 consists of a single Queue Manager called "DEVQMGR" providing queues for both the front and back ends of a Multi-Protocol Gateway (MPG) called "MQtest.". The "`rfhutil`" and "amqsech" programs will be used to send and receive messages via these queues into the Request and Response Rules of the MPG.
This is illustrated in the following diagram:



The infrastructure for lab 4 consists of two Queue Managers providing queues for the front and back ends separately, as shown in the following diagram:



1.2 Windows Logon

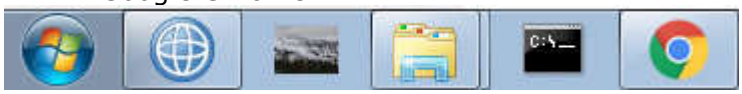
If your Windows image is not already logged on, use the following:

Username: Administrator
Password: web1sphere

1.3 Windows Icons

Several icons have been added to the Task Bar in order to simplify running some of the applications needed for these labs. They are shown in the following screen shot, reading from left to right:

- Windows Start Menu
- MQ Explorer
- Rfhutil
- Windows Explorer
- Command Prompt
- Google Chrome



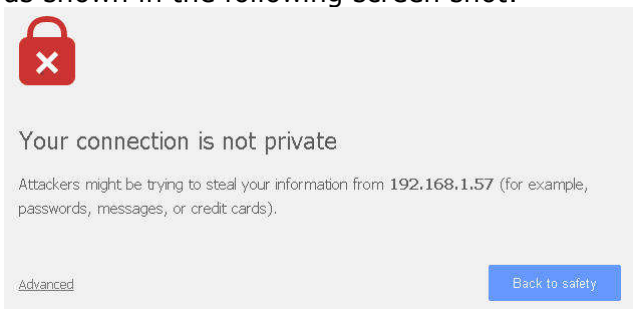
(There are also corresponding icons on the Desktop.)

1.4 DataPower Logon

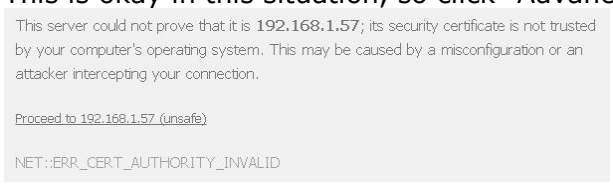
Run Google Chrome and enter the following URL:

`https://192.168.1.57:9090`

The first time you connect to DataPower, you will get a certificate mismatch warning, as shown in the following screen shot:



This is okay in this situation, so click "Advanced" then "Proceed...":

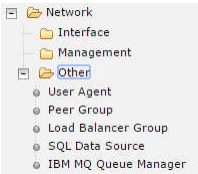


Logon to DataPower with the following information:

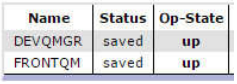
Username: student01
Password: stud1
Domain: student01_domain
Graphical Interface: WebGUI

1.5 Verify Queue Manager Objects

In the left-side menu, navigate to Network -> Other -> IBM MQ Queue Manager:



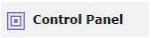
Verify that the objects are running by observing the status of "[up]" to the right of each object name as shown in this screen shot:



Name	Status	Op-State
DEVQMGR	saved	up
FRONTQM	saved	up

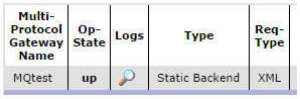
1.6 Verify Multi-Protocol Gateway

Click back to the Control Panel:



Click on the Multi-Protocol Gateway icon:

Look for the "MQtest" Multi-Protocol Gateway in the list and verify the "Op-State" is "up" as shown in this screen shot:

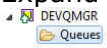


Multi-Protocol Gateway Name	Op-State	Logs	Type	Req-Type
MQtest	up		Static Backend	XML

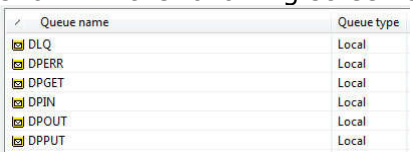
1.7 Verify MQ Infrastructure

Click the MQ Explorer icon on the Task Bar, or Double-click the "MQ Explorer" icon on the desktop:

Expand the "DEVQMGR" folder and click on the "Queues" folder:



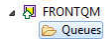
Look in the Content panel to the right and verify that 6 Local Queues are defined as shown in the following screen shot:



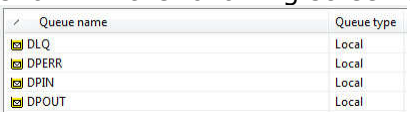
Queue name	Queue type
DLQ	Local
DPERR	Local
DPGET	Local
DPIN	Local
DPOUT	Local
DPPUT	Local

Check the "Current queue depth" for each of these queues. If any are non-zero, clear the messages from the queue so you have a clean starting environment.

Expand the "FRONTQM" folder and click on the "Queues" folder:



Look in the Content panel to the right and verify that 4 Local Queues are defined as shown in the following screen shot:



Queue name	Queue type
DLQ	Local
DPERR	Local
DPIN	Local
DPOUT	Local

Check the "Current queue depth" for each of these queues. If any are non-zero, clear the messages from the queue so you have a clean starting environment.

End of Exercise 1

2 Exercise 2: Echo Test

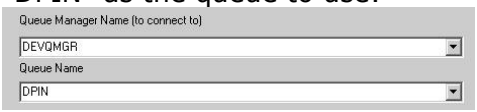
The first part of this exercise will simply demonstrate passing a single MQ message from the front side to the back side of a Multi-Protocol Gateway. Next, it will use a triggered "echo" program to demonstrate the default behavior of the Front Side Handler (FSH) in relation to the "Reply To Queue" field in the Message Descriptor, and will demonstrate the use the "MQ Header" Action to affect this behavior.

2.1 Check Triggering

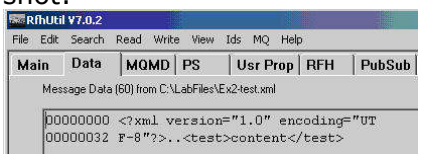
In MQ Explorer, click on the Queues folder, then right-click "DPPUT" and choose "Properties." Click on the "Triggering" menu item in the left side of the panel and verify that "Trigger control" is currently "Off." (You will turn triggering on later in this exercise.)

2.2 Send a Message

Load the "**`rfhutil`**" program by clicking its icon () on the task bar or the desktop. When the program loads, select "DEVQMGR" as the Queue Manager and "DPIN" as the queue to use:



Click the "Open File" button, navigate to C:\LabFiles, and choose "Ex2-test.xml" to be loaded. Click the "Data" tab and verify that the text is as shown in this screen shot:



Click the "Main" tab, then click "Write Queue":

2.3 Read a Message

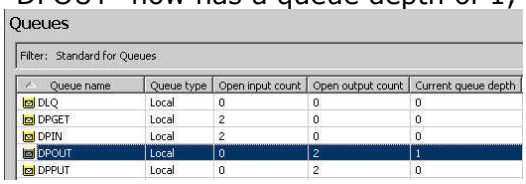
Click on the Command Prompt icon () on the Task Bar or double-click the Command Prompt icon on the desktop. Enter following command:

```
amqsget DPPUT DEVQMGR
```

This program will retrieve the message destructively from the queue and display it as shown in the following screen shot. After the message is displayed, the program will time out after 20 seconds and exit:

2.4 Error Message

Wait at least 30 seconds, then go to MQ Explorer and click on the "Queues" folder in the navigator panel. Click the Content Panel "Refresh" icon ( ) and notice how "DPPUT" now has a queue depth of zero, as would be expected. But notice how "DPOUT" now has a queue depth of 1, as shown in the following screen shot:



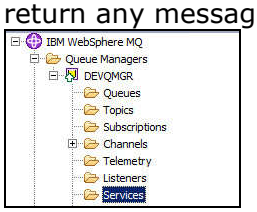
This is because the back side timeout parameter is set to 30 seconds, and no reply message was received within that time. As a result, the DataPower back side MQ interface creates a null message in reply as a signal that it timed out with no reply. If you were to examine the Correlation ID of this null message, you would see it has the same value as the Message ID of the original test message sent to "DPIN." Return to the Command Prompt and enter following command:

```
amqsget DPOUT DEVQMGR
```

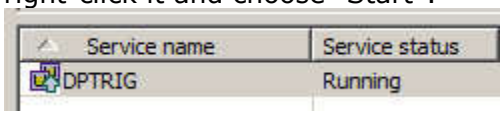
This program will retrieve the error message destructively from the queue and display it as shown in the following screen shot. Notice that the content of the message is empty, as indicated by nothing appearing between the "<" and the ">":

2.5 Set Up the Echo Service

In this part of the exercise, you will enable a backend "echo" application that will return any message it receives. In MQ Explorer, click on the Services folder:



In the content panel to the right, check that the "DPTRIG" service is running. If not, right-click it and choose "Start":



If you don't see the DPTRIG service at all, you will need to set it up. See Appendix 7 for screenshots of the necessary parameters.

