Impact2011
Changing the Way Business and IT Leaders Work

*IBM Software Accel*erated Value Program

AVP-2930

How WebSphere MQ SupportPacs can help you with administering your WebSphere MQ Environment

http://www-01.ibm.com/software/support/acceleratedvalue/

WebSphere MQ SupportPacs Lab

What this lab is about

WebSphere MQ SupportPacs can help with the administration and monitoring of your WMQ environment. This lab will give an overview of the different categories of SupportPacs and how you can make use of them in your organization. You will be able to test out selected SupportPacs designed to assist with the Administration and Monitoring of WebSphere MQ to assess their suitability for your environment.

This lab is provided **AS-IS**, with no formal IBM support.

Lab requirements

List of system and software required for the attendee to complete the lab.

- WebSphere MQ V7.0.1.3

What you should be able to do

At the end of this lab you should be able to:

- Appreciate the benefits of some of the WMQ SupportPacs available that can be used for monitoring, moving messages and troubleshooting
- Use the SupportPacs to perform common administration tasks.

Lab notes

- All actions that need to be completed in the lab are highlighted with a grey background.
- Each part of this Lab can be completed **independently**:
  - Part 2: Monitoring MQ
  - Part 3: Moving Messages between queues
  - Part 4: Troubleshooting
- Runtimes of each part might vary depending on your individual interest and skills. The approximate runtime for each part is 20 – 30 minutes.

  Hint:

  - Chose up to 2 parts to get deeper insight into the specific SupportPacs
  - Chose 1 part to perform an even “deeper dive” into the specific SupportPac

Part 1: Lab Set Up

- The VM Image “AVP-L03_MQ_SupportPac_lab” required for you to complete the lab should already be up and running for you. If it isn’t please contact any of the AVP representatives in the room.

- Login to the VMWare image with the following username/password :

  Username :**Administrator**

  Password : **Happy2Be**

Part 2: Monitoring WMQ

There are a number of SupportPacs that have been developed to assist with the monitoring of WMQ. This lab will demonstrate some of the functionality provided by the following SupportPacs:

- **MH04 Queue Statistics Tool**

  http://www-01.ibm.com/support/docview.wss?rs=171&q1=Xa02&uid=swg24025857

- **MP08 WMQ Performance Monitor**

  http://www-01.ibm.com/support/docview.wss?rs=171&uid=swg24025448

- **MO01 Event Monitor for Websphere MQ**

  https://www-304.ibm.com/support/docview.wss?uid=swg24000676

The following table gives a comparison of the features and potential uses for the SupportPacs:

	MH04	MP08	MO01
Type of Monitoring	Real-time queue monitoring	Accounting and Statistics	WMQ Events
GUI or Command Line	Command Line	GUI	Command Line
Purpose	Enables you to determine the current state of queues within a queue manager.	Enables you to record information about MQI operations performed by WMQ applications, or activities occurring in a WMQ system.	Enables you to detect occurrences of instrumentation events in a queue manager or channel instance.
Possible uses	Troubleshooting application issues. Determining the messages throughput on a queue. Monitor and record the activity on the queue.	Account for application resource use & record application activity. Capacity planning. Detect problems in your queue manager network. Assist in determining the causes of problems in your queue manager network. Improve the efficiency of your queue manager network. Confirm that your queue manager network is running correctly.	Detect problems in your queue manager network Assist in determining the cause of problems in your queue manager network Generate an audit trail React to queue manager state changes
SupportPac Category	Category 2	Category 2	Category 4

2.1 MH04 Queue Statistics Tool

The MH04 SupportPac supplies two command line tools:

1. **xmqqstat :** displays real time statistics about a single WMQ local queue.

This tool is particularly useful for troubleshooting application issues.

2. **xmqqstab :** displays statistics for a list of WebSphere MQ local queues.

This tool is particularly useful for monitoring the use of queues overtime and to prevent queue full situations.

The tools provided in the SupportPac use the WMQ Java classes for PCF. They are supplied with WMQ V7, but if you are at v6 you will also need to install the SupportPac MS0B – WebSphere MQ Java Classes for PCF. You can find the MS0B SupportPac at

http://www-01.ibm.com/support/docview.wss?uid=swg24000668.

This lab will show you the type of output you can generate using the tools to enable you to evaluate whether they would be useful tools to implement in your enterprise. Both of the tools are supported in client or bindings mode. This lab will just be using bindings mode.

For the purpose of this lab, the following queue manager has been created:

MonitoringQM_1

The following objects have been created on the queue manager:

Local Queues	Alias Queues	Topics	Subscriptions

MONITORQ_1
MonQ(QMGR)*
	MONITORQ_TOPIC
	TargType(TOPIC)
	Target(MONITORQ_TOPIC)
		MONITORQ_TOPIC
		TopicStr(/MonitorQ)
			MONITORQ_1_SUB
			TopicStr(/MonitorQ)
			Dest(MONITORQ_1)
			DestQmgr(MonitoringQM_1)

MONITORQ_2
MonQ(Qmgr)*
			MONITORQ_2_SUB
			TopicStr(/MonitorQ)
			Dest(MONITORQ_2)
			DestQmgr(MonitoringQM_1)

MONITORQ_3
MonQ(Qmgr)
			MONITORQ_3_SUB
			TopicStr(/MonitorQ)
			Dest(MONITORQ_3)
			DestQmgr(MonitoringQM_1)

MONITORQ_4
MonQ(Qmgr)*
			MONITORQ_4_SUB
			TopicStr(/MonitorQ)
			Dest(MONITORQ_4)
			DestQmgr(MonitoringQM_1)

*The MonQ parameter on the queue manager has been set to HIGH. This means that queue monitoring has been turned on to the highest level. This enables us to gather all the available statistics with the tools in the SupportPac.

2.1.1 Using the xmqqstat command to display usage information about a single queue.

The xmqqstat utility displays real time statistics information about a single WebSphere MQ queue. This tool would be useful for the following scenarios:

- Troubleshooting application issues. The tool displays usage information for the queue such as the message enqueue and dequeue count, the number of uncommitted messages and the open input/output counts to help you determine how your applications are using the queue and if they are behaving as expected.

- Determining the messages throughput on a queue. This could be useful in a test environment for measuring the workload on a queue using the enqueue and dequeue counts and the OQTS and OQTL counts which give the length of time a message remain on a queue.

- Monitoring and recording the activity on the queue. The tool enables you to monitor the queues usage over time and enables you to identify if you need to increase capacity or move workload to another queue. The PQF value gives the depth of the queue as a percentage of the maximum allowed depth.

Before starting this lab ensure you have a command prompt open from the shortcuts provided on the desktop on the VMWare image.

- In the command window change directory to C:\IMPACT2011\MonitoringLab\mh04 lab

```
> cd c:\IMPACT2011\MonitoringLab\mho4 lab
```

- Issue the command “xmqqstat” without any parameters. This will list details of all the input and output options supplied with the tool.

```
> xmqqstat
```

To demonstrate the information you can receive from the xmqqstat tool there are a couple of windows command files located in the current directory which simulate putting a message load onto the queue (using the mqput2 command from the SupportPac IH03) and retrieving the messages from the queue (which uses the mqcapture command from the SupportPac IH03).

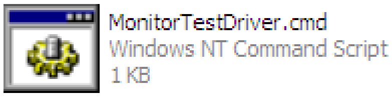
- Open the directory C:\IMPACT2011\MonitoringLab\mh04 in Windows Explorer. There is a shortcut on the desktop “mh04 lab” which will take you to this directory.

You will be using the following *.cmd files to put and get messages to MONITORQ_1 on the queue manager MonitoringQM_1:

MonitorTestDriver.cmd – puts 10000 messages onto MONITORQ_1. Once the queue has a current depth of 150 messages the application stops putting messages until the current depth falls below 150.

GetMessages.cmd – pulls 150 messages from MONITORQ_1 at a time until the queue has a curdepth of 0

- Run the MonitorTestDriver.cmd file by double clicking on it in the Windows Explorer



MonitorTestDriver.cmd
Windows NT Command Script
1 KB

- A new command prompt will open for the MonitorTestDriver, you can just minimize this window.

- Start the xmqqstat tool from the other command prompt youand direct it to write output to a new file named mh04.output (note the command is case sensitive).

```
> xmqqstat –m MonitoringQM_1 –q MONITORQ_1 –d 20 –i 1 –t –e –f mh04.output
```

Where:

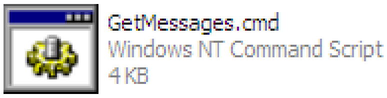
-d specifies the duration the command should run for in seconds

-i specifies the interval at which to display statistics in seconds

-e displays extended data

-f specifies the name of the file to write output to.

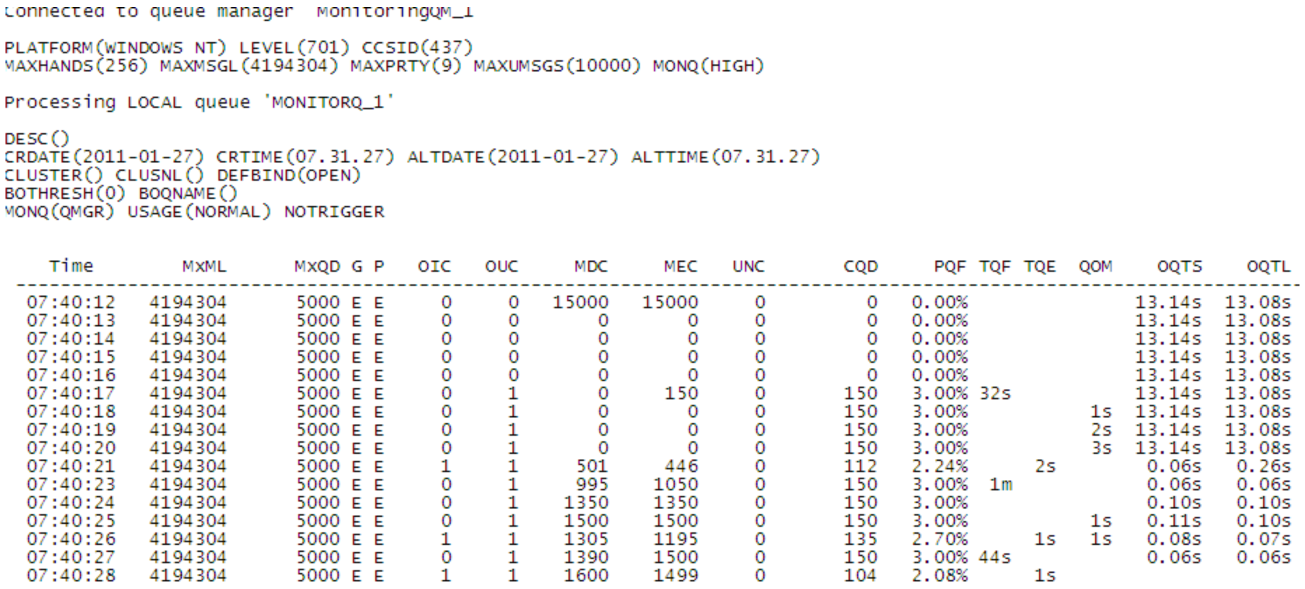
- Now run the GetMessage.cmd file by double clicking on the files in the Windows Explorer:



GetMessages.cmd
Windows NT Command Script
4 KB

- When the xmqqstat command has completed open up the newly created mh04.output file which will be in the current directory C:\Impact2011\Monitoringlab\mh04 lab

You will see similar output to this (but not the same because the MonitorTestDriver and GetMessages commands will have been run at different intervals):



```
Connected to queue manager  MonitoringQM_1

PLATFORM(WINDOWS NT) LEVEL(701) CCSID(437)
MAXHANDS(256) MAXMSGL(4194304) MAXPRTY(9) MAXUMSGS(10000) MONQ(HIGH)

Processing LOCAL queue 'MONITORQ_1'

DESC()
CRDATE(2011-01-27) CRTIME(07.31.27) ALTDATE(2011-01-27) ALTTIME(07.31.27)
CLUSTER() CLUSNL() DEFBIND(OPEN)
BOTHRESH(0) BOQNAME()
MONQ(QMGR) USAGE(NORMAL) NOTRIGGER


   Time         MxML      MxQD G P   OIC    OUC      MDC      MEC    UNC        CQD      PQF TQF TQE   QOM    OQTS     OQTL
  ---------------------------------------------------------------------------------------------------------------------------
  07:40:12   4194304      5000 E E     0      0    15000    15000      0          0    0.00%                13.14s   13.08s
  07:40:13   4194304      5000 E E     0      0        0        0      0          0    0.00%                13.14s   13.08s
  07:40:14   4194304      5000 E E     0      0        0        0      0          0    0.00%                13.14s   13.08s
  07:40:15   4194304      5000 E E     0      0        0        0      0          0    0.00%                13.14s   13.08s
  07:40:16   4194304      5000 E E     0      0        0        0      0          0    0.00%                13.14s   13.08s
  07:40:17   4194304      5000 E E     0      1        0      150      0        150    3.00% 32s            13.14s   13.08s
  07:40:18   4194304      5000 E E     0      1        0        0      0        150    3.00%            1s  13.14s   13.08s
  07:40:19   4194304      5000 E E     0      1        0        0      0        150    3.00%            2s  13.14s   13.08s
  07:40:20   4194304      5000 E E     0      1        0        0      0        150    3.00%            3s  13.14s   13.08s
  07:40:21   4194304      5000 E E     1      1      501      446      0        112    2.24%      2s          0.06s    0.26s
  07:40:23   4194304      5000 E E     0      1      995     1050      0        150    3.00%  1m               0.06s    0.06s
  07:40:24   4194304      5000 E E     0      1     1350     1350      0        150    3.00%                   0.10s    0.10s
  07:40:25   4194304      5000 E E     0      1     1500     1500      0        150    3.00%            1s   0.11s    0.10s
  07:40:26   4194304      5000 E E     1      1     1305     1195      0        135    2.70%      1s    1s   0.08s    0.07s
  07:40:27   4194304      5000 E E     0      1     1390     1500      0        150    3.00% 44s             0.06s    0.06s
  07:40:28   4194304      5000 E E     1      1     1600     1499      0        104    2.08%      1s
```

The following information in the output of the xmqqstat is of particular interest for this test:

1. The CQD (Current Queue Depth) never rises above 150, which confirms the application is behaving as expected
2. The PQF (Percentage Queue Full) never rises above 3% during the time period which would suggest there aren't any capacity issues.
3. The OIC (Open Input Count) and OUC (Open Output Count) are never more than 1, which confirms that only the applications we are expecting are accessing the queue to put or get messages.