In MQ Explorer, click on the Queues folder, then right-click "DPPUT" and choose "Properties." Click on the "Triggering" menu item in the left side of the panel and select "On" in the "Trigger control" pick list. Then click "Ok":

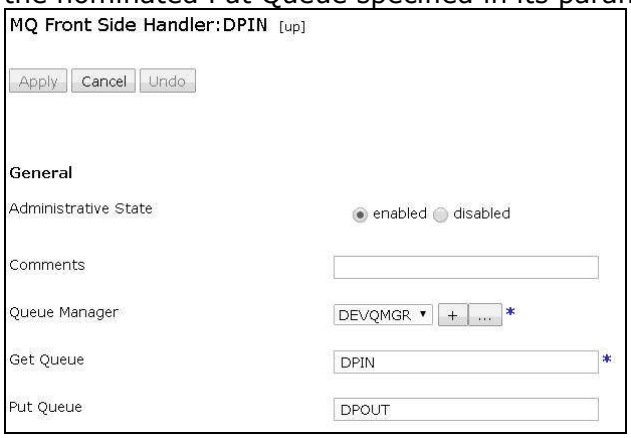
2.6 Send Message using rfhutil

Return to **`rfhutil`** and click on the MQMD tab and choose "MQSTR" as the Message Format, "Request" as the Message Type, and enter "DPGET" in the Reply-To Queue field. Leave the "Reply To Queue Manager" field blank:

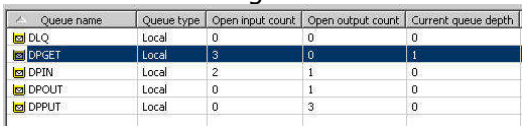
Click the "Main" tab, then click "Write Queue":

2.7 Find the Message

When the message arrives on the queue "DPPUT" at the back end, the trigger monitor service will run the "amqsech" sample program, which will simply echo the message back to the nominated Reply To Queue – in this case, "DPGET."
The MPG back side URL will then retrieve the message and pass it through the Response rule, handing it back to the Front Side Handler, which should put it onto the nominated Put Queue specified in its parameters, as seen in this screen shot:



However, if you look at the queue display in MQ Explorer, you will see instead that there's one message on DPGET but no messages on DPOUT, as in this screen shot:



Queue name	Queue type	Open input count	Open output count	Current queue depth
DLQ	Local	0	0	0
DPGET	Local	3	0	1
DPIN	Local	2	1	0
DPOUT	Local	0	1	0
DPPUT	Local	0	3	0

Why is this? There are two possibilities:

- The MPG Back End did not retrieve the message from DPGET.
- The Front Side Handler put the message onto DPGET instead of DPOUT.

In order to troubleshoot this problem, let's look at the message in detail. In the Command Prompt, enter the following command:

amqsbcg DPGET DEVQMGR

This will display the full contents of the Message Descriptor, as shown here:

```
Select Command Prompt

C:\>amqsbcg DPGET DEVQMGR

AMQSBCG0 - starts here
**********************

 MQOPEN - 'DPGET'


 MQGET of message number 1, CompCode:0 Reason:0
****Message descriptor****

  StrucId  : 'MD  '  Version : 2
  Report   : 0  MsgType : 2
  Expiry   : -1  Feedback : 0
  Encoding : 546  CodedCharSetId : 819
  Format : 'MQSTR   '
  Priority : 0  Persistence : 0
  MsgId : X'414D5120444556514D47522020202020F9FC41542000DE02'
  CorrelId : X'414D5120444556514D47522020202020F9FC41542000240A'
  BackoutCount : 0
  ReplyToQ       : 'DPGET                                          '
  ReplyToQMgr    : 'DEVQMGR                                        '
  ** Identity Context
  UserIdentifier : 'MUSR_MQADMIN'
  AccountingToken :
   X'160105150000009B0BC11EE12A9375235F636BEF030000000000000000000000B'
  ApplIdentityData : '                                '
  ** Origin Context
  PutApplType    : '6'
  PutApplName    : 'WebSphere Datapower MQClient'
  PutDate   : '20141018'   PutTime   : '07542268'
  ApplOriginData : '    '

  GroupId : X'000000000000000000000000000000000000000000000000'
  MsgSeqNumber   : '1'
  Offset         : '0'
  MsgFlags       : '0'
  OriginalLength : '-1'

****   Message      ****

 length - 59 of 59 bytes

00000000:  3C3F 786D 6C20 7665 7273 696F 6E3D 2231 '<?xml version="1'
00000010:  2E30 2220 656E 636F 6469 6E67 3D22 5554 '.0" encoding="UT'
00000020:  462D 3822 3F3E 0A3C 7465 7374 3E63 6F6E 'F-8"?>.<test>con'
00000030:  7465 6E74 3C2F 7465 7374 3E              'tent</test>    '


 No more messages
 MQCLOSE
 MQDISC
C:\>_
```

Notice the "PutApplName" field says "WebSphere Datapower MQClient" -- this indicates that it was DataPower that put the message onto that queue, and not **amqsech**.

The reason this happened is because **amqsech** copies the Reply To Queue name from the incoming message into the Reply To Queue field of the message it echoes. This can be seen in the screen shot above, where the "ReplyToQ" value is "DPGET."

The default behavior of the Front Side Handler is to send the message to the Reply To Queue (if one is present) instead of the nominated Put Queue. Only if the Reply To Queue in the message is blank does it use the Put Queue value.

Use MQ Explorer to clear the DPGET queue before proceeding.

2.8 Choices to Fix Front Side Handler Behavior

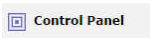
There are several ways this problem can be fixed:

- Ensure the backend program blanks out the Reply To Queue field in the MQMD
- Have the backend program insert the correct name of the destination queue in the Reply To Queue field in the MQMD.
- Have the Response Rule in the MPG either blank the Reply To Queue field or supply the correct value.

2.9 Edit the Multi-Protocol Gateway Policy

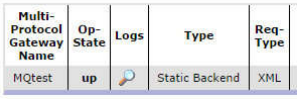
In this exercise, you will use an MQ Header Action in the Response Rule to set the Reply To Queue to the correct value.

In the browser WebGUI interface, click back to the Control Panel:



Click on the Multi-Protocol Gateway icon:

Click on the "MQtest" Multi-Protocol Gateway in the list:



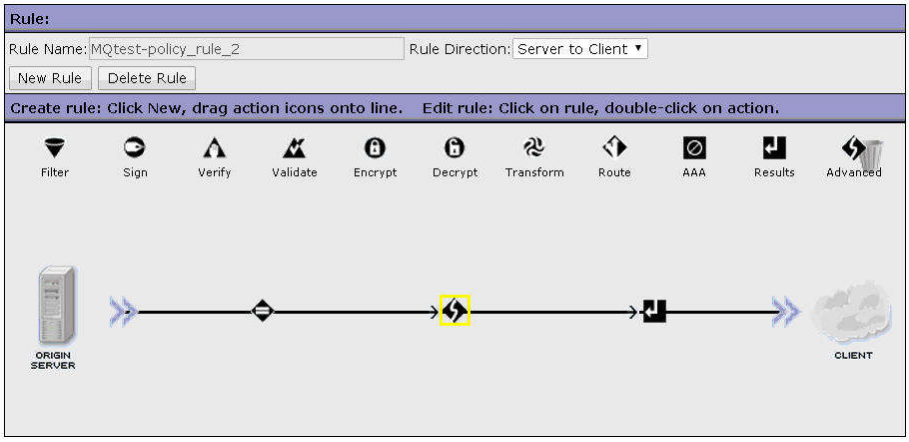
Multi-Protocol Gateway Name	Op-State	Logs	Type	Req-Type
MQtest	up		Static Backend	XML

Click on the ellipsis (...) to the right of the "MQtest-policy" Multi-Protocol Gateway Policy in the list:

In the "Configured Rules" panel at the bottom of the screen, click on "MQtest-policy_rule_2" so it is selected (the text will become bold when it is selected):