4. The QOM (Queue oldest message age) is 3 seconds at it's highest point. This value could be evaluated against any service level agreements for the application.

- Close the file "mh04.output"

2.1.2 Using the xmqqstab command to display usage information multiple queues.

The xmqqstab command displays usage information for a supplied list of WMQ queues. This tool is more suited for monitoring the usage of queues over time than for troubleshooting specific issues with a queue. Examples of tasks the xmqqstab utility would be useful for would be:

- Checking that workload is being correctly distributed between a number of queues if workload balancing is being used.
- Ensuring that queues don't exceed their capacity. Provides early warning of queues reaching their maxdepth.

You should still have a command prompt open in the directory c:\IMPACT2011\MonigoringLab\mh04 lab. If you don't, open a new command prompt and change to this directory.

- In the command prompt window Issue the command "xmqqstab" without any parameters. This will list details of all the input and output options supplied with the tool.

```
> xmqqstab
```

You will be using the following *.cmd file (also located in C:\Impact2011\MonitoringLab\mh04 lab) to put and get messages onto multiple queues to simulate workload to demonstrate the information you can receive from the xmqqstab tool:

MonitorTestDriver_PutToMany.cmd – puts 5000 messages onto MONITORQ_TOPIC using the amqsblst utility (supplied with WMQ V7). MONITORQ_TOPIC is an aliasq that points to a topic object also called MONITORQ_TOPIC. There are 4 subscribers to this topic, each using the following subscription queues to receive publications:

MONITORQ_1,

MONITORQ_2

MONITORQ_3

MONITORQ_4

5000 messages should be placed on each of these queues and then taken off the queues until the current depth returns to 0

If you do not already have the directory C:\IMPACT2011\MonitoringLab\mh04 open in Windows Explorer from the last exercise, open it now. There is a shortcut on the desktop "mh04 lab" which will take you to this directory.

- Run the MonitorTestDriver_PutToMany command file by double clicking on the file in the Windows Explorer window:

- In the C:\IMPACT2011\MonitoringLab\mh04 lab command window, start the xmqqstab tool and direct it to write output to a new file named xmqqstab.output. Once the tool has been started you can then run the MonitorTestDriver_PutToMany.cmd file by double clicking on the file in the Windows Explorer window you have open.

```
> xmqqstab -m MonitoringQM_1 -q MONITORQ* -r > xmqqstab.output
```

Where:

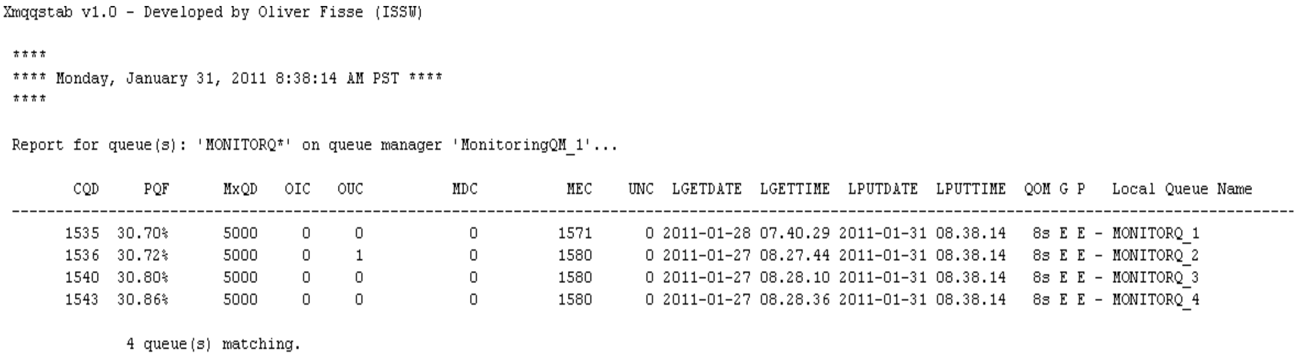
-m specifies the queue manager name

-q specifies the queues to run the tool against (in this case it will be run against all queues starting with the prefix MONITORQ)

-r displays and resets dequeue and enqueue counts

- When the xmqqstab command has completed open up the newly created xmqstab.output file which will be in the current directory C:\Impact2011\Monitoringlab\mh04 lab

You will see similar output to this (but not the same)



```
Xmqqstab v1.0 - Developed by Oliver Fisse (ISSW)

 ****
 **** Monday, January 31, 2011 8:38:14 AM PST ****
 ****

 Report for queue(s): 'MONITORQ*' on queue manager 'MonitoringQM_1'...

        CQD     PQF      MxQD   OIC   OUC         MDC         MEC    UNC  LGETDATE  LGETTIME  LPUTDATE  LPUTTIME  QOM G P   Local Queue Name
 -----------------------------------------------------------------------------------------------------------------------------------------------
       1535  30.70%      5000     0     0           0        1571      0 2011-01-28 07.40.29 2011-01-31 08.38.14   8s E E - MONITORQ_1
       1536  30.72%      5000     0     1           0        1580      0 2011-01-27 08.27.44 2011-01-31 08.38.14   8s E E - MONITORQ_2
       1540  30.80%      5000     0     0           0        1580      0 2011-01-27 08.28.10 2011-01-31 08.38.14   8s E E - MONITORQ_3
       1543  30.86%      5000     0     0           0        1580      0 2011-01-27 08.28.36 2011-01-31 08.38.14   8s E E - MONITORQ_4

             4 queue(s) matching.
```

This output tells us the following:

- At the time the tool was run all the queues were approximately 30% full. The Percentage Queue Full (PQF) field could be monitored to ensure it doesn’t exceed a threshold.

- At the time the tool was run all off the queue have a positive message enqueue count but no message dequeue count. This could indicate an issue with the application consuming the messages, but for this test it is expected as the messages are all put to the queues before they are then consumed.

- The queues MONITORQ_2, MONITORQ_3 and MONITORQ_4 had all retrieved 1580 messages but MONITORQ_1 had only retrieved 1571 messages. This is not unexpected in our test as messages were still being delivered to the queues but if you were expecting an even distribution of messages across the queues you could use the Message Enqueue Count (MEC) to check the distribution over time.

- Close the file “xmqqstab.output” and the Windows Explorer Window.

2.2 MP08 Performance Monitor for WebSphere MQ

The performance monitor for WebSphere MQ is supplied as a plug-in to the WebSphere MQ Explorer. It enables you to view monitoring data for WebSphere MQ objects including MQI statistics, Queue statistics and Channel statistics. You need to install the Performance Monitor Agent on the machine you want to gather statistics information from, currently the agent is only available for Windows.

The Performance Monitor reads accounting and statistics message generated by the queue manager on the SYSTEM.ADMIN.ACCOUNTING.QUEUE and the SYSTEM.ADMIN.STATISTICS.QUEUE and displays it in the MQ Explorer. This information can also be recorded to file.

Accounting messages are used to record information about the MQI operations performed by WebSphere MQ Applications. They are useful for the following purposes:

- Accounting for application resource use.
- Recording application activity.

Statistics messages are used to record information about the activities occurring in a WebSphere MQ system. They are most useful for the following purposes:

- Capacity planning.
- Detecting problems in your queue manager network.
- Assisting in determining the causes of problems in your queue manager network.
- Improving the efficiency of your queue manager network.
- Familiarizing yourself with the running of your queue manager network.
- Confirming that your queue manager network is running correctly.

The SupportPac provides a monitoring Agent, which needs to be installed on the machine hosting the queue manager you want to monitor, and the Performance Monitor itself which needs to be installed on a machine with a WMQ Explorer installation. For the purpose of the lab both the Agent and the WMQ Explorer Plug-in have been pre-installed and configured for use.

2.2.1 Capturing MQI Statistics using the Performance Monitor for WebSphere MQ

For the purpose of this lab, the following queue managers have been created:

MonitoringQM_2

MonitoringQM_3

When you install the Monitoring Agent supplied with the SupportPac you are given the option to set the following options on the queue manager. The options are required for the Performance Monitor to function correctly and have been set on the queue managers created for this lab:

statq(on) statmqi(on) statchl(high) statint(300) acctq(on) acctmqi(on) acctint(300)

where:

statq(on) =queue statistics information is collected for every connection to the queue manager that opens the queue

statmqi(on) =MQI statistics information is collected for every connection to the queue manager

statchl(high) =statistics data collection for this channel is enabled with a high ration of data collection

statint(300) =statistics messages are generated every 300 seconds (the default is 1800 seconds)

acctq(on) =queue accounting information is collected for every connection to the queue manager that opens the queue

acctmqi(on)=MQI accounting information is collected for every connection to the queue manager

acctint(300) =accounting messages are generated every 300 seconds (the default is 1800 seconds)

For full information about the data collected in accounting and statistics message please refer to the Infocenter:

http://publib.boulder.ibm.com/infocenter/wmqV7/v7r0/topic/com.ibm.mq.csqzax.doc/mo13510_.htm

The following objects have been created on the queue managers MonitoringQM_2 and MonitoringQM_3:

MonitoringQM_2

RemoteQ	XmitQ	Channel

MONITORINGQM_3.LOCALQ
RName(LOCALQ)
RQMName(MonitoringQM_3)
Xmitq(MONQM3)
	MONQM3
	TrigData(MONQM_2.TO.MONQM_3)
		MONQM_2.TO.MONQM_3
		Chltype(SDR) XmitQ(MONQM3)

MonitoringQM_3

Local Queues	Alias Queues	Topics	Subscriptions	Channel

MONITORQ_1
MonQ(QMGR)*
	MONITORQ_TOPIC
	TargType(TOPIC)
	Target(MONITORQ_TOPIC
	)
		MONITORQ_TOPIC
		TopicStr(/MonitorQ)
			MONITORQ_1_SUB
			TopicStr(/MonitorQ)
			Dest(MONITORQ_1)
			DestQmgr(MonitoringQM_1)
				MONQM_2.TO.MONQM_3
				Chltype(RCVR)

MONITORQ_2
MonQ(Qmgr)*
			MONITORQ_2_SUB
			TopicStr(/MonitorQ)
			Dest(MONITORQ_2)
			DestQmgr(MonitoringQM_1)	

MONITORQ_3
MonQ(Qmgr)
			MONITORQ_3_SUB
			TopicStr(/MonitorQ)
			Dest(MONITORQ_3)
			DestQmgr(MonitoringQM_1)	

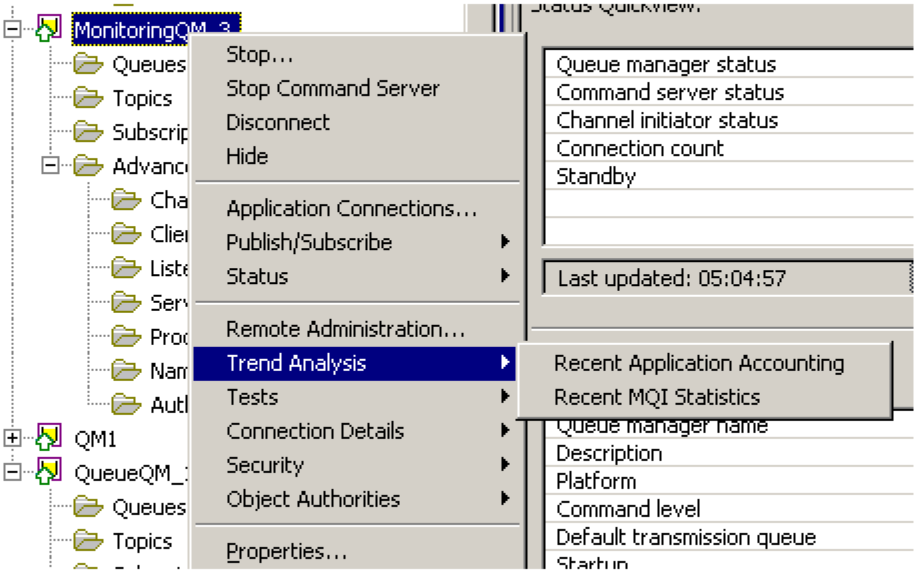
MONITORQ_4
MonQ(Qmgr)*
			MONITORQ_4_SUB
			TopicStr(/MonitorQ)
			Dest(MONITORQ_4)
			DestQmgr(MonitoringQM_1)	

LOCALQ				

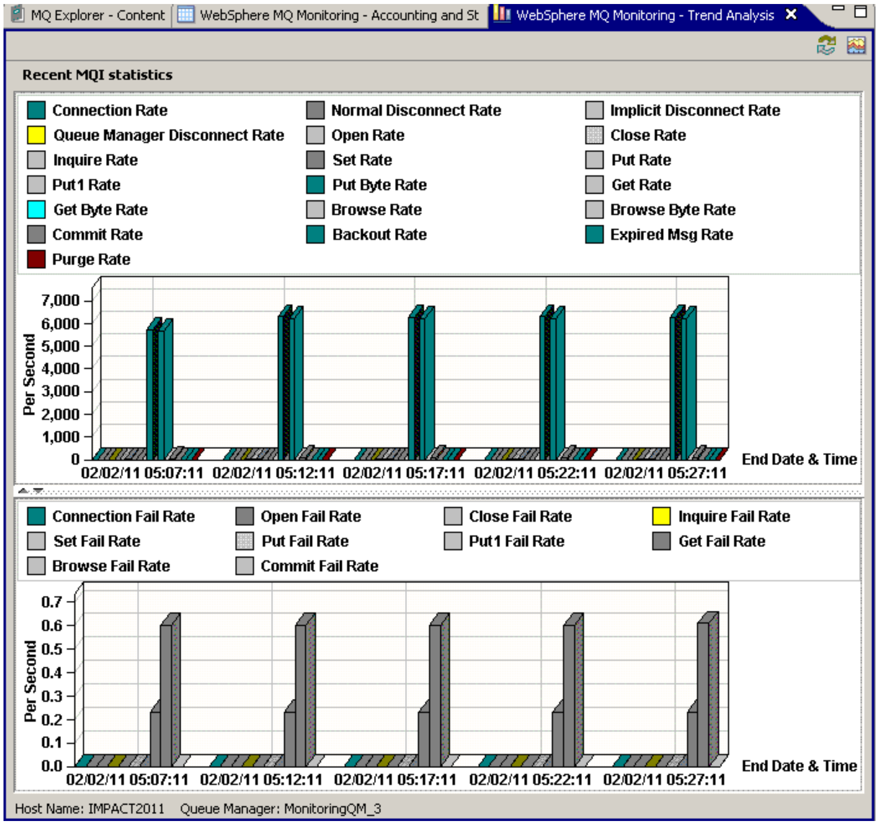
Open up the WMQ Explorer via the desktop shortcut if you haven’t already done so and ensure both MonitoringQM_2 and MonitoringQM_3 are in a running status.

- Display the “MQI Statistics” view within the WMQ Explorer for the queue manager MonitoringQM_3

Right click on the queue manager MonitoringQM_3 and select Trend Analysis → Recent MQI Statistics



You should now see 2 new views in the WMQ Explorer. The “WebSphere MQ Monitoring – Trend Analysis” view shows the statistics in a graphical form and the “WebSphere MQ Monitoring – Accounting and Statistics” view shows the output in tabular form.



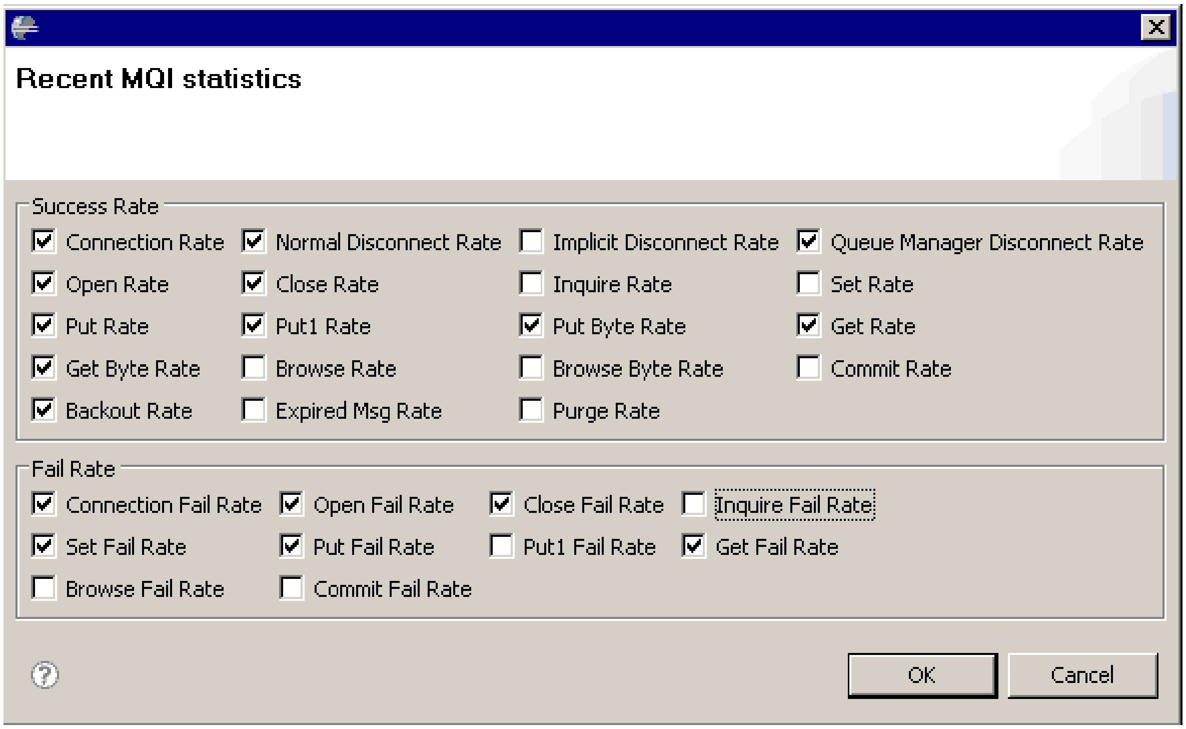
Host Name: IMPACT2011 Queue Manager: MonitoringQM_3

- You will notice that the “open rate” and “open count” count columns of the “WebSphere MQ Monitoring – Accounting and Statistics” view have positive values even though we are not putting workload through the queue manager yet (you may need to scroll across the table to see these values). This is because the WMQ Explorer uses internal queues as part of its normal processing. You can make a note of these values and compare them to the values you get later in the lab after sending some messages to the queue manager.

- Tip: You can check which queues have open input or output counts by right clicking on “Queues” in the WMQ Explorer navigator view under MonitoringQM_3 and choosing “status”

- If you click on the graph symbol in the top right hand side of the “WebSphere MQ Monitoring – Trend Analysis” view you can select the output that you want displayed in the WebSphere MQ Explorer views.

Click on the 📊 symbol and alter the selected statistics to match the screenshot below:



Recent MQI statistics

Success Rate

☑ Connection Rate ☑ Normal Disconnect Rate □ Implicit Disconnect Rate ☑ Queue Manager Disconnect Rate
☑ Open Rate ☑ Close Rate □ Inquire Rate □ Set Rate
☑ Put Rate ☑ Put1 Rate ☑ Put Byte Rate ☑ Get Rate
☑ Get Byte Rate □ Browse Rate □ Browse Byte Rate □ Commit Rate
☑ Backout Rate □ Expired Msg Rate □ Purge Rate

Fail Rate

☑ Connection Fail Rate ☑ Open Fail Rate ☑ Close Fail Rate □ Inquire Fail Rate
☑ Set Fail Rate ☑ Put Fail Rate □ Put1 Fail Rate ☑ Get Fail Rate
□ Browse Fail Rate □ Commit Fail Rate

Click OK

- Send some messages to the queue manager to see the effect on the statistics. The MonitorTestDriver_PutToMany.cmd file in C:\Impact2011\MonitoringLab\mp08 lab sends 5000 messages to the alias queue MONITOR_TOPIC. This alias points to the topic object MONITOR_TOPIC which has 4 subscribers with the following subscription queues:
    - MONITORQ_1
    - MONITORQ_2
    - MONITORQ_3
    - MONITORQ_4

Open up the short cut to the mp08 lab directory on the desktop and double click on the MonitorTestDriver_PutToMany.cmd file.

MonitorTestDriver_PutToMany...
Windows NT Command Script
1 KB

- Return to the WMQ Explorer window and the “Statistics and Accounting tab”. You should start to see changes in the statistics, including an increase in the “Put Rate” and “Put Count”. You will notice that the “Put Count” doesn’t exactly match the number of messages sent by the test application. This is because the statistics are taken over a 300 second interval and some messages will have been put outside the interval and will fall into the next set of statistics.

- Compare the new values of the “open rate” and “open count” parameters to the ones you made a note of before.

Although it is useful to view the statistics in the MQ Explorer, if you want to analyze the data, you also have the ability to export the statistics to a spreadsheet.

Click on the table symbol on the top right hand side of the “WebSphere MQ Monitoring – Accounting and Statistics” View. Save the file in C:\Impact2011\MonitoringLab\mp08 lab. You can rename the file or use the default provided.

Open the Windows Explorer in the directory C:\Impact2011\MonitoringLab\mp08 lab. There is a shortcut on the desktop to this directory. The output is saved as a .csv file so can be opened by most spreadsheets. In this case it should be opened with Lotus Symphony. When you open the file in Lotus Symphony a “Text Import” window will open. The “Separator options” should be set to “Separated by comma”. Click okay to accept the Text Import options.

- All of the statistics you have collected should now be viewable in a spreadsheet and can be used for analysis, you may want to compare the results at different times of day for example.

Close the spreadsheet when you have finished viewing the data that has been produced.

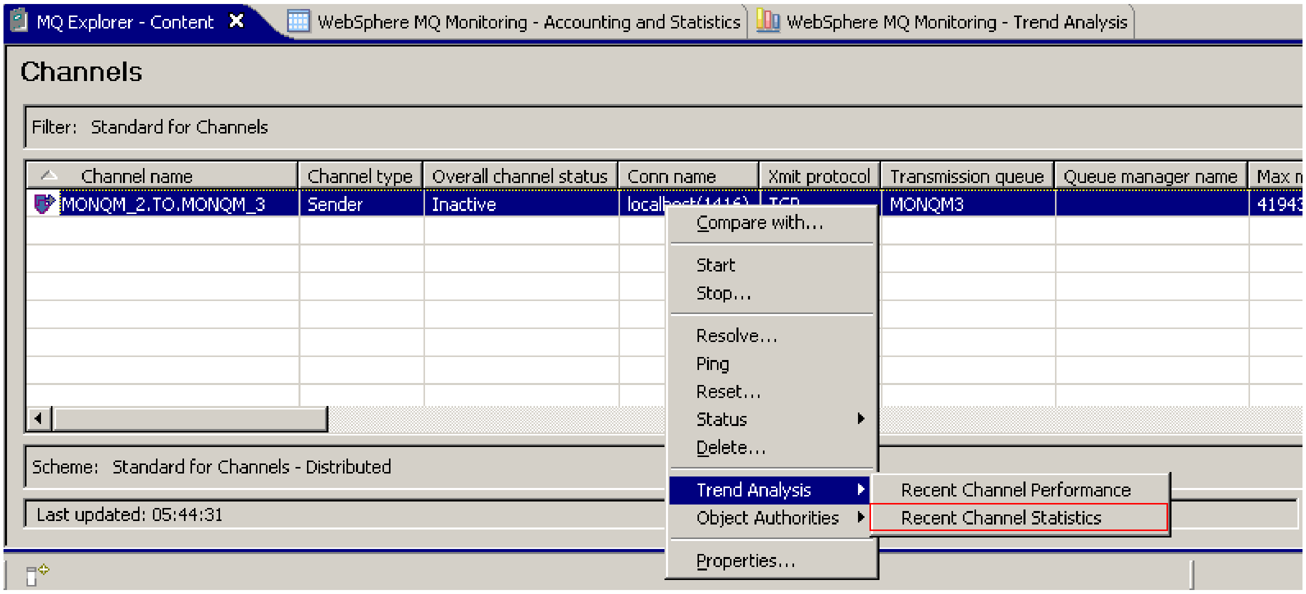
2.2.2 Capturing Channel Statistics using the Performance Monitor for WebSphere MQ

The Performance Monitor enables you to capture accounting and statistics data on queue manager, queue and channel objects. The previous exercise showed an example of collecting MQI statistics data at the queue manager level. This exercise will demonstrate how to gather channel statistic information and the sort of data that can be captured.

For this exercise, we will be collecting statistics data for the channel MONQM_2.TO.MONQM_3 which is defined between MonitoringQM_2 and MonitoringQM_3. If you display the channel in the WebSphere MQExplorer it should currently be in an inactive status.

- Start collecting statistics data for the channel MONQM_2.TO.MONQM_3

In the WebSphere MQ Explorer display the sender side of the channel which is defined on MonitoringQM_2 and right click on the channel in the content view. Select “Trend Analysis → Recent Channel Statistics”


MQ Explorer - Content
WebSphere MQ Monitoring - Accounting and Statistics
WebSphere MQ Monitoring - Trend Analysis
Channels

Filter: Standard for Channels

Channel name	Channel type	Overall channel status	Conn name	Xmit protocol	Transmission queue	Queue manager name	Max m
MONQM_2.TO.MONQM_3	Sender	Inactive	localhost(1416)	TCP	MONQM3		41943

Scheme: Standard for Channels - Distributed

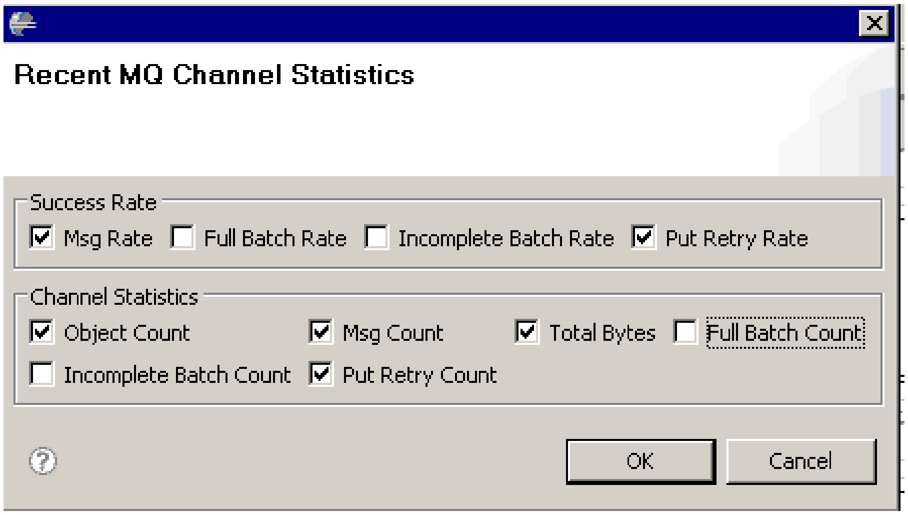
Last updated: 05:44:31

Compare with...
Start
Stop...
Resolve...
Ping
Reset...
Status
Delete...
Trend Analysis
Object Authorities
Properties...
Recent Channel Performance
Recent Channel Statistics

The views “WebSphere MQ Monitoring – Accounting and Statistics” and “WebSphere MQ Monitoring-Trend Analysis” will be empty to start with as there is no data to collect at the moment.