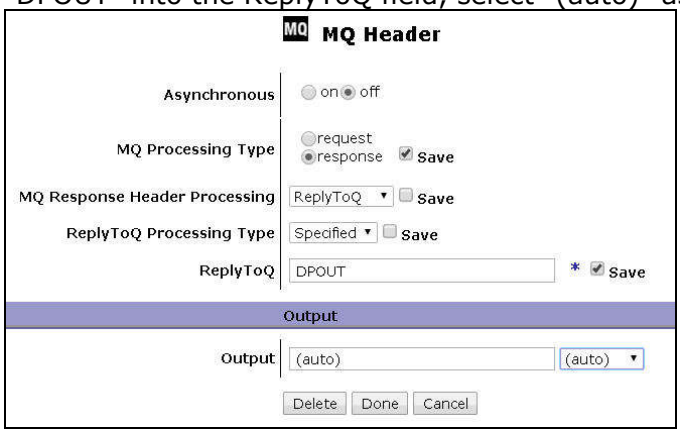
Configured Rules		
Order	Rule Name	Direction
⇧⇩	MQtest-policy_rule_1	Client to Server
⇧⇩	**MQtest-policy_rule_2**	**Server to Client**

In the rule edit panel in the center of the window, click and drag the Advanced Action icon to the rule bar between the Match Action and the Results Action:

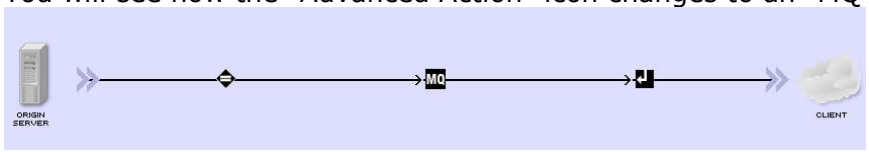


Double-click the "Advanced Action" icon, choose "MQ Header," then click "Next":

Check the "response" radio button, select "ReplyToQ" from the pick list, enter "DPOUT" into the ReplyToQ field, select "(auto)" as the Output, then click "Done":



You will see how the "Advanced Action" icon changes to an "MQ Action" icon:



Click "Apply Policy" in the upper left corner of the Policy editor window, then click the "Close Window" link toward the upper right area of the Policy editor (do not click the Windows "X" in upper right corner). Click "Apply" in the MPG configuration editor, then click the "Save Configuration" link in the right side of the DataPower status bar.

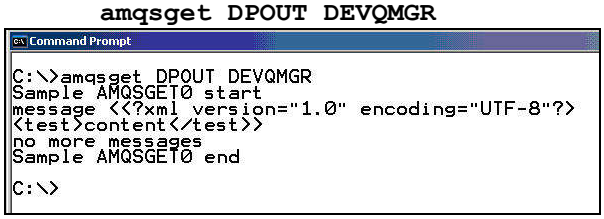
Return to **`rfhutil`** and click "Write Queue":

Refresh the queue display in MQ Explorer and note that there is now one message on "DPOUT":

Queue name	Queue type	Open input count	Open output count	Current queue depth
DLQ	Local	0	0	0
DPGET	Local	4	0	0
DPIN	Local	3	1	0
DPOUT	Local	0	2	1
DPPUT	Local	0	4	0

Read the message destructively from the queue by using the following command in the Command Prompt:

```
amqsget DPOUT DEVQMGR
```



End of Exercise 2

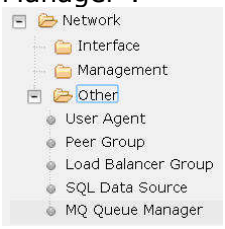
3 Exercise 3: Syncpoint and Poison Messages

In this exercise, you will use syncpoint processing to see how DataPower Policy Rules handle errors by rolling back current MQ transactions. You will also set a threshold and a backout queue to prevent a continuous read-rollback loop, commonly known as a "poison message."
There are two places where the "Backout Threshold" and "Backout Requeue Queue" can be specified:

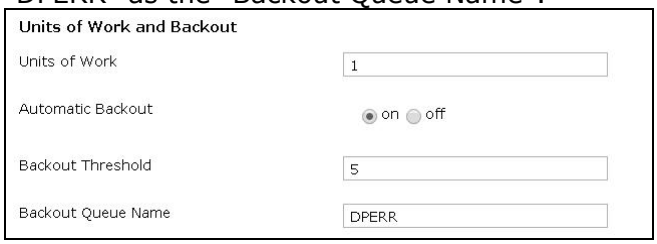
- In the Queue Manager Object within DataPower. One set of parameters applies to all queues opened on that Queue Manager. This is set by the DataPower administrator or developer.
- In the Local Queue definition on the Queue Manager itself. A different set of parameters can be defined for each separate Local Queue. This is set by the MQ administrator or a suitably authorized developer.

3.1 Set up Syncpoint on the Queue Manager Object

Using the DataPower menu, navigate to "Network" -> "Other" -> "MQ Queue Manager":



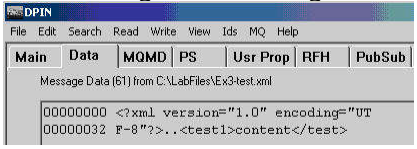
Click on "DEVQMGR" from the list of Queue Manager Objects, scroll down to "Units of Work" and change the value from "0" to "1." You will need to click into a white space area on the screen in order to make the "Automatic Backout" radio button appear. Click "on" for "Automatic Backout" then set the "Threshold" to "5" and enter "DPERR" as the "Backout Queue Name":



Click "Apply" and "Save Configuration."

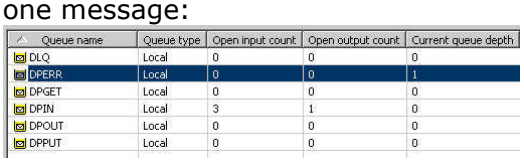
3.2 Send a Bad Message

Return to "**`rfhutil`**," click the "Open File" button, navigate to C:\LabFiles, and choose "Ex3-test.xml" to be loaded. Click the "Data" tab and verify that the text is as shown in this screen shot. Note that the XML is malformed in that the opening and closing element tags do not match:



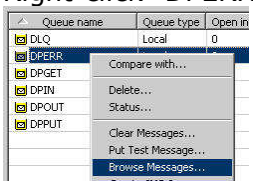
Click the "Main" tab, verify that the Queue Name is still "DPIN," then click "Write Queue":

Return to MQ Explorer, refresh the Queues display, and note that "DPERR" now has one message:

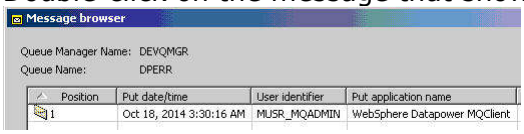


Queue name	Queue type	Open input count	Open output count	Current queue depth
DLQ	Local	0	0	0
DPERR	Local	0	0	1
DPGET	Local	0	0	0
DPIN	Local	3	1	0
DPOUT	Local	0	0	0
DPPUT	Local	0	0	0

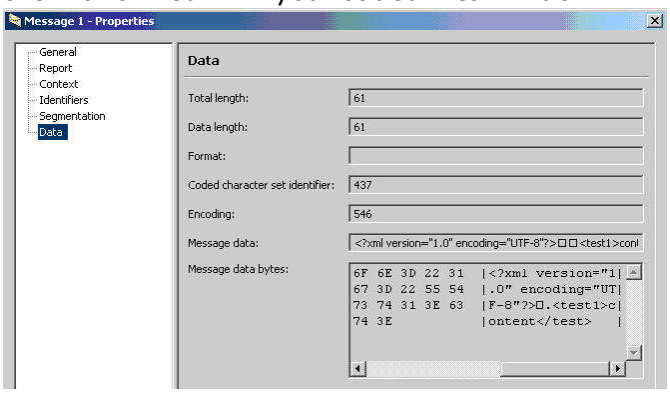
Right-click "DPERR" and choose "Browse Messages":



Double-click on the message that shows in the popup panel:

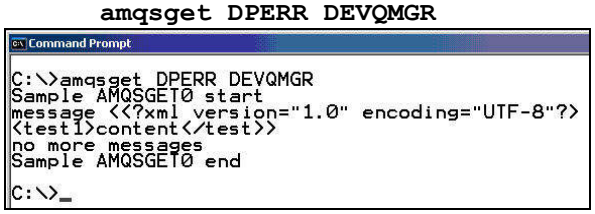


Click the "Data" menu item in the "Message Properties" panel, and verify the data is the malformed XML you loaded into **`rfhutil`**:



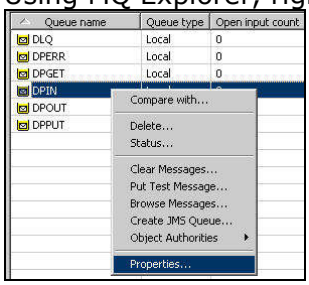
Close this panel and the Message Browser panel. Next, delete the message from the "DPERR" queue by using MQ Explorer or by running the following command in the Command Prompt:

```
amqsget DPERR DEVQMGR
```

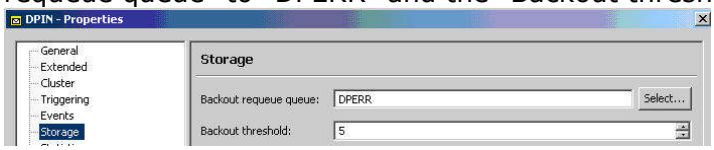


3.3 Set Backout Parameters on the Queue

Using MQ Explorer, right-click on "DPIN" and choose "Properties":

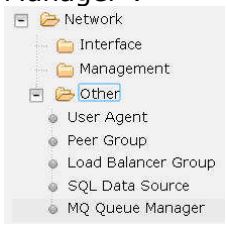


Click the "Storage" menu item at the left of the "Properties" panel, set the "Backout requeue queue" to "DPERR" and the "Backout threshold" to "5," then click "Ok":

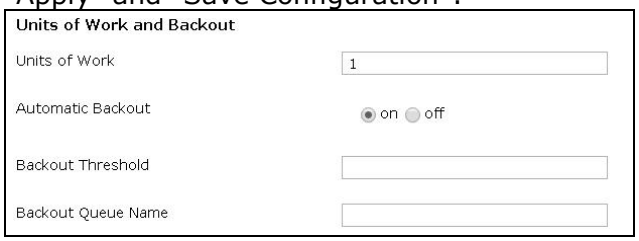


3.4 Remove the Backout Parameters from the Queue Manager Object

Using the DataPower menu, navigate to "Network" -> "Other" -> "MQ Queue Manager":



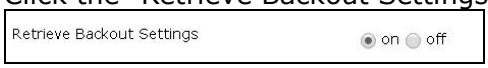
Click on "DEVQMGR" from the list of Queue Manager Objects, scroll down to "Units of Work" and blank out the values in "Threshold" and "Backout Queue Name." Click "Apply" and "Save Configuration":



3.5 Set the Front Side Handler to retrieve Backout settings

In the DataPower web page, navigate to the "MQtest" MPG, and select the "DPIN" Front Side Handler. Click the edit button with the ellipsis (...) to the right of the pick list to edit it.

Click the "Retrieve Backout Settings" radio button on:



Click "Apply" on the FSH, "Apply" the MPG, then "Save Configuration."

3.6 Send another Bad Message

Return to "**`rfhutil`**" and click "Write Queue":

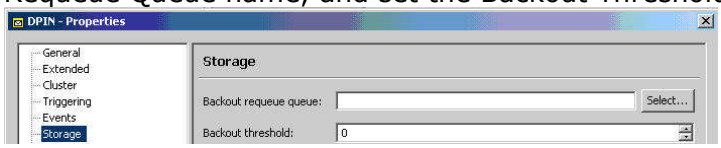
Return to MQ Explorer, refresh the Queues display, and note that "DPERR" again has one message:

Queue name	Queue type	Open input count	Open output count	Current queue depth
DLQ	Local	0	0	0
DPERR	Local	0	0	1
DPGET	Local	0	0	0
DPIN	Local	3	1	0
DPOUT	Local	0	0	0
DPPUT	Local	0	0	0

If you wish to look at the message, follow the instructions above to browse it. Be sure to delete it when you're done.
In this case, the FSH retrieved the backout settings from the queue itself and applied them to the bad message.

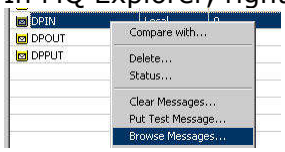
3.7 Poison Message

In the next example you will create a situation where there are no backout parameters in either the Queue Manager object or the Local Queue. When you send the bad message, it will continually be rolled back and re-read.
Return to MQ Explorer, right-click on the DPIN queue, blank out the Backout Requeue Queue name, and set the Backout Threshold to zero:

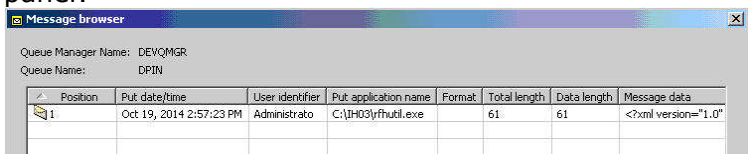


Return to "**`rfhutil`**" and send the bad message again by clicking "Write Queue":

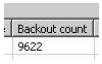
In MQ Explorer, right-click on "DPIN" and select "Browse Messages":



The Browse Message panel pops up, but it may not show the message since the FSH may have read it from the queue and be processing it at this time.
Click the "Refresh" button () repeatedly until you see a message appear in the panel:

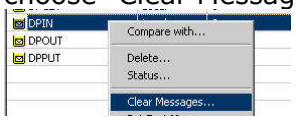


Scroll the message display to the right until you see the column "Backout Count":

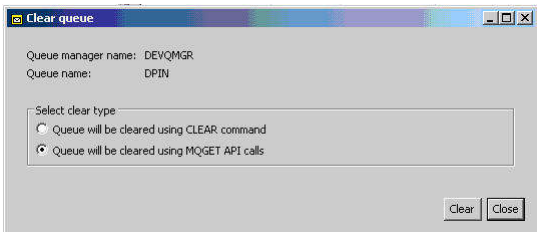


Note the number shown, click "Refresh" and note the number again:

You will notice the number increase in value each time you click "Refresh" since the FSH is repeated reading and rolling the message back. This will continue unchecked until the message is removed message permanently from the queue. You will do this using MQ Explorer. Close the Message Browser panels, right click on "DPIN," and choose "Clear Messages":



Select the radio button "Queue will be cleared using MQGET API calls" and click "Clear":



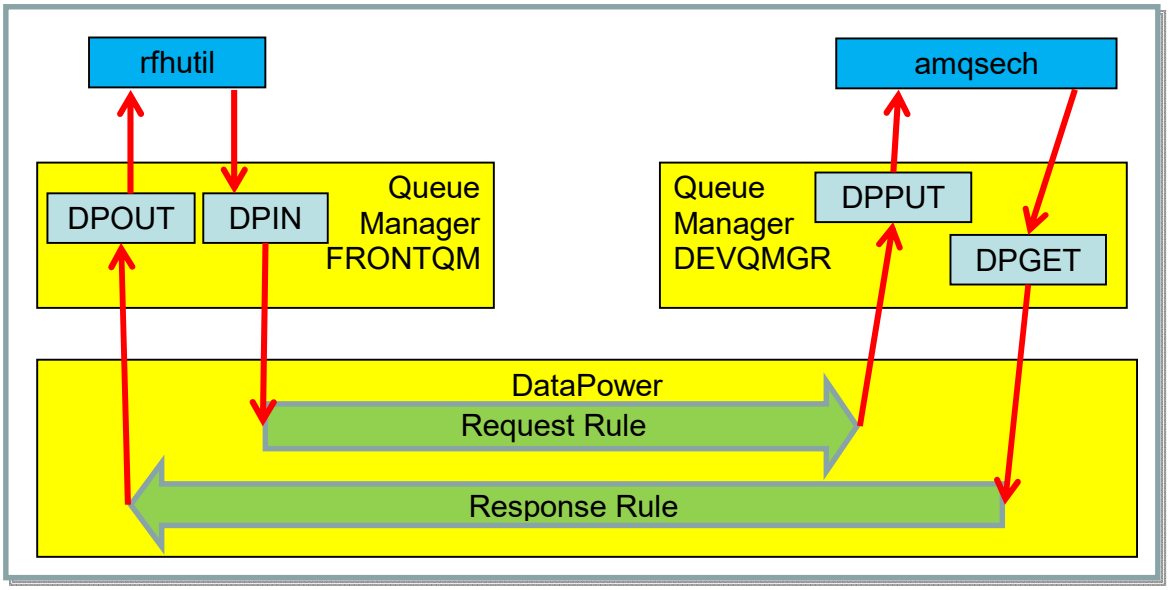
Note that you may need to repeat this several times in order to really clear the queue since the message may be being processed by the FSH at the time you issue the "Clear" call. You can check whether the queue is really clear by watching the "Current Depth" value for the queue in the MQ Explorer each time it refreshes its display.