- Select the filters you want to apply when collecting the channel statistics data. Click on the graph symbol in the top right hand side of the “WebSphere MQ Monitoring – Trend Analysis”

Click on the symbol and select the following options:



Recent MQ Channel Statistics

Success Rate

☑ Msg Rate ☐ Full Batch Rate ☐ Incomplete Batch Rate ☑ Put Retry Rate

Channel Statistics

☑ Object Count ☑ Msg Count ☑ Total Bytes ☐ Full Batch Count

☐ Incomplete Batch Count ☑ Put Retry Count

Click OK

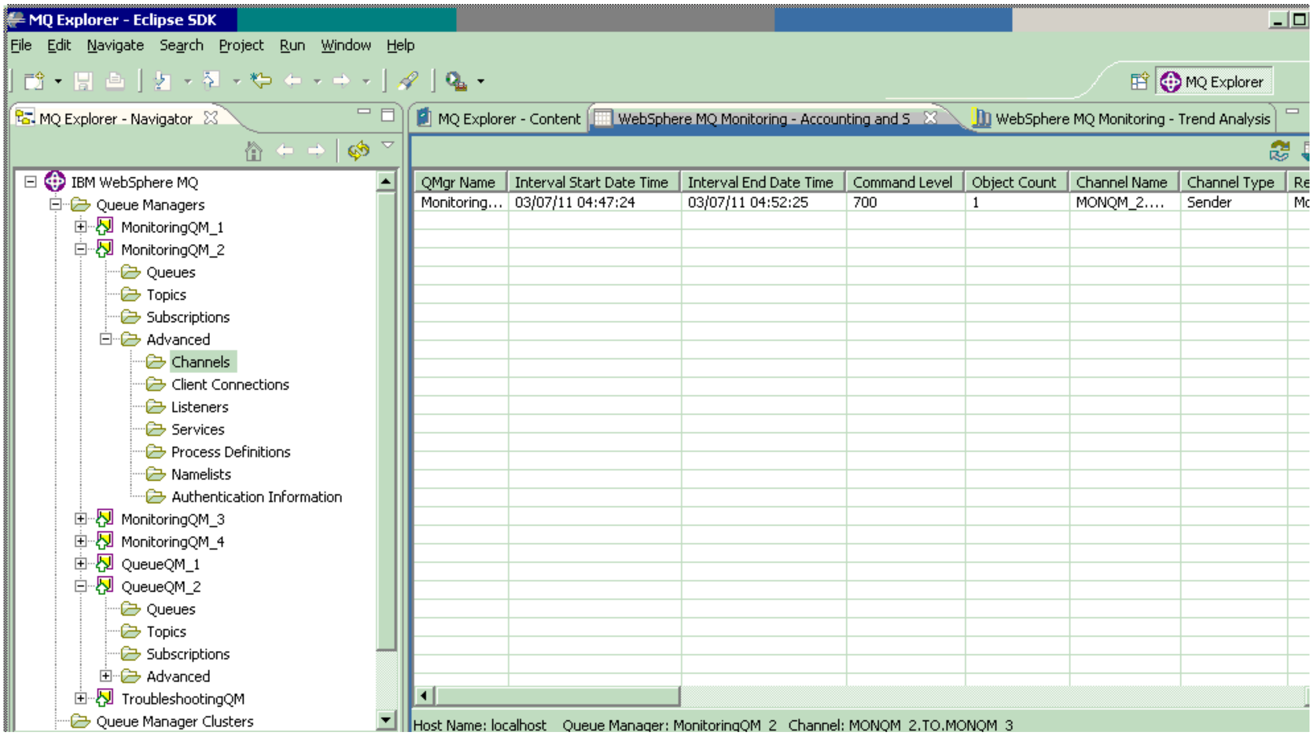
- Send some messages across the channel to see the data that is collected by the WebSphere MQ Performance Monitor.

In the directory C:\Impact2011\MonitoringLab\mp08 lab there is a command file called “MonitorTestDriver_PutRemote.cmd” that sends 4 batches of 5000 messages across the channel MONQM_2.TO.MONQM_3

Open the directory C:\Impact2011\MonitoringLab\mp08 lab using the Windows Explorer, if you don’t already have it open from the previous exercise. (there is a short cut to the directory on the desktop).

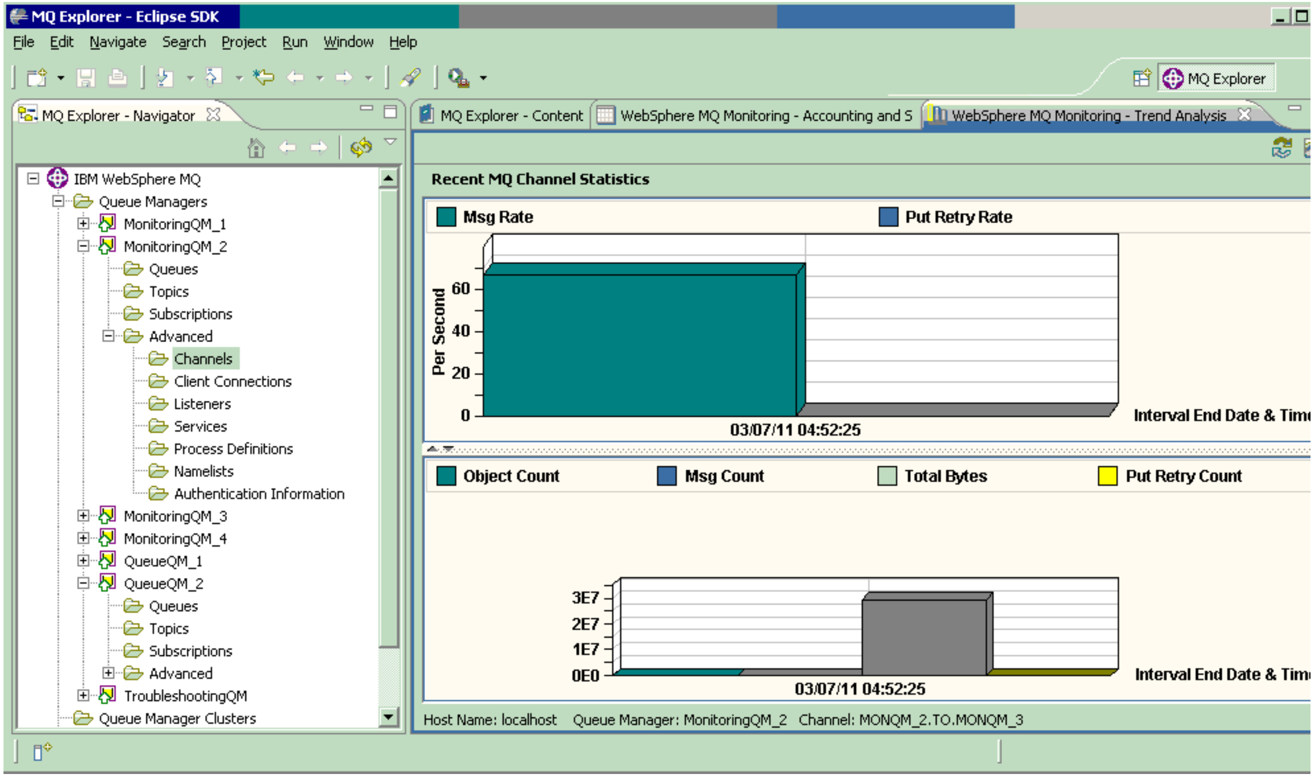
Double click on the file “MonitorTestDriver_PutRemote.cmd”. MonitorTestDriver_PutRemote.cmd

- Return to the WebSphere MQ Explorer. The channel MONQM_2.TO.MONQM_3 should now be in a running state. After several minutes, the views “WebSphere MQ Monitoring – Accounting and Statistics” and “WebSphere MQ Monitoring- Trend Analysis” will start to populate with data. It can take several minutes for these views to populate so you may want to go on to the next exercise and come back and view the output later. Clicking the refresh button on the top right of the view after 5 to 10 minutes should allow it to populate.
- This is the sort of output you should see:



QMgr Name	Interval Start Date Time	Interval End Date Time	Command Level	Object Count	Channel Name	Channel Type	Re
Monitoring....	03/07/11 04:47:24	03/07/11 04:52:25	700	1	MONQM_2....	Sender	Mo

Host Name: localhost Queue Manager: MonitoringQM_2 Channel: MONQM_2.TO.MONQM_3



Host Name: localhost Queue Manager: MonitoringQM_2 Channel: MONQM_2.TO.MONQM_3

- The following information may be of particular interest if you are monitoring the performance of the channel and you suspect you may have network issues:

“NetTime Min/Avg/Max” = The Min/Avg and Max values for the channel round trip in the recording interval in microseconds.

This can help you identify if the network is slow. These indicators are measured when the sending channel asks its partner for a response. This happens at the end of each batch and when a channel is idle during heartbeating. If the indicator shows that round trips are taking longer than expected you can then use network monitoring tools to investigate the performance of the network.

- You should see the “Total Bytes” and “Message Count” values increase to reflect the number and size of messages sent across the channel during the interval. This includes data for persistent and nonpersistent messages

- The “Exit Time” values are set to n/a as there are no exits defined on this channel. If you are using exits these statistics will be useful to determine the amount of time spent executing the exit during the interval.

Close the Windows Explorer and Command prompt windows you have open

2.3 MO01 Event Monitor for WebSphere MQ

The Event Monitor for WebSphere MQ waits on an event queue and alerts the user when an event message has arrived on the queue. It prints out the data in the message to stdout in a readable form. The source code is a useful example of how to read PCF messages for you to develop your own event monitoring applications.

WebSphere MQ enables you to generate a number of different types of event depending on the information that you require, these are the types of event that you can produce:

Queue manager events
Queue manager events are related to the use of resources within queue managers. For example, a queue manager event is generated if an application tries to put a message on a queue that does not exist.

Channel and bridge events
Channels report these events as a result of conditions detected during their operation. For example, when a channel instance is stopped.

Performance events
Performance events are notifications that a resource has reached a threshold condition. For example, a queue depth limit has been reached.

Configuration events
Configuration events are generated when a configuration event is requested explicitly or automatically when an object is created, modified, or deleted.

Command events
Command events are reported when an MQSC or PCF command runs successfully. Command events are available on WebSphere MQ for z/OS® only.

Logger events
Logger events are reported when a queue manager that uses linear logging starts writing log records to a new log extent or, on i5/OS®, to a new journal receiver. Logger events are not available with WebSphere MQ for z/OS.

For the purpose of this lab we will give an example of monitoring for Queue Manager Events and Performance Events.

2.3.1 Monitoring for Queue Manager Events using the Event Monitor for WebSphere MQ

Queue manager events are related to the use of resources within queue managers. For example, a queue manager event is generated if an application tries to put a message on a queue that does not exist, an attempt to put to a queue that is put inhibited or a user tries issue an MQI command they are not authorized to.

The following section of the Infocenter gives details of all the types of queue manager events available:

http://publib.boulder.ibm.com/infocenter/wmqv7/v7r0/topic/com.ibm.mq.csqzax.doc/mo10440_.htm

When a queue manager event occurs a message is sent to the SYSTEM.ADMIN.QMGR.EVENT queue.

For the purpose of this lab, the following queue manager has been defined:

MonitoringQM_4

The following object been created on the queue manager:

Local Queues
QLOCAL MAXDEPTH(4800) QDPHIEV(ENABLED) QDPMAXEV(ENABLED) QDEPTHHI(80)

Local queue manager events have been enabled on MonitoringQM_4 e.g in runmqsc the following command has been issued:

ALTER QMGR LOCALEV(ENABLED)

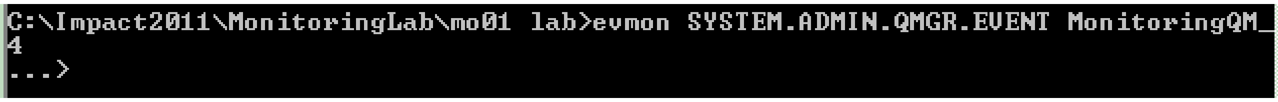
- Start the event monitor

Open a command prompt (using the short cut provided on the desktop) and change directory to C:\Impact2011\MonitoringLab\mo01 and then start the event monitor “evmon.exe” to read the event queue SYSTEM.ADMIN.QMGR.EVENT on the queue manager MonitoringQM_4 as shown below:

➢ cd C:\Impact2011\MonitoringLab\mo01 lab

➢ evmon SYSTEM.ADMIN.QMGR.EVENT MonitoringQM_4

You should get a prompt to show that the event monitoring is started and is waiting for messages to arrive on the event queue:



```
C:\Impact2011\MonitoringLab\mo01 lab>evmon SYSTEM.ADMIN.QMGR.EVENT MonitoringQM_
4
...>
```

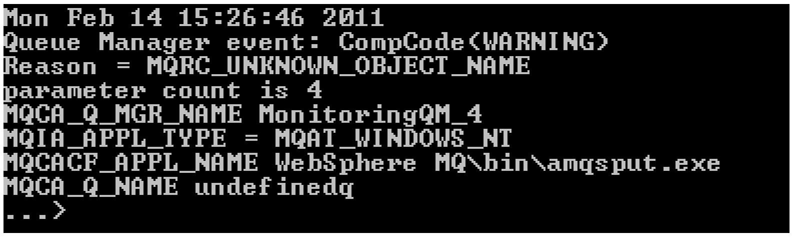
- Put a message on to a non-existent queue using amqsput

Open another command prompt and issue the following command :

- amqsput undefinedq MonitoringQM_4

This command should fail with the reason code 2085

- Return to the first command prompt and you should see that the evmon application has picked up an event that has been generated notifying the user that an attempt has been made to put to a queue that hasn't been defined:



```
Mon Feb 14 15:26:46 2011
Queue Manager event: CompCode(WARNING)
Reason = MQRC_UNKNOWN_OBJECT_NAME
parameter count is 4
MQCA_Q_MGR_NAME MonitoringQM_4
MQIA_APPL_TYPE = MQAT_WINDOWS_NT
MQCACF_APPL_NAME WebSphere MQ\bin\amqsput.exe
MQCA_Q_NAME undefinedq
...>
```

Close both the open command prompts.

2.3.2 Monitoring for Performance Events using the Event Monitor for WebSphere MQ

Performance events are notifications that a resource has reached a threshold condition, for example a queue depth limit has been reached. The following section of the manual describes all the performance events that are available:

http://publib.boulder.ibm.com/infocenter/wmqv7/v7r0/topic/com.ibm.mq.csqzax.doc/mo10800_.htm

Performance events have been enabled on queue manager MonitoringQM_4 e.g in runmqsc the following command has been issued:

ALTER QMGR PERFMEV(ENABLED)

Queue Depth High and Queue Depth Max Events have also been enabled on the queue QLOCAL. The Queue Depth High Limit has been set to 80% of the Maximum Queue Depth which is 4800 e.g the following command has been issued within runmqsc:

ALTER QL(QLOCAL) MAXDEPTH(4800) QDPHIEV(ENABLED) QDPMAXEV(ENABLED) QDEPTHHI(80)

- Start the Event Monitor for WebSphere MQ and instruct it to wait for messages on the performance event queue SYSTEM.ADMIN.PERFM.EVENT on queue manager MonitoringQM_4.

Open a command prompt by using the shortcut on the desktop and change directory to "C:\Impact2011\MonitoringLab\mo01 lab" and start the evmon executable by using the following commands:

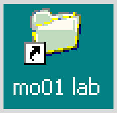
- cd C:\Impact2011\MonitoringLab\mo01 lab
- evmon SYSTEM.ADMIN.PERFM.EVENT MonitoringQM_4

As the Queue Depth High Event for the queue QLOCAL has been set to 80% of its maximum capacity, an event should be generated when 3840 messages have been sent to the queue. An event will also be generated if the maximum queue depth of 4800 messages is reached.