Note: Obviously, the situation where a poison message is continually being rolled back and re-read is not desirable, so be sure to configure your DataPower and/or MQ parameters to specify a backout threshold and a backout requeue name.

End of Exercise 3

4 Exercise 4: Syncpoint between Two Queue Managers

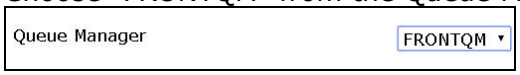
In this exercise, you will connect to a second Queue Manager that will be accessed by the FSH separate from the back-end. This environment will be used to demonstrate how syncpoint works when there are two separate resources being handled. Here's a diagram showing how the Queue Managers will be connected to DataPower:



4.1 Configure DataPower to use "FRONTQM"

Return to the DataPower WebGUI, select "Multi-Protocol Gateway," then click on "MQtest."
Choose the Front Side Handler "DPIN" and click the ellipsis (...) to edit it:

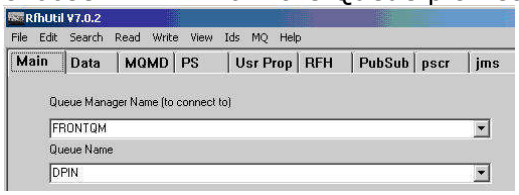
Choose "FRONTQM" from the Queue Manager pick list, then click "Apply":



Click "Apply" on the MPG then "Save Configuration."

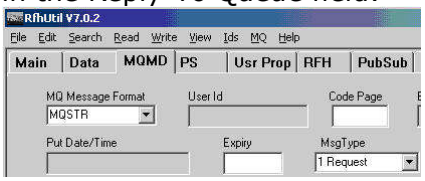
4.2 Send a Test Message

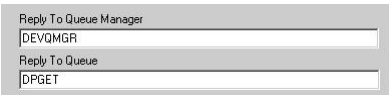
Return to **`rfhutil`**, choose "FRONTQM" from the Queue Manager picklist, then choose "DPIN" from the Queue picklist:



Click "Open file," choose "Ex4-test.xml," and click "Open." The **rfhutil** program will read the file and display the file status as shown here:

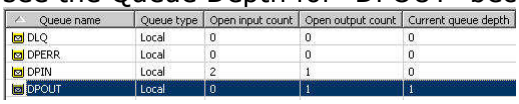
Click on the MQMD tab and choose "MQSTR" as the Message Format, and "Request" as the Message Type.
Since this message will be sent initially to "FRONTQM," the Reply-To Queue Manager name would therefore default to "FRONTQM" but the back-end echo program would not be able to send its reply there since it's running under "DEVQMGR." So you need to set the Reply-To Queue Manager name explicitly to "DEVQMGR" and set "DPGET" in the Reply-To Queue field:





Click the "Main" tab, then click "Write Queue":

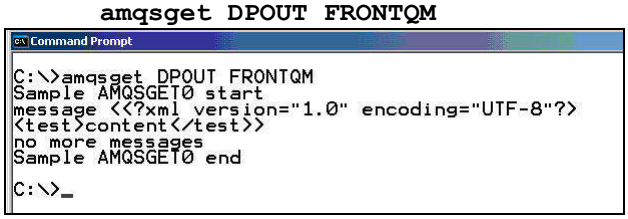
Return to MQ Explorer and click on the "Queue" folder for "FRONTQM." You should see the Queue Depth for "DPOUT" become "1" after a few seconds:



Queue name	Queue type	Open input count	Open output count	Current queue depth
DLQ	Local	0	0	0
DPERR	Local	0	0	0
DPIN	Local	2	1	0
DPOUT	Local	0	1	1

Go to the Command Prompt and issue the following command to retrieve the message:

```
amqsget DPOUT FRONTQM
```



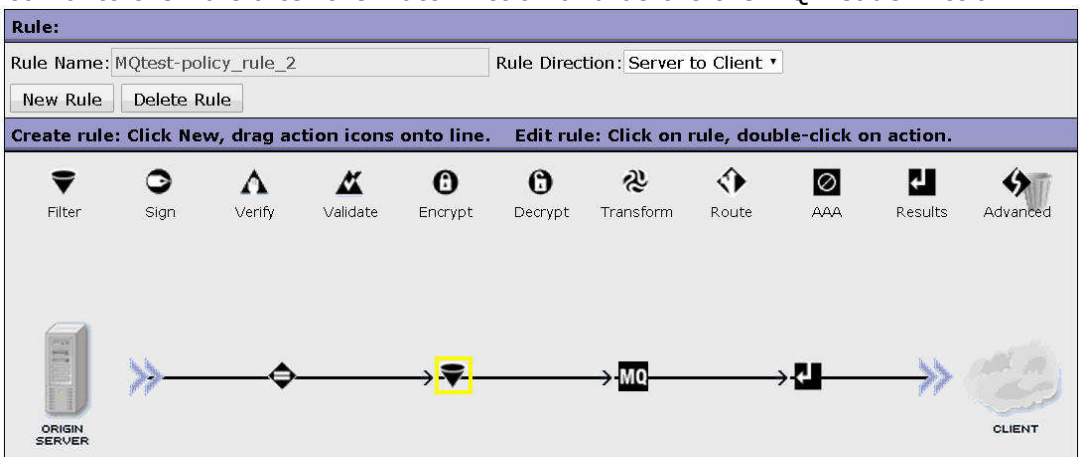
4.3 Test Syncpoint

The preceding part of this exercise was done to show how messages can be propagated between two queue managers via DataPower in a conversational fashion. This next section will demonstrate the effect of syncpoint processing when handling errors.
First, you'll set syncpoint on the back end and see what happens when an error occurs during the Response Rule within the MPG. A Filter Action that has an unconditional "DP:Reject" directive will be used to force an error during Response Rule processing.

Go to the "MQtest" Multi-Protocol Gateway editor on DataPower and alter the Backend URL to read as follows (ie, change "Transactional" to "true"):

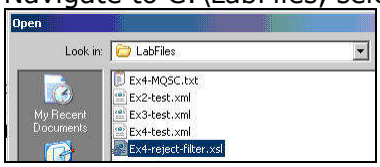
```
dpmq://DEVQMGR/?RequestQueue=DPPUT;ReplyQueue=DPGET;
        Transactional=true;Sync=false
```

Edit the MPG Policy, choose the Response Rule, then click and drag the Filter Action icon onto the Rule after the Match Action and before the MQ Header Action:

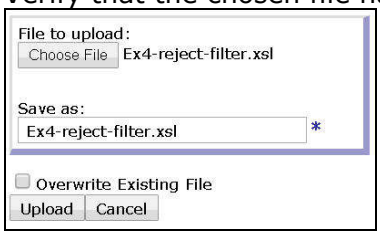


Double-click the Filter Action to edit it, then click the "Upload" button to the right of "Transform File." In the popup window, click "Choose File":

Navigate to C:\LabFiles, select "Ex4-reject-filter.xsl," then click "Open":

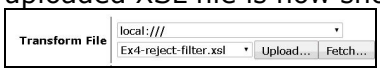


Verify that the chosen file name appears in the popup panel. Click "Upload":



A status panel will pop up. Click "Continue":

You will be returned to the Filter Action editor where you'll see that the newly-uploaded XSL file is now showing beside "Transform File":

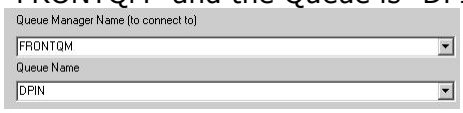


Toward the bottom of the Filter Action editor, choose "Null" from the pick list to the right of "Output." This will cause "NULL" to appear in the Output field to the left of the pick list:

Click "Done" to save the Filter Action, then "Apply Policy," then "Apply" and "Save Configuration" for the Multi-Protocol Gateway.