For this exercise we will use the EventTestDriver.cmd file in "C:\Impact2011\MonitoringLab\mo01 lab" to send 5000 messages to QLOCAL on MonitoringQM_4.

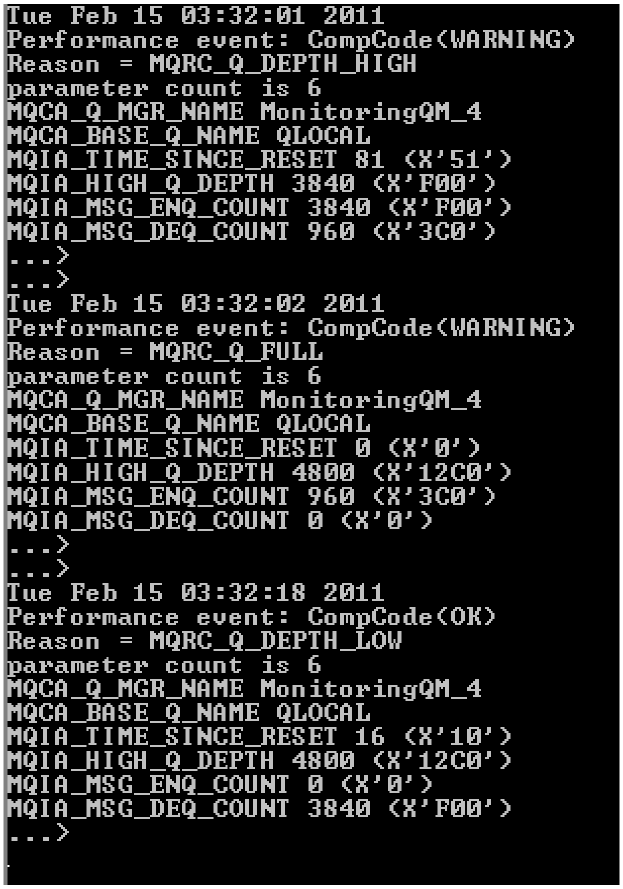
- Start the EventTestDriver.cmd file

Open the folder “C:\Impact2011\MonitoringLab\mo01 lab” in Windows Explorer. There is a short cut to this folder on the desktop.



Double click on the MonitorTestDriver.cmd File MonitorTestDriver.cmd

- Switch to the command prompt which you started the WebSphere MQ Event Monitor in. You should see the following output:



```
Tue Feb 15 03:32:01 2011
Performance event: CompCode(WARNING)
Reason = MQRC_Q_DEPTH_HIGH
parameter count is 6
MQCA_Q_MGR_NAME MonitoringQM_4
MQCA_BASE_Q_NAME QLOCAL
MQIA_TIME_SINCE_RESET 81 (X'51')
MQIA_HIGH_Q_DEPTH 3840 (X'F00')
MQIA_MSG_ENQ_COUNT 3840 (X'F00')
MQIA_MSG_DEQ_COUNT 960 (X'3C0')
...>
...>
Tue Feb 15 03:32:02 2011
Performance event: CompCode(WARNING)
Reason = MQRC_Q_FULL
parameter count is 6
MQCA_Q_MGR_NAME MonitoringQM_4
MQCA_BASE_Q_NAME QLOCAL
MQIA_TIME_SINCE_RESET 0 (X'0')
MQIA_HIGH_Q_DEPTH 4800 (X'12C0')
MQIA_MSG_ENQ_COUNT 960 (X'3C0')
MQIA_MSG_DEQ_COUNT 0 (X'0')
...>
...>
Tue Feb 15 03:32:18 2011
Performance event: CompCode(OK)
Reason = MQRC_Q_DEPTH_LOW
parameter count is 6
MQCA_Q_MGR_NAME MonitoringQM_4
MQCA_BASE_Q_NAME QLOCAL
MQIA_TIME_SINCE_RESET 16 (X'10')
MQIA_HIGH_Q_DEPTH 4800 (X'12C0')
MQIA_MSG_ENQ_COUNT 0 (X'0')
MQIA_MSG_DEQ_COUNT 3840 (X'F00')
...>
```

You will see that a Queue Depth Low has been generated even though we haven’t enabled this type of event for QLOCAL. This is because a Queue Depth Low Event is automatically enabled by a Queue Depth High event or a Queue Full event on the same queue.

The Event Message includes statistics relating to the event including the following:

MQIA_TIME_SINCE_RESET – The elapsed time since statistics were last reset (statistics are reset when an event is generated but it is also possible to reset them using a PCF command and if a qmgr is recycled)

MQIA_HIGH_Q_DEPTH – The maximum number of messages on the queue since statistics were last reset.

MQIA_MSG_ENQ_COUNT – The number of message enqueued (the number of MQPUT calls to the queue) since statistics were last reset

MQIA_MSG_DEQ_COUNT – The number of message dequeued (the number of MQGET calls to the queue) since statistics were last reset.

You can monitor events for all queue managers in your network centrally by configuring the event queues as remote queues pointing to a central queue manager running the Event Monitor for WMQ

Close all the command prompts and Windows Explorer sessions you have open.

Part 3: Moving Messages between queues

The purpose of this section of the lab is to demonstrate how to move messages between queues and files. You will learn how message format or properties can be changed using these tools. Links have been provided to each SupportPac page to allow you to download the SupportPac as well as the documentation. For more details and examples refer to this documentation

1. **MO03 WMQ Queue Load/ Unload Utility**

   This SupportPac provides a simple command line application for loading and unloading messages off queues.

   http://www-01.ibm.com/support/docview.wss?rs=171&uid=swg24009368

2. **IH03 WMB V7 Message display, test and performance utilities (RHFUTIL)**

   This SupportPac contains utilities (rfhutil) that are useful for the development and testing of WebSphere MQ and WebSphere Message Broker Version 7 applications.

   http://www-01.ibm.com/support/docview.wss?rs=203&uid=swg24000637

   The following table gives a comparison of both utilities:

MO03 SupportPac (qload utility)	IH03 SupportPac (RFHutil utility)
Command line	GUI
Day to Day Operations	Development and Testing
Easier readability in file format	Quick and easy way to see what's on the queue
Archiving Messages from a queue to a file	Debugging
Allows you to easily edit the data	Full control over headers but cannot modify data
Easy to move message between queues such as in unloading/loading thousands of messages from a backout queue back to the input queue after an issue has been resolved OR get messages that could not be processed in a production environment to replay them into a test environment.	Data in the GUI has easy readability with different tabs for the various pieces of an MQ message.
Category 2 SupportPac	Category 2 SupportPac

3.1 MO03 WMQ Queue Load/ Unload Utility

There are a number of situations where this utility might be very handy:

- Multiple instances of an application are processing messages from parallel queues and one of the applications (perhaps running on a separate machine using an MQ client connection) becomes unavailable, so messages need to be re-routed to another queue
- Repetitively replaying a known series of messages for QA or performance testing

- Selectively manipulate message contents that are preventing message processing

This lab is based on the SupportPac documentation for this utility. There are examples beyond what is in this lab to help further your knowledge of this utility. The documentation for the SupportPac can be downloaded from the link above. During this lab, the documentation can be found in the SupportPacs folder on the desktop of the VMImage under MO03.

The parameters used in the first command are described. After that, only the new parameters for a command are described. For more parameter details, please refer to the SupportPac documentation – Chapter 3 “Parameters.”

For this lab we will be using the following MQ objects:

Queue manager: **QueueQM_1**

Queues: **QueueA, QueueB**

QueueA has been pre-loaded with 20 messages.

Open the WMQ Explorer using the shortcut on the desktop if it is not already open.

These messages can be seen in the open MQ Explorer window if you browse the message on QueueA (by right clicking on “QueueA” and selecting “Browse Messages”):

```
This is the first test message
This is the second test message
Here is the third test message
Welcome to IMPACT 2011
Are you having fun in Las Vegas?
Here is another test message
IMPACT is great!
Here is test message number 8
Las Vegas is hot!
This is the tenth test message
This is the eleventh test message
Here is test message 12
And here is message 13
Hope you are doing great!
Message 15
Message 16
Message 117
Message 18
Message 19
Message 20
```

3.1.1 Move and copy messages from one queue to another.

We will start by moving all of the messages from QueueA to QueueB. There are various reasons why you would do this but one is that you cannot rename queues. So if you created a new queue, you could simply copy messages over to it before deleting it.

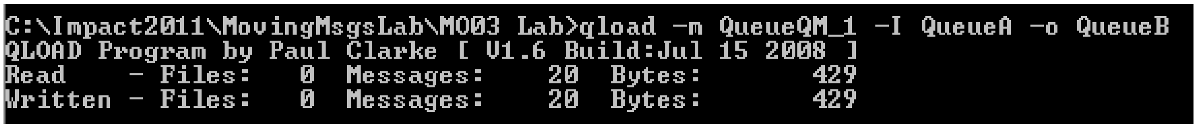
To invoke the MO03 utility, you need to be in the correct directory. For this lab, it will always be C:\Impact2011\MovingMsgsLab\MO03 Lab.

- Open the command prompt from the icon on the desktop. Change directory to C:\Impact2011\MovingMsgsLab\MO03 Lab and start the qload utility, with the following parameters:

```
> cd C:\Impact2011\MovingMsgsLab\MO03 Lab
> qload -m QueueQM_1 -I QueueA -o QueueB
```

```
where:
-m is the name of the queue manager to connect to
-I specifies the name of the input queue to consume messages from
-o is the output queue name
```

Once the utility has completed executing, it will output a summary of what it did. For example:

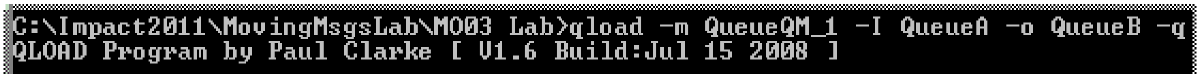


```
C:\Impact2011\MovingMsgsLab\MO03 Lab>qload -m QueueQM_1 -I QueueA -o QueueB
QLOAD Program by Paul Clarke [ V1.6 Build:Jul 15 2008 ]
Read    - Files:   0  Messages:    20  Bytes:       429
Written - Files:   0  Messages:    20  Bytes:       429
```

To suppress this summary, use the quiet option, -q e.g:

```
> qload -m QueueQM_1 -I QueueA -o QueueB -q
```

```
-q quiet (no summary)
```



```
C:\Impact2011\MovingMsgsLab\MO03 Lab>qload -m QueueQM_1 -I QueueA -o QueueB -q
QLOAD Program by Paul Clarke [ V1.6 Build:Jul 15 2008 ]
```

Going forward, we will NOT be suppressing the summary.

In IBM WebSphere MQ Explorer, verify that QueueA now has a depth of 0 and QueueB now has a depth of 20.

- Now let's say you wanted to copy all of the messages from one queue to another for either backup or testing purposes. You may want to run an application or WebSphere Message Broker flow that destructively reads off a queue but don't want to have to re-populate the queue each time.

- Issue the following command from the command prompt to copy the messages from QueueB to QueueA:

```
> qload -m QueueQM_1 -i QueueB -o QueueA
```

```
where:
-i specifies the name of the input queue to browse messages from
      NOTE: -I is destructively gets from the queue where -i only
      browses a queue
```

In IBM WebSphere MQ Explorer, verify that both QueueA and QueueB have a depth of 20.

3.1.2 Unload a queue to a file or files

You may want to save messages on a queue to a file to possibly archive or later reload them back to a queue. Issue the following command from the command prompt to take a copy of the messages from the queue and saves them in the file specified.

```
> qload -m QueueQM_1 -i QueueA -f “c:\Impact2011\MovingMsgsLab\MO03
Lab\MO03LabFile1”
```

```
where:
-f is the target filename
```

- You should still have a dept of 20 on both QueueA and QueueB. But now you should also have a file in the C:\Impact2011\MovingMsgsLab\MO03 Lab directory called "MO03LabFile1" or whatever name you used.

Open a Windows Explorer window for the directory C:\ Impact2011\MovingMsgsLab\MO03 Lab (there is a shortcut to the MO03 Lab directory available on the desktop).

- Click to open the file in the editor of your choice (Notepad and Wordpad are installed on the VMWare image). As you can see, the message data is in hex. If you specify the –da option at the end of your command, the file will show the hex value of your message data on the left and any displayable characters on the right (note that we are using upper case –I not lower case –i now).

```
> qload -m QueueQM_1 -I QueueA -f "c:\Impact2011\MovingMsgsLab\MO03
Lab\MO03LabFile2" -da
```

- The depth of QueueA should now be 0 and you should also have a file in the c:\Impact2011\MovingMsgsLab\MO03 Lab directory called "MO03LabFile2" or whatever name you used.

Going forward, we will be using the –da option to also show displayable characters.

This program is capable of writing and reading to multiple files in a single command. To facilitate this, special characters can be placed in the file name which will be replaced with a value when the program runs. Inserts are preceded by the % character in the file name. In this example, we have used the 'n' parameter which is for the current file number. For more options, refer to chapter 3 of the documentation on page 8.

- In the command prompt issue the following command:

```
> qload -m QueueQM_1 -I QueueB -f "c:\Impact2011\MovingMsgsLab\MO03
Lab\MO03LabFile3_%n" -da
```

where:

```
-f is the target file name and %n increments each filename by 1
```

- The depth of QueueB should now be 0 also. You should also have 20 additional files in the c:\Impact2011\MovingMsgsLab directory called "MO03LabFile3_*n*" or whatever name you used where n=n+1 starting with n=1.

3.1.3 Load a queue from a file

There are two sets of context information in the Message Description (MQMD) – the identity context fields and origin context fields. They are described in more detail in the WebSphere MQ Application Programming Guide. The default action of this utility is to set all the context information in the MQMD to that which was saved in the file being loaded. This required the user ID under which the utility is running to have the appropriate authority to set all the context fields. These fields can be manipulated using the –C flag. Details of this flag can be found on page 11 of the documentation.

- Now that we have unloaded QueueA to a single file and QueueB to 20 different files, we can make modifications and reload them back to the queues. You may want to do this because you are unloading a queue from one environment such as production and uploading it to a new environment such as test to determine why there are issues.

Using this utility, you will not have to recreate all of your messages and can preserve the integrity of the messages.

- Open MO03LabFile1 in C:\Impact2011\MovingMsgsLab\MO03 Lab with the editor of your choice (notepad or wordpad). Search for the first few occurrences of “Administrato” (the r has been truncated in the file – this is not a typo) and replace with *Your Name* and save and close the file.

- After completing this, load the file to QueueA using the following command from a command prompt.

```
>qload -m QueueQM_1 -o QueueA -f “c:\Impact2011\MovingMsgsLab\MO03
Lab\MO03LabFile1”
```

```
where:
-f is the source filename
```

- QueueA should once again have 20 messages. Browse the messages using WMQ Explorer and note the change in user name on some of the messages from Administrator to *Your Name.*

Now that you have uploaded a single file to a single queue, let us explore uploading multiple files to a single queue.

We are going to load the 20 files MO03LabFile3_*n* to QueueB. It is possible to change values within the messages before uploading them.

- Open MO03LabFile3_17 and change the value of “Message 117” to be “Message17”. In order to do this you will need to modify the last line of code in the file by removing one of the occurrences of X’31’ at the end of the line so it changes

FROM:

```
X 4D6573736167652031313137                                  <Message 117>
```

TO:

```
X 4D65737361676520313137                                    <Message 17>
```

- Save and close the file

- Issue the following command from a command prompt

```
> qload -m QueueQM_1 -o QueueB -f “c:\Impact2011\MovingMsgsLab\Mo03
Lab\MO03LabFile3_%n”
```

```
where:
-f is the source or target file name – in this case it is the source
%n increments each filename by 1
```

- QueueB should once again have 20 messages. Browse the messages on the queue in the WMQ Explorer. You should see that the message data for the message we altered in Position 17 has been changed to 17.

3.1.4 Copy the first 10 messages from one queue to another

One of the other features of this SupportPac is that you can copy a selection of messages. You can choose the first *x* number of messages, from message *x* to message *y* and so on.

For our example, we are going to want to copy the first 10 messages from QueueA to QueueB. Type the following command in the command prompt:

```
> qload -m QueueQM_1 -i QueueA -o QueueB -r#10
```

```
where:
-r#10 specifies the first 10 messages
```

Note that we copied the messages, we didn't move them so QueueA should still have a depth of 20 but now QueueB has a dept of 30.

This completes the first part of this exercise. If you would like to explore this SupportPac in more detail, please continue onto the optional section.

OPTIONAL:

3.1.5 Message Age

You can display the ages of messages on a queue to see how long they have been on that queue. After doing this, you may choose to manually process messages older than a certain time interval as they are no longer pertinent within your application because they have "expired". This interval can be specified in days, hours and minutes.

- This command will display the actual message age on your screen:

```
> qload -m QueueQM_1 -i QueueA -f stdout -dT
```

```
where:
-f is the source or target file name
      stdout displays to the screen
-dT displays the time the message has been on the queue
```

- Once you have done this, you can get more specific detail such as display messages older than 15 minutes:

```
> qload -m QueueQM_1 -i QueueA -f stdout -T15
```

```
where:
-T with the number 15 browses messages that are older than 15
minutes
```

In this lab, this will most likely display ALL of your messages. Make your own modifications to this command based on the age of your messages to view various outputs. Additional examples of using message age are available in the SupportPac documentation on page 15.

3.1.6 File Insert

This SupportPac program is capable of writing and reading to multiple files in a single command. This can be useful if you wish to unload a queue where each message goes to a separate file or if you want to load multiple files to a queue as we did in our previous exercises. When we copied messages from a queue to multiple output files using the %n option (`qload -m QueueQM_1 -I QueueB -f c:\Impact2011\MovingMsgsLab\MO03 Lab\MO03LabFile3_%n -da`), that was an example of using File Insert.

The full list of inserts can be found beginning on page 8 of the documentation.

You may want to add the current date and time to an output file to preserve when these messages were copied or removed from the queue and saved for backup or archiving purposes:

```
> qload -m QueueQM_1 -I QueueA -f
“c:\Impact2011\MovingMsgsLab\MO03 Lab\MO03LabFile4_%c_%t”
```

```
where:

-f is the target file name
      _%c appends the current date to the filename
      _%t appends the current time to the filename
```

The depth of QueueA should now be 0. You should also see a new file in the C:\Impact2011\MovingMsgsLab directory called MO03LabFile4 with a date followed by a time appended to the name.

You may have an application that writes files with a specified name and may overwrite your existing files or if you choose to preserve your files with a method other than the date and timestamp. In this case, you can read the files with one format and write them with a new format:

```
> qload -m QueueQM_1 -f “c:\Impact2011\MovingMsgsLab\MO03 Lab
\MO03LabFile3_%n” -da -f “c:\Impact2011\MovingMsgsLab\MO03
Lab\MO03LabFileNewName_%n”
```

```
where:
-f reads files MO03LabFile3_%n
-da adds ASCII columns to the hex output in the file to aid
readability
-f is where the new files are written: MO03LabFileNewName_%n
```

You should now see 20 new files in the C:\Impact2011\MovingMsgsLab\MO03 Lab directory called MO03LabFileNewName with a number appended to it that is incremented by 1 from 1 through 20.

Another benefit of this utility is that it can selectively read messages from a queue that may contain a specific character or string of characters. These messages can then be preserved in their own file.

```
qload -m QueueQM_1 -I QueueB -s Message -f
“c:\Impact2011\MovingMsgsLab\MO03 Lab\MO03LabFileMessage%i”
```

```
where:
-s parameter says that only messages containing the characters
following it
-f is the destination filename
      %i is the message index
```

The depth of QueueB should now be 24 because 6 messages were removed that contained the characters “Message”. There will also be 6 new files called MO03LabFileMessage with the numbers 15 through 20 appended at the end as those are the index numbers of the files containing “Message.”

This concludes the optional portion of the MS03 exercises. Close all open Windows.

3.2 IH03 WMB V7 Message display, test and performance utilities (RHFUTIL)

This section of the lab is based on the SupportPac documentation for this utility. There are examples beyond what is in this lab to help further your knowledge of this utility. During this lab, the documentation can be found in the SupportPacs folder on the desktop of the VMImage under IH03.

NOTE that RFHUtil has a significant amount of functionality including but not limited to Performance Measurement Utilities. We are not able to touch anything beyond the basics in this lab. It is advisable to read through the documentation to learn more about all the features and benefits of this utility.

What the RFHUTIL utility does (from documentation):

The RFHUtil utility program provided with this SupportPac will read data from files and/or queues and write data to files and/or queues. It will display data in a variety of formats. The user data portion of the message can be displayed in different formats but the user data itself cannot be changed. Another program must be used to create or change the user data. The utility program will format Rules and Formatting headers (RFH) headers, CICS Information Headers (CIH), IMS Information Headers (IIH) or Dead letter headers included with messages or files it reads. It can add or remove these headers as well. The new publish and subscribe functions and message properties functions supported in Version 7 of MQ are also fully supported.

There is one exception where the user data can actually be changed. Carriage return and line feed characters can be removed when a file is read if the remove CR/LF or UNIX text options found on the Read menu is selected. If this data is subsequently written back to a file or message, the resulting data may be different than the original data in the file.
The supported headers can be changed, removed or added, and the contents of these headers can be changed or created.

For this lab we will be using the following MQ objects:

Queue Manager: **QueueQM_2**

Queues: **Queue2A, Queue2B**

Queue2A has been pre-loaded with 20 messages. These messages can be seen in the open MQ Explorer window:

```
This is the first test message
This is the second test message
Here is the third test message
Welcome to IMPACT 2011
Are you having fun in Las Vegas?
Here is another test message
IMPACT is great!
Here is test message number 8
Las Vegas is hot!
This is the tenth test message
This is the eleventh test message
Here is test message 12
And here is message 13
Hope you are doing great!
Message 15
Message 16
Message 117
Message 18
Message 19
Message 20
```

3.2.1 Browse and Read messages

- We will start by browsing the message on the Queue Manager QueueQM_2 in the queue Queue2A. You may want to do this because you are having issues with incorrect values in your messages when you are running, for example, a Message Broker message flow. This will allow you to review the various fields as well as data in your message without removing the message so it can be further processed if needed.

- To invoke the IH03 utility from the command line, you need to be in the correct directory. For this lab, it will always be C:\Impact2011\MovingMsgsLab\IH03 Lab.

- Open the command prompt from the icon on the desktop.

NOTE: For these labs, we will be running with the main version of RFHUtil, not the client version(RFHUtilc). Details on the client version can be found in the documentation.

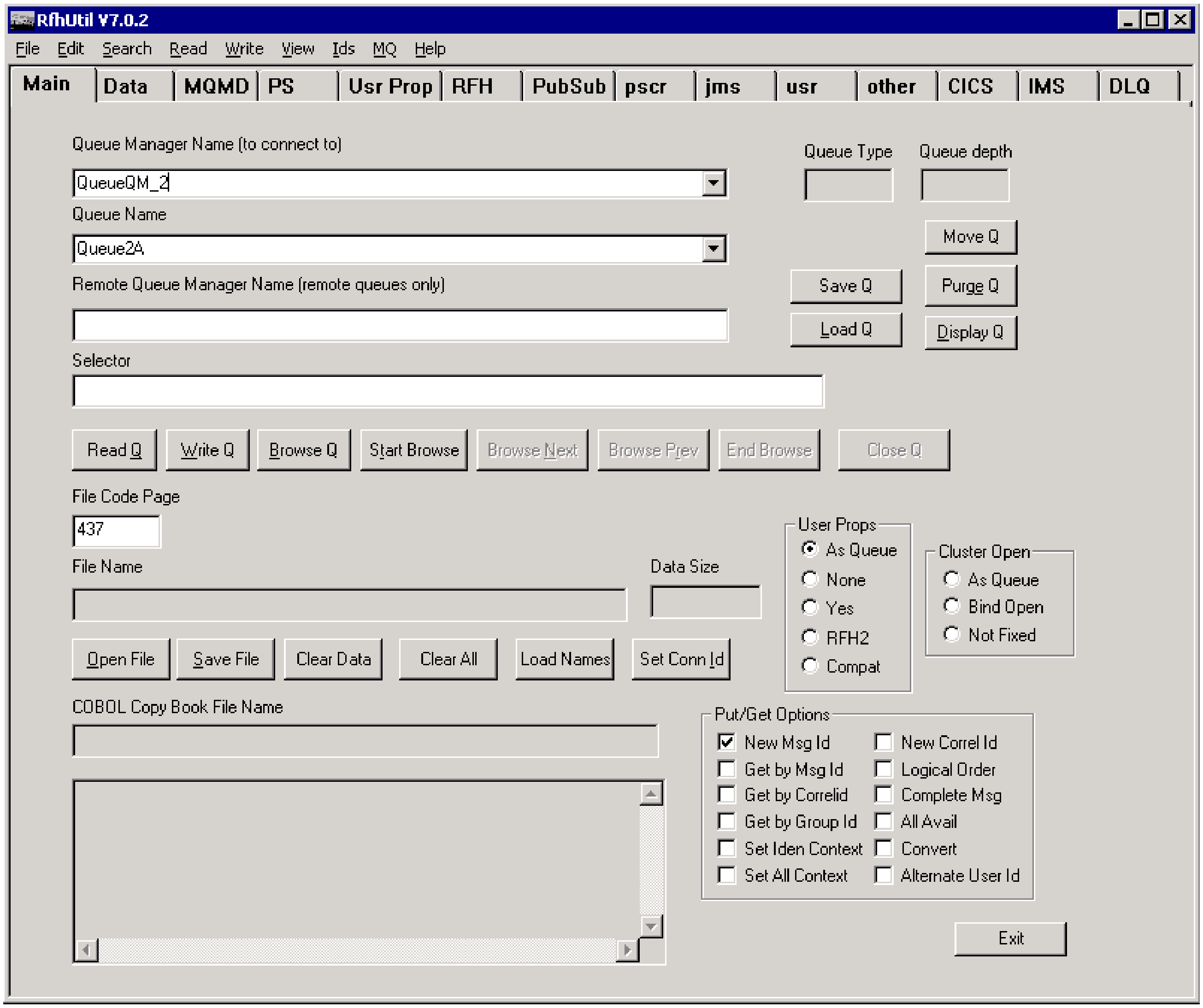
- Issue the following commands from the command prompt:

```
> cd C:\Impact2011\MovingMsgsLab\IH03 Lab

> RFHUtil QueueQM_2 Queue2A
```

```
where:
the first parameter is the queue manager name
the second parameter is the starting queue name
```

NOTE: Once the GUI is opened, these values can easily be changed. You do not need to invoke a new GUI each time unless you want to have multiple windows open to different queues/queue managers.

- Once invoked, you should see a window like this:



Queue Manager Name (to connect to): QueueQM_2

Queue Name: Queue2A

File Code Page: 437

User Props: ● As Queue ○ None ○ Yes ○ RFH2 ○ Compat

Cluster Open: ○ As Queue ○ Bind Open ○ Not Fixed

Put/Get Options:

☑ New Msg Id	□ New Correl Id
□ Get by Msg Id	□ Logical Order
□ Get by CorrelId	□ Complete Msg
□ Get by Group Id	□ All Avail
□ Set Iden Context	□ Convert
□ Set All Context	□ Alternate User Id

You will note various tabs. Here is a description of each tab. For further details on the tabs and each field on that tab reference the documentation pages listed.