4.4 Send Message

Prepare "**`rfhutil`**" to send a test message. Check that the Queue Manager is "FRONTQM" and the Queue is "DPIN":

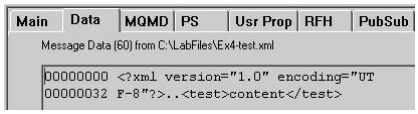


Click on "Open File" and choose:

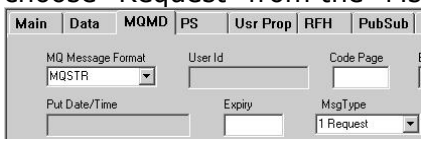
`C:\LabFiles\Ex4-test.xml`

The file name will show in the panel once you've loaded it:

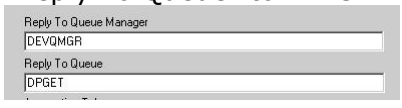
Click the "Data" tab and ensure the data looks like this:



Click the "MQMD" tab, choose "MQSTR" from the "MQ Message Format" pick list, and choose "Request" from the "MsgType" pick list:

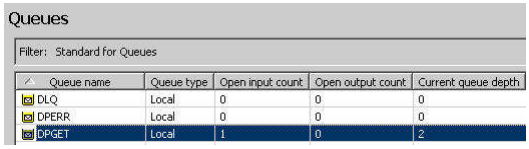


While still on the "MQMD" tab, set the "Reply To Queue Manager" to "DEVQMGR" and "Reply To Queue" to "DPGET":



Click the "Main" tab, then click "Write Queue":

Go to the MQ Explorer, click on the Queues folder beneath "DEVQMGR" wait for about 10 seconds. You should see "DPGET" showing a Current Depth of 2:

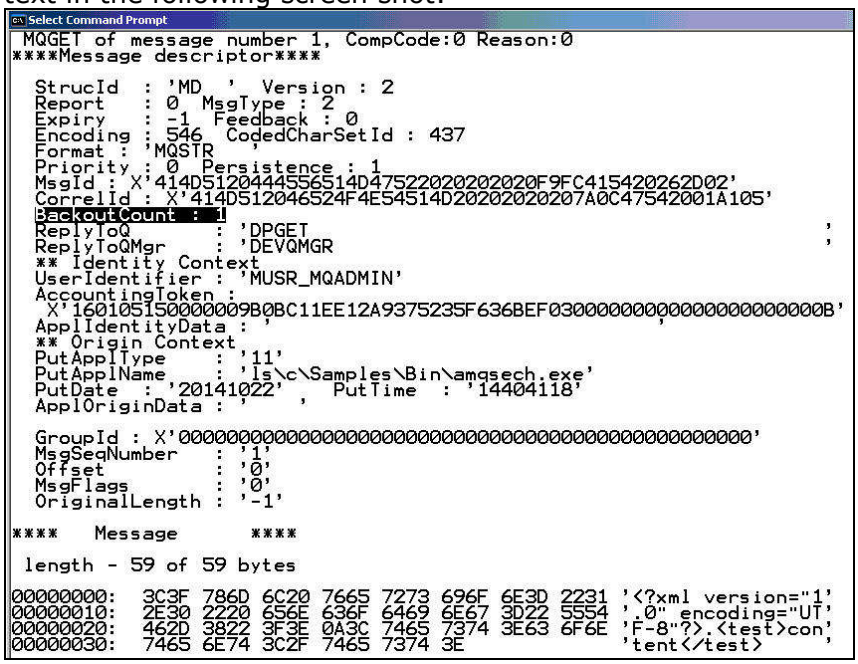


To see what they contain, go to a Command Prompt and run

amqsbcg DPGET DEVQMGR

This will return about two screenloads of information, representing the contents of the two messages on the queue.

The first message is the reply from the echo program that was forcibly rejected by the Filter Action. Since you set "Transactional=true" on the backend URL, the Filter reject causes the transaction to roll back, thereby placing the message back on the queue where it came from. Note the "Backout Count" in the Message Descriptor is set to "1" meaning that it has been read and rolled back once. See the highlighted text in the following screen shot:



The second message is an error notification generated automatically by the Response rule when the Filter Action throws its "Reject." The fault message is as follows:

```
Select Command Prompt

 MQGET of message number 2, CompCode:0 Reason:0
****Message descriptor****

  StrucId  : 'MD  '  Version : 2
  Report   : 0  MsgType : 2
  Expiry   : -1  Feedback : 0
  Encoding : 546  CodedCharSetId : 819
  Format : 'MQSTR   '
  Priority : 0  Persistence : 0
  MsgId : X'414D5120444556514D47522020202020F9FC415420262E02'
  CorrelId : X'414D512046524F4E54514D20202020207A0C47542001A105'
  BackoutCount : 0
  ReplyToQ       : '                                                '
  ReplyToQMgr    : 'DEVQMGR                                         '
  ** Identity Context
  UserIdentifier : 'MUSR_MQADMIN'
  AccountingToken :
   X'160105150000009B0BC11EE12A9375235F636BEF0300000000000000000000000B'
  ApplIdentityData : '
  ** Origin Context
  PutApplType    : '6'
  PutApplName    : 'WebSphere Datapower MQClient'
  PutDate   : '20141022'   PutTime   : '14404119'
  ApplOriginData : '    '

  GroupId : X'000000000000000000000000000000000000000000000000'
  MsgSeqNumber    : '1'
  Offset          : '0'
  MsgFlags        : '0'
  OriginalLength : '-1'

****   Message      ****

 length - 240 of 240 bytes

00000000:  3C3F 786D 6C20 7665 7273 696F 6E3D 2231 '<?xml version="1'
00000010:  2E30 2220 656E 636F 6469 6E67 3D22 5554 '.0" encoding="UT'
00000020:  462D 3822 3F3E 0A3C 656E 763A 456E 7665 'F-8"?>.<env:Enve'
00000030:  6C6F 7065 2078 6D6C 6E73 3A65 6E76 3D22 'lope xmlns:env="'
00000040:  6874 7470 3A2F 2F73 6368 656D 6173 2E78 'http://schemas.x'
00000050:  6D6C 736F 6170 2E6F 7267 2F73 6F61 702F 'mlsoap.org/soap/'
00000060:  656E 7665 6C6F 7065 2F22 3E3C 656E 763A 'envelope/"><env:'
00000070:  426F 6479 3E3C 656E 763A 4661 756C 743E 'Body><env:Fault>'
00000080:  3C66 6175 6C74 636F 6465 3E65 6E76 3A53 '<faultcode>env:S'
00000090:  6572 7665 723C 2F66 6175 6C74 636F 6465 'erver</faultcode'
000000A0:  3E3C 6661 756C 7473 7472 696E 673E 2028 '><faultstring> ('
000000B0:  6672 6F6D 2073 6572 7665 7229 3C2F 6661 'from server)</fa'
000000C0:  756C 7473 7472 696E 673E 3C2F 656E 763A 'ultstring></env:'
000000D0:  4661 756C 743E 3C2F 656E 763A 426F 6479 'Fault></env:Body'
000000E0:  3E3C 2F65 6E76 3A45 6E76 656C 6F70 653E '></env:Envelope>'
```

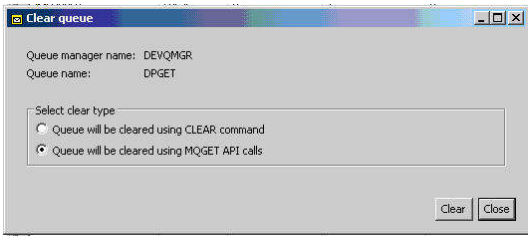
Some things to note:

- The body of the message is the standard DataPower "<env:Fault>"
- The message is sent to the same queue the bad message came from
- The MessageID and CorrelationID are identical to the original message
- The PutApplName is set to "WebSphere Datapower MQClient"
- The PutApplType is "6" (Unix), even though DataPower is not really Unix
- The AccountingToken is identical to the original message

In a production environment, it would obviously be preferable to code an "On Error" action and an Error Rule to handle this in a more sophisticated fashion.

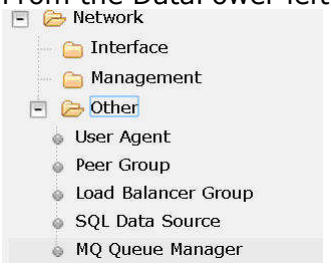
Clear the messages off "DPGET" in preparation for the next part of this exercise. Right click on "DPGET" in "DEVQMGR" and select "Clear messages":

Check the radio button "Queue will be cleared using MQGET API calls" then click the "Clear" button:

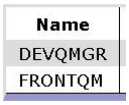


4.5 Enable Front Side Syncpoint

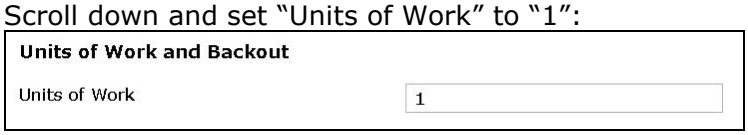
Next you will enable syncpoint control on the FSH and send the bad message once more.
From the DataPower left side menu, select Network -> Other -> MQ Queue Manager:



Click on "FRONTQM":

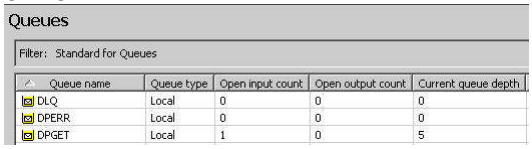


Scroll down and set "Units of Work" to "1":



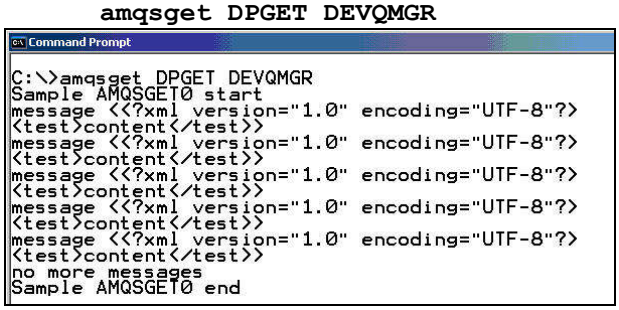
Leave all other fields as is. Be sure "Automatic Backout" is "off."
Click "Apply" and "Save Configuration."

Return to **`rfhutil`** and click "Write Q." Return to MQ Explorer and select the "Queues" folder under "DEVQMGR." You will see that "DPGET" now has 5 messages on it:



Display these messages by using the following command:

```
amqsget DPGET DEVQMGR
```



Notice that the "Fault" message you saw earlier is not present. The reason for this is that the FSH is now operating under its own syncpoint, and when the Response Rule throws an error, the MPG Policy acts upon it by issuing a rollback of the frontside transaction. It considers that to be sufficient remedial action and does not create a fault message.

Here's the sequence of events that occurs in this situation:

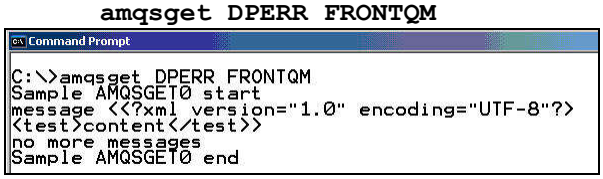
1. The FSH retrieves the "BOTHRESH" value of "5" from "DPIN"
2. The FSH retrieves the "BOQUEUE" value of "DPERR" from "DPIN"
3. The FSH gets the message and passes it through the Request Rule
4. The echo program puts the message onto "DPGET" on "DEVQMGR"
5. The Response Rule gets the message from "DPGET" and throws an error
6. An MQBACK is issued to "DEVQMGR"
7. "DEVQMGR" puts the response message back on "DPGET"
8. An "MQBACK" is also issued to "FRONTQM"
9. "FRONTQM" puts the original message back on "DPIN"
10. "FRONTQM" increments the "BackoutCount" on the message by 1
11. Steps 3 through 8 are repeated until the "BackoutCount" exceeds 5
12. The message is sent to "DPERR" on "FRONTQM" and an "MQCMIT" is issued

This behavior has important architectural implications, since the original message is being sent 5 times in this case, and if the backend program were doing some database update, it would be doing it 5 times. If this were an incremental transaction of some kind, such as a bank account withdrawal, the results would be unacceptable.

4.6 Clean up

Remove the message from "DPERR" on "FRONTQM" with the following command:

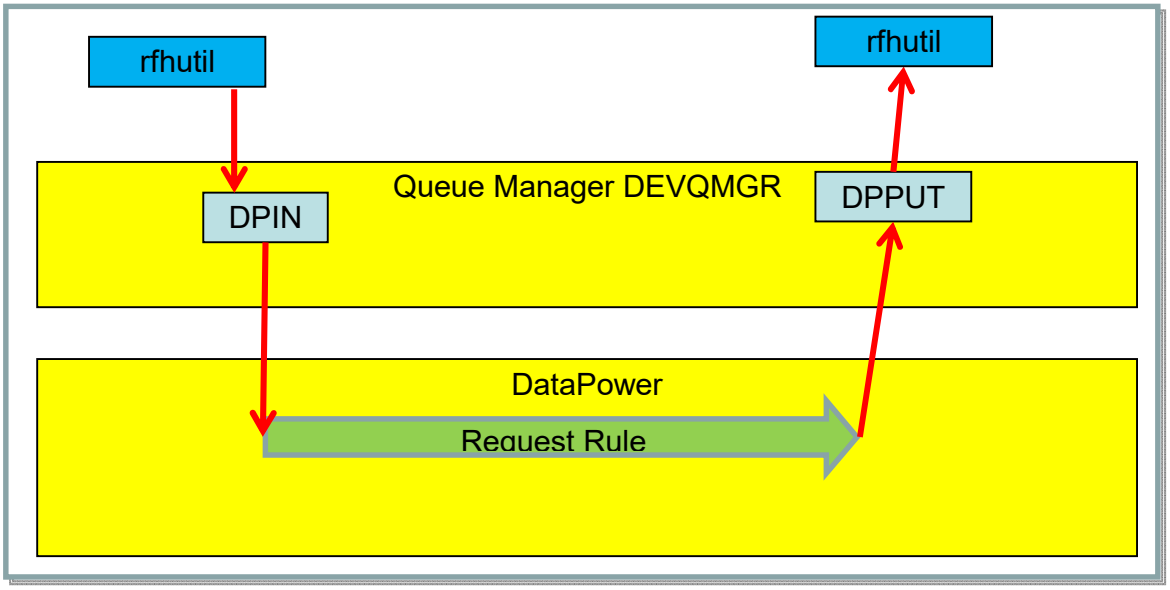
```
amqsget DPERR FRONTQM
```



End of Exercise 4

5 Exercise 5: MQ Message Manipulation

In this exercise, you will create a Request Rule in your Multi-Protocol Gateway that will directly access fields in the Message Descriptor. The environment will look like this:

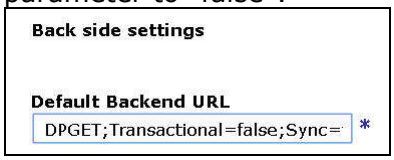


First, you will need to reset the infrastructure to have a single Queue Manager, and disable the back end triggering of the echo program.

5.1 Single Queue Manager Setup

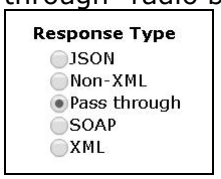
Edit the DataPower Multi-Protocol Gateway Front Side Handler, selecting "DEVQMGR" as the Queue Manager from the pick list. Then click "Apply":

Scroll down to the Backend URL in the MPG editor and change the "Transactional" parameter to "false":



5.2 Set MPG for One-Way Messaging

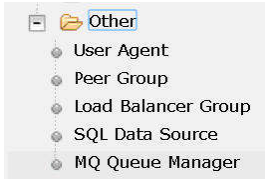
For this exercise, we only use the Request Rule and don't need to process a Response Rule. Scroll down the left side of the MPG editor and click the "Pass through" radio button under "Response Type":



Click "Apply" then "Save Configuration."

5.3 Disable Syncpoint on the Queue Manager Object

From the DataPower menu, choose Network -> Other -> MQ Queue Manager:



Click on "DEVQMGR" from the list of Queue Managers:

Scroll down and set "Units of Work" to "0":

Click "Apply" then "Save Configuration."

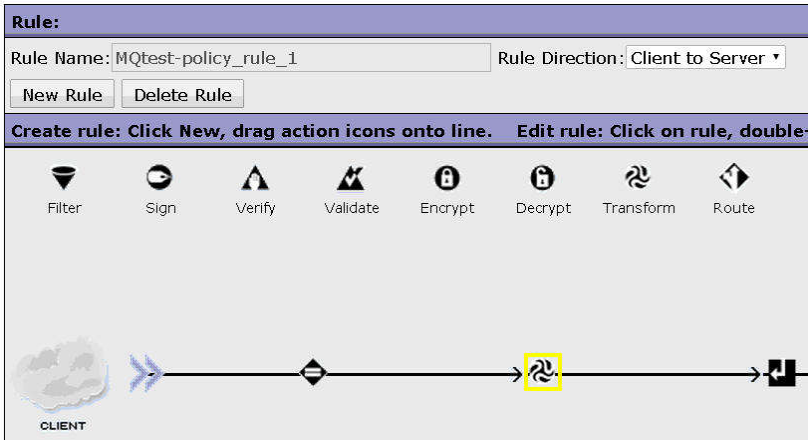
5.4 Disable Triggering on the DPPUT Queue

Use MQ Explorer to edit the properties on the DPPUT queue. Click on the Triggering tab, and select "Off" in the Trigger Control pick list, then click "Ok" to save the changes:

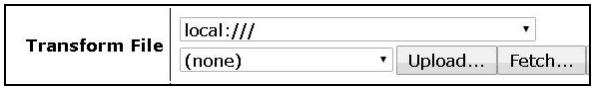
5.5 Read MQMD Fields

You will now use a Transform Action to run a Style Sheet that extracts certain MQMD fields and creates a Log entry containing those fields.