***Main (page 15)** – This tab contains the five buttons provided for the purpose of reading or writing files or messages. It is also where you will designate the queue manager and queue names to perform these operations.*

***Data (page 30)** – This tab contains the data window and a number of radio buttons that control the format that the date will be displayed (Character, HEX, XML etc.). These buttons only affect the data display and do not change the contents of the data in the buffer.*

***MQMD (page 35)** – This tab contains fields that display the fields in the MQSeries message descriptor. In some cases, the contents of the display can be changed, such as the code page. Changing the fields in the MQMD does not change the data in the data buffer.*

***PS (page 38)** – This tab supports the new Pub/Sub functions contained in MQ V7. The functions allow subscriptions to be created and used and for messages to be published and received. The functions on this tab require that version 7 of MQ be installed on the system that RFHUtil is running on and that the queue manager that RFHUtil is connected to must be a V7 queue manager.*

***UsrProp (page 43)** – This tab supports the user properties that were introduced in MQ V7.*

***RFH (page 45)** – This tab contains the RFH(Rules and Formatting) header information. This iptional data header gives the Message Broker information about the format of the message date. RFH is contained at the front of the user data in a message.*

***PubSub (page 48)** – This tab is used to generate publish and subscribe requests. This tab will allow most but not all publish/subscribe commands to be entered or displayed. Please be aware that different publish and subscribe brokers will not support all of the options that are available.*

***pscr (page 51)** – This tab is used to display fields within a publish/subscribe response folder.*

***jms (page 52)** – This tab is used to display and set fields within a jms folder in an RFH version 2 header.*

***usr (page 54)** – This tab is used to display and set the data in a usr folder in an RFH version 2 header. The actual data contained in RFH2 folders is described as "XML-like". Many ordinary XML constructs are not allowed in an RFH2 folder. This would include things like XML comments, CDATA sections, embedded DTDs and user defined entities. All user data is parsed character data.*

***other (page 56)** – This tab is used to display and set data in folders that are not recognized by the RFHUtil program.*

***CICS (page 57)** – This tab displays the contents of the MQSeries CICS Information header (CIH), if present.*

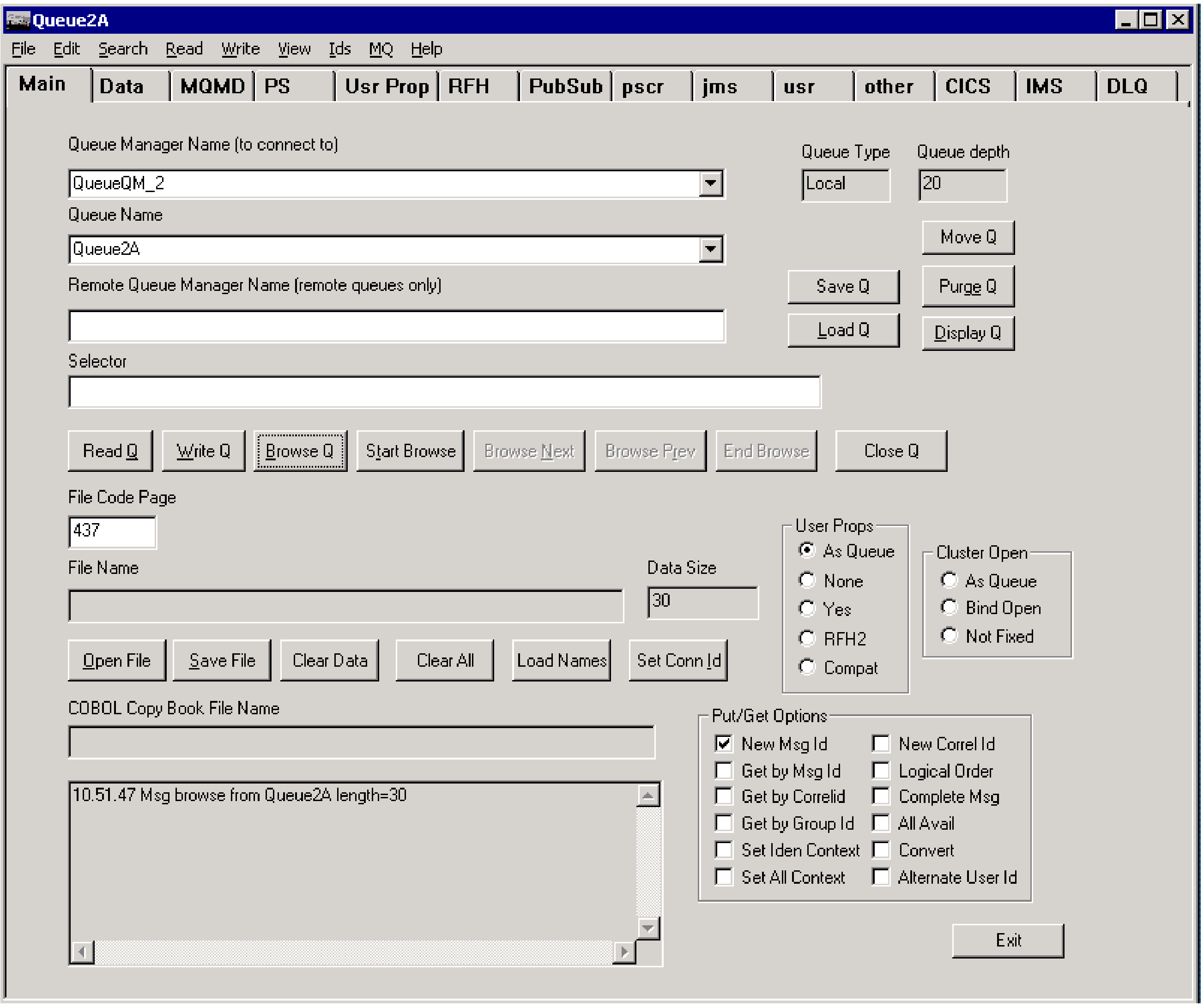
***IMS (page 61)** – This tab displays the contents of the MQSeries IMS information header, if present. If an IMS header is present, then the data must begin with an LLBB (length) field for the first message segment. The length fields are validated by MQ.*

***DLQ (page 63)** – This tab displays the contents of the MQSeries dead letter header, if present.*

Note the depth of Queue2A is 20 and the depth of Queue2B is 0.

- Click on the button "BrowseQ".

- The first message will be loaded into the GUI. You will now see the Queue depth field populated on the top right of the window as well as a message in the bottom window of the GUI stating that you are browsing Queue2A and the length of its data:



Queue Manager Name (to connect to): QueueQM_2

Queue Name: Queue2A

Remote Queue Manager Name (remote queues only):

Selector:

Queue Type: Local

Queue depth: 20

File Code Page: 437

File Name:

Data Size: 30

COBOL Copy Book File Name:

```
10.51.47 Msg browse from Queue2A length=30
```

User Props: ● As Queue ○ None ○ Yes ○ RFH2 ○ Compat

Cluster Open: ○ As Queue ○ Bind Open ○ Not Fixed

Put/Get Options:

☑ New Msg Id	□ New Correl Id
□ Get by Msg Id	□ Logical Order
□ Get by Correlid	□ Complete Msg
□ Get by Group Id	□ All Avail
□ Set Iden Context	□ Convert
□ Set All Context	□ Alternate User Id

- Traverse the tabs to see the various details that is brought into the message. Pay particular attention to the Data tab which contains the message. You can choose various display options to see your data. The MQMD tab is also useful when working with WebSphere Message Broker.

- Choose “BrowseQ” several times to scroll through the different messages and note that the depth has not changed. Also note that the data is the same as you are browsing the first message each time you press the button.

- To scroll through the rest of the messages, you will need to choose “Start Browse.” Once you do this, The “ReadQ,” “WriteQ” and “BrowseQ” buttons are now inactive but others are now active including “Browse Next.”

- Click “Browse Next” to go to the next message. You can also choose “Browse Prev” to browse the previous message. Now you will see the data in the Data tab changing and the length of the message also changing.

- When you are done browsing, choose “End Browse.”

- Now that we have browsed the messages, we will read the messages or GET them off of the queue. Perhaps in reviewing your messages, you realized there were errors in some but not all of the messages. You can selectively delete the messages that are incorrect and proceed with processing the rest of the messages.

- Choose the button “ReadQ.” Just as when you chose “BrowseQ”, the first message will be loaded into the GUI. You will again be able to traverse the tabs to see the various details brought into the message. However, this time note that the Queue Depth has been decremented by 1 and is now at a depth of 19. Every time you choose “ReadQ”, the Queue Depth will be decremented.

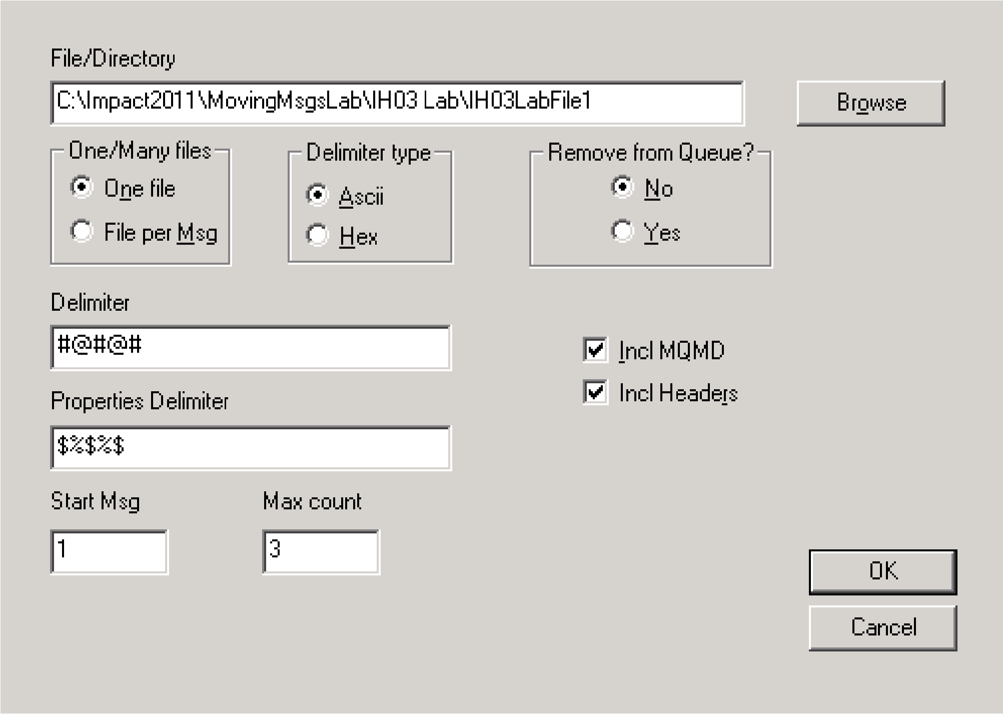
- Do this 4 more times until the depth of Queue2A is 15.

3.2.2 Write messages to a file

Now that we have done both browsing and getting messages from Queue2A, let us write 3 messages from Queue2A to a file. Just as with the MO03 SupportPac, you may want to do this because you are unloading a queue from one environment such as production and uploading it to a new environment such as test to determine why there are issues. Using this utility, you will not have to recreate all of your messages and can preserve the integrity of the messages.

- Choose the “SaveQ” button to save messages from a queue to one or more files.

- Once this button is pressed, the follow dialog is displayed:


File/Directory: C:\Impact2011\MovingMsgsLab\IH03 Lab\IH03LabFile1
One/Many files: ● One file ○ File per Msg
Delimiter type: ● Ascii ○ Hex
Remove from Queue?: ● No ○ Yes
Delimiter: #@#@#
Properties Delimiter: $%$%$
☑ Incl MQMD
☑ Incl Headers
Start Msg: 1
Max count: 3

- Here, a fully qualified file name must be provided. Specify the File/Directory to be C:\Impact2011\MovingMsgsLab\IH03 Lab\IH03LabFile1. We are also designating Start Msg to be 1 and Max Count to be 3 as we want to write only 3 messages from the queue to our file beginning with the first message.

- Click OK

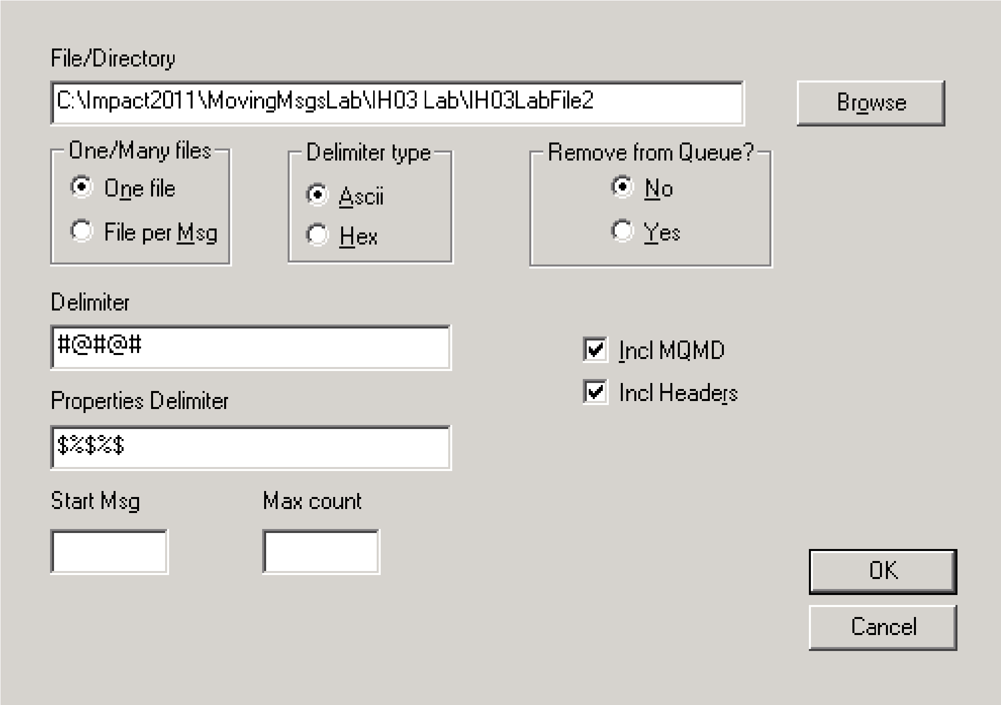
- Note the Queue depth field has not decremented because we left the value of “Remove from Queue?” as No.

- Open a Windows Explorer window in C:\Impact2011\MovingMsgsLab\IH03 Lab. There is a shortcut to this directory on the desktop called “IHO3 Lab”

- Open the file C:\Impact2011\MovingMsgsLab\IH03 Lab\IH03LabFile1 with the editor of your choice. If you choose Notepad, be sure to turn Word Wrap on for easier viewing.

- Note that with MO03 (qload), the file had better readability than with this utility.

- Repeat this process but this time save all of the messages to a queue to file C:\Impact2011\MovingMsgsLab\IHO3 Lab\IH03LabFile2. You do this by leaving the Start Msg and Max Count fields default (blank) as in the screenshot below:



File/Directory: C:\Impact2011\MovingMsgsLab\IH03 Lab\IH03LabFile2

One/Many files: ● One file ○ File per Msg

Delimiter type: ● Ascii ○ Hex

Remove from Queue?: ● No ○ Yes

Delimiter: #@#@#

☑ Incl MQMD

☑ Incl Headers

Properties Delimiter: $%$%$

Start Msg:

Max count:

3.2.3 Load a queue with messages from a file and purge first queue

Now that we have written the contents of Queue2A to a file, we will load the data to Queue2B. Once this is complete, the depth of both Queues should be identical as will the data as Queue2B currently has a depth of 0.