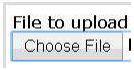
Edit the MPG Policy and select the Request Rule.
Click and drag the Transform Action icon onto the Rule line between the Match Action and the Results Action:



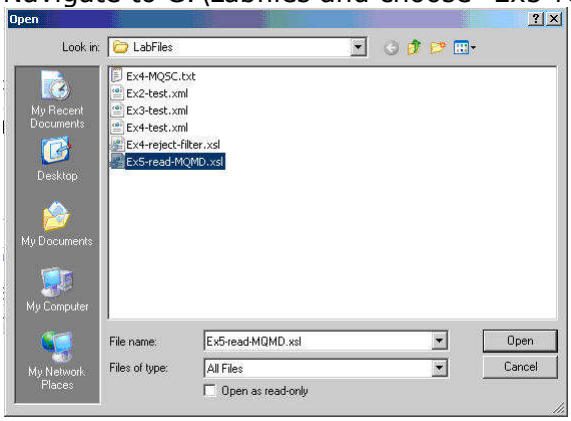
Double-click the Transform Action to edit it. Click "Upload" to the right of "Transform File":



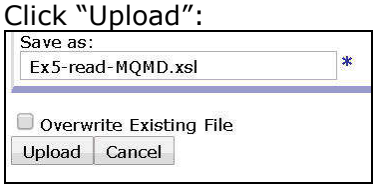
Click "Choose File":



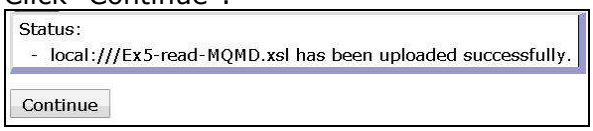
Navigate to C:\Labfiles and choose "Ex5-read-MQMD.xsl" then click "Open":



Click "Upload":



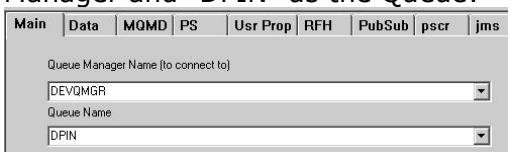
Click "Continue":



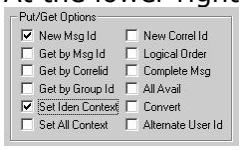
Click "Done" to close the Transform Action editor.
"Apply" the Policy, close the window, "Apply" the MPG, and "Save Configuration."

5.6 Send a Message

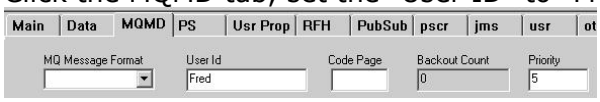
Run "**`rfhutil`**," click on the "Main" tab, then select "DEVQMGR" as the Queue Manager and "DPIN" as the Queue:



At the lower right corner of the "Main" tab, check the "Set Iden Context" check box:



Click the MQMD tab, set the "User ID" to "Fred" and the "Priority" to "5":



Click the "Main" tab and click "Write Q":

In the DataPower MPG editor screen, click the "View Log" link as shown at lower right of the following screen shot:

Near the top of the Log display, you will see the extracted information from the MQMD displayed as a log entry:

5.7 Stylesheet Code

Here's a code snippet from the "Ex5-read-MQMD.xsl" stylesheet that obtained the fields from the MQMD. The embedded comments explain what each line does:

```
<xsl:template match="/">
<xsl:message dp:priority="'warn'">Start Read MQMD.</xsl:message>
<!-- get the MQMD header from the request message -->
<xsl:variable name="entries" select="dp:request-header('MQMD')"/>
<!-- parse into a usable nodeset -->
<xsl:variable name="header" select="dp:parse($entries)"/>
<!-- store the desired values in a variable available for later -->
<xsl:variable name="Priority" select="$header//Priority"/>
<xsl:variable name="UserID" select="$header//UserIdentifier"/>
<!-- Generate a log entry containing the captured fields -->
<xsl:message dp:priority="notice">
<xsl:value-of select="concat ('Priority = ',$Priority,' User ID =
',$UserID)"/>
</xsl:message>
</xsl:template>
```

5.8 Summary

This exercise showed how to access any of the fields in the MQMD using XSLT code within a processing rule. The extracted data was displayed using the DataPower Log facility but could have just as easily been used for any other purpose.

End of Exercise 5

6 Appendix: MQ Constants

6.1 Message Types

MQMT_REQUEST	1
MQMT_REPLY	2
MQMT_DATAGRAM	8
MQMT_REPORT	4

6.2 Put Message Options

MQPMO_SYNCPOINT	2
MQPMO_NO_SYNCPOINT	4
MQPMO_DEFAULT_CONTEXT	32
MQPMO_NEW_MSG_ID	64
MQPMO_NEW_CORREL_ID	128
MQPMO_PASS_IDENTITY_CONTEXT	256
MQPMO_PASS_ALL_CONTEXT	512
MQPMO_SET_IDENTITY_CONTEXT	1024
MQPMO_SET_ALL_CONTEXT	2048
MQPMO_ALTERNATE_USER_AUTHORITY	4096
MQPMO_FAIL_IF_QUIESCING	8192
MQPMO_NO_CONTEXT	16384
MQPMO_LOGICAL_ORDER	32768
MQPMO_ASYNC_RESPONSE	65536
MQPMO_SYNC_RESPONSE	131072
MQPMO_RESOLVE_LOCAL_Q	262144
MQPMO_RETAIN	2097152
MQPMO_MD_FOR_OUTPUT_ONLY	8388608
MQPMO_SCOPE_QMGR	67108864
MQPMO_SUPPRESS_REPLYTO	134217728
MQPMO_NOT_OWN_SUBS	268435456
MQPMO_RESPONSE_AS_Q_DEF	0
MQPMO_RESPONSE_AS_TOPIC_DEF	0
MQPMO_NONE	0

6.3 Get Message Options

MQGMO_WAIT	1
MQGMO_NO_WAIT	0
MQGMO_SET_SIGNAL	8
MQGMO_FAIL_IF_QUIESCING	8192
MQGMO_SYNCPOINT	2
MQGMO_SYNCPOINT_IF_PERSISTENT	4096
MQGMO_NO_SYNCPOINT	4
MQGMO_MARK_SKIP_BACKOUT	128
MQGMO_BROWSE_FIRST	16
MQGMO_BROWSE_NEXT	32
MQGMO_BROWSE_MSG_UNDER_CURSOR	2048
MQGMO_BROWSE_HANDLE	17825808
MQGMO_BROWSE_CO_OP	18874384
MQGMO_MSG_UNDER_CURSOR	256
MQGMO_LOCK	512
MQGMO_UNLOCK	1024
MQGMO_ACCEPT_TRUNCATED_MSG	64
MQGMO_CONVERT	16384
MQGMO_LOGICAL_ORDER	32768
MQGMO_COMPLETE_MSG	65536
MQGMO_ALL_MSGS_AVAILABLE	131072
MQGMO_ALL_SEGMENTS_AVAILABLE	262144
MQGMO_MARK_BROWSE_HANDLE	1048576
MQGMO_MARK_BROWSE_CO_OP	2097152
MQGMO_UNMARK_BROWSE_CO_OP	4194304
MQGMO_UNMARK_BROWSE_HANDLE	8388608
MQGMO_UNMARKED_BROWSE_MSG	16777216
MQGMO_PROPERTIES_FORCE_MQRFH2	33554432
MQGMO_NO_PROPERTIES	67108864
MQGMO_PROPERTIES_IN_HANDLE	134217728
MQGMO_PROPERTIES_COMPATIBILITY	268435456
MQGMO_PROPERTIES_AS_Q_DEF	0

7 Appendix: Triggering Setup