We have 2 options to utilize this utility with Queue2B. We can simply use the drop down button in the Queue Name field of the Main tab and choose Queue2B or we can open up a new GUI window so we don’t have to switch back and forth between the queues.

- For this exercise, we will open a new window. This time instead of doing it on the command line, we will double click on the utility in Windows Explorer.

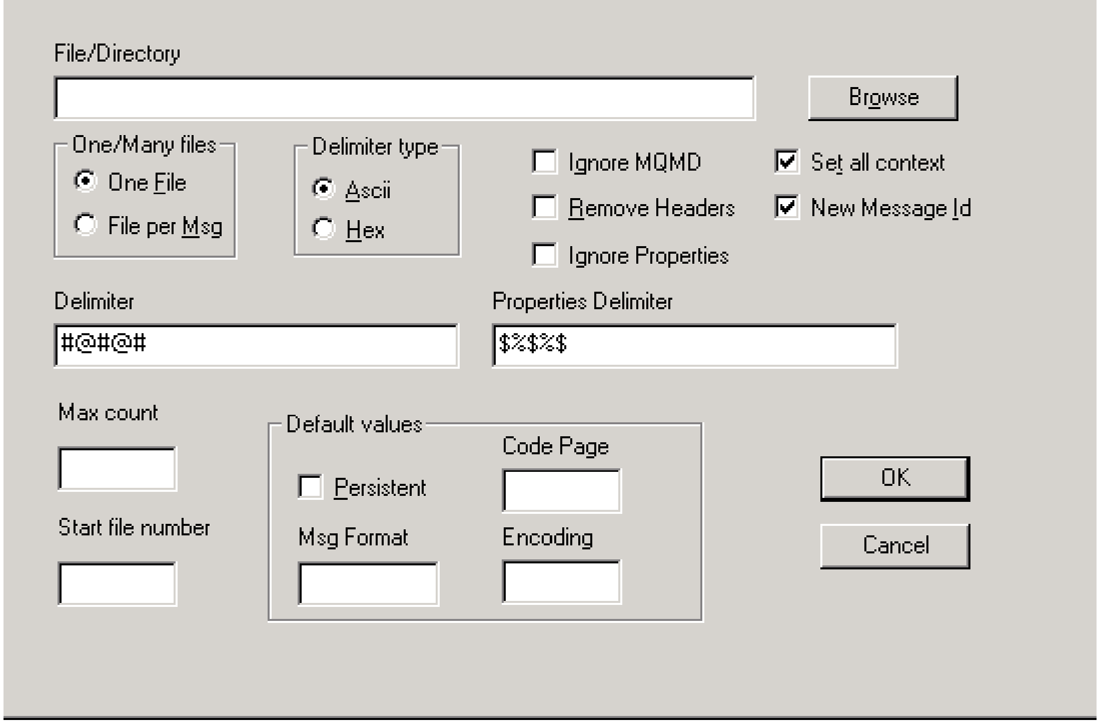
- Open the directory C:\Impact2011\MovingMsgsLab\IH03 Lab in Windows Explorer (if it is not already open), using the shortcut on the desktop

- Double click on rfhtuil.exe.

You will notice the fields match the previous instance but the window heading is RfhUtil V7.0.2 instead of Queue2A.

- Populate Queue Manager Name with QueueQM_2 and Queue Name with Queue2B. Click on BrowseQ and note the Queue Depth field is 0 and the message on the bottom says “No messages in queue.”

- Click on the “LoadQ” button to write messages from our file to Queue2B. The following dialog will be displayed:



File/Directory

One/Many files: ● One File ○ File per Msg

Delimiter type: ● Ascii ○ Hex

□ Ignore MQMD
□ Remove Headers
□ Ignore Properties
☑ Set all context
☑ New Message Id

Delimiter: #@#@#

Properties Delimiter: $%$%$

Max count

Start file number

Default values: □ Persistent

Code Page

Msg Format

Encoding

O

- Once again, the fully qualified file name must be provided. Since we are choosing the file that contains our 15 messages, populate this field with C:\Impact2011\MovingMsgsLab\IH03 Lab\IH03LabFile2. We will leave defaults for the rest of the values.

- Click OK and note that the Queue depth filed of Queue2B should be 15 – the same as Queue2A

- Now that we have made a copy of our messages by unloading Queue2A to a file and loading the file to Queue2B, we will purge Queue2A. Go to the correct RFHUtil Window and verify that the queue name is Queue2A.

- Choose the “PurgeQ” button which will attempt to remove all messages in the current queue. A warning message will be displayed and you will choose “Yes” to purge the queue. The Queue depth of Queue2A is now 0.

This completes the first part of this exercise. If you would like to explore this SupportPac in more detail, please continue onto the optional section.

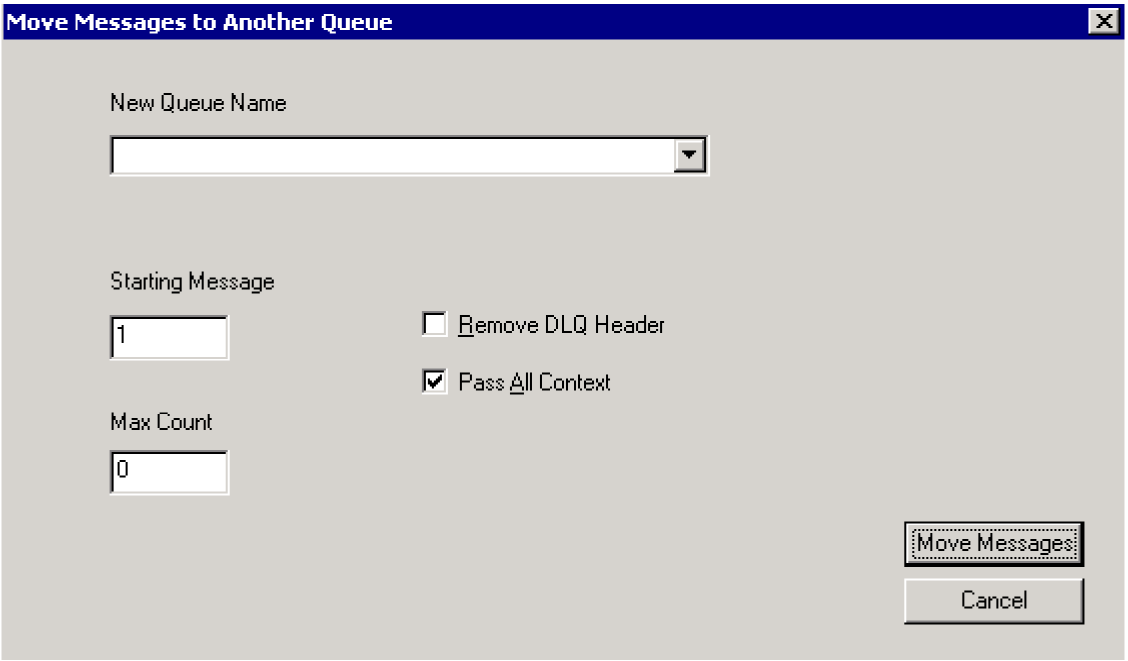
OPTIONAL:

3.2.4 Move Messages between queues

Just as with MO03, this SupportPac is capable of moving messages from one queue to another as well as moving a selection of messages from one queue to another. Both queues must reside on the same queue manager. Currently, Queue2A has a depth of 0 and Queue2B has a depth of 15.

We will start by moving the first 3 messages from Queue2B to Queue2A.

- Be sure that the main Window of RFHUtil is pointing to Queue2B. When you click the "MoveQ" button, a new dialog box will appear:



Move Messages to Another Queue

New Queue Name:

Starting Message: 1

□ Remove DLQ Header

☑ Pass All Context

Max Count: 0

- Once this dialog appears, use the drop down to select the New Queue Name to be Queue2A. Also change the Starting Message to be 2 and the MaxCount to be 3.

In V7 of Websphere MQ, all of the message properties of the message are automatically copied. There are several things to consider when performing this operation such as messages being added to or removed from the queue by an application. More details on these considerations can be found on page 15 of the documentation.

- Choose "Move Messages" and note that the depth of Queue2B has decremented by 3 to be 12. Switch over to the window that has Queue2A in the Main tab, choose Browse Q and note that the Queue depth changes from 0 to 3.

- Now, we will move the rest of the messages from Queue2B to Queue2A.

- Go back to the RFHUtil window that has Queue2B on the Main tab. Once again click the MoveQ button but this time after setting the New Queue Name to be Queue2A, leave the rest of the options as default. When Max Count is 0, all of the messages on the queue will be moved.

- Choose "Move Messages" and verify that Queue2B now has a depth of 0. Switch over to the window that has Queue2A in the Main tab, choose Browse Q and note that the Queue depth changes from 3 to 15.

3.2.5 Clear the messages

The “Clear Data” button will delete the contents of the data buffer only. This is useful if a message has been published and a different type of pub/sub message is to be sent next with no user data. Using this button will not affect the MQMD or RFH header information.

- Before performing this step of the exercise, copy the messages from Queue2A to Queue2B using what you have learned in the first part of the IH03 exercise OR the qload utility. This will provide a basis of comparison for the next step. Once you have completed this and verified that Queue2A and Queue2B are the same, you may continue.

- Be sure that the main Window of RFHUtil is pointing to Queue2B. Also verify that there is data in the Data Tab. If there isn’t click “Browse Q”

- Click the “Clear Data” button,

- You should note that the Data Tab is now blank, the Data Size has changed from 16 to 0 but the Current Depth of the Queue has not changed. Compare the first message of Queue2A and Queue2B and note that the RFH and MQMD tabs are the same but the first message on Queue2A will have data in it.

- Click Browse Q to reset the fields. Then, choose “Clear All.” Not only will the data buffer and RFH folders be cleared, but the MQMD and RFH header fields will be initialized. Compare the first message of Queue2A to Queue2B and notice that the messages in Queue2B not only have their data cleared but also the values in the MQMD will be different. The RFH fields are not populated in this exercise, but if they were they would also be different.

- Close all your open windows.

This concludes the Moving Messages between queues Lab.

Part 4: Troubleshooting

The purpose of this section of the lab is to demonstrate how to trace API calls to WebSphere MQ and how to browse and analyze messages in queues. You will learn how set up API tracing to intercept API calls made from WebSphere MQ client application to a WebSphere MQ queue manager.. Links have been provided to each SupportPac page to allow you to download the SupportPac as well as the documentation. For more details and examples refer to this documentation

MA0W WMQ API Trace

This SupportPac provides the functionality to trace API calls.

http://www-01.ibm.com/support/docview.wss?rs=171&uid=swg24010343

The following table gives you an overview of this utility:

MA0W SupportPac (WMQ API Trace utility)
Command line
Development, Testing, Problem determination
Quick and easy way to trace API calls
Debugging
Full insight to issued API calls.
Function calls can easily be traced and traced data can easily be formatted to enhance readability.
Category 2 SupportPac

4.1 MA0W WMQ API Trace

This lab is based on the SupportPac documentation for this utility. There are examples beyond what is in this lab to help further your knowledge of this utility.

The parameters used for this lab will be described in this section. For more parameter details, please refer to the SupportPac documentation – Chapter 4. Customization

The objective of API trace utility is to provide method to trace all MQ API calls including parameters and process results in easy way - this is achieved by using the exit interface provided by WebSphere MQ.

In addition to API exit, exit interface in sending and receiving message data via MQ channels has become available.

In this lab we will concentrate on configuration settings aimed at functional test, problem determination and will walk through the following scenarios:

- Trace specific MQ process
- Trace particular active MQ process during specific time frame
- Trace all MQ processes during specific time frame

To reduce lab preparation time the following steps have been already performed:

- Create the Lab queue manager: TroubleshootingQM
- Created the test queue: PD_QUEUE_1
- Define API exit modules (for details please refer to MA0W documentation)
- Altered the channels to point to API exit used during the lab, channels involved:

  SYSTEM.DEF.SVRCONN

  SYSTEM.DEF.CLNTCONN
- Altered the parameter file located in:

  C:\IMPACT2011\PDLab\tracedata\iselmqapiexit.def

  REMINDER: File name “iselmqapiexit.def” is case sensitive.

4.1.1 Trace specific MQ process

As the target processes to be traced can often vary, it is recommended to exploit “ExecFlag=env” parameter setting in parameter file. When this parameter is set, trace function is activated for processes running with non-null MA0W environment value.

Example in command prompt on Windows

set MA0W=y (run MQ application)

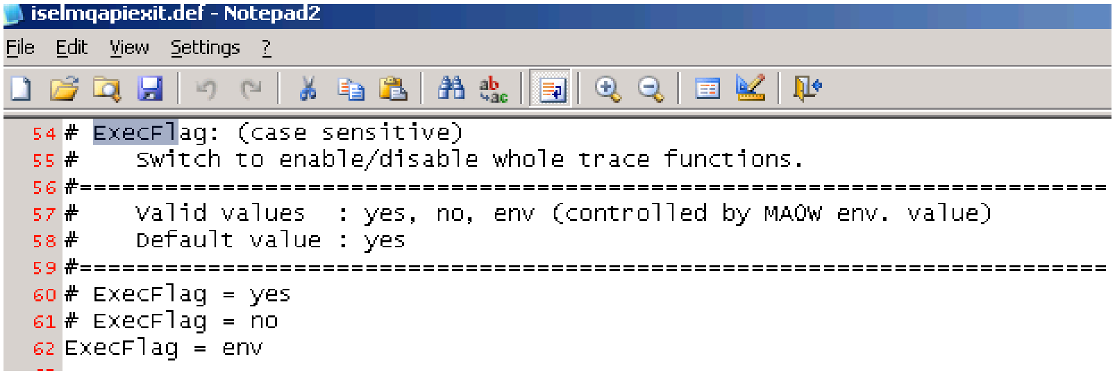
- Open a new command prompt using the shortcut on the desktop

- Stop the queue manager issuing “endmqm TroubleshootingQM”, wait for queue manager to end normally:

```
> endmqm TroubleshootingQM
```

- Open the C:\IMPACT2011\PDLab\tracedata directory in Windows Explorer using the shortcut on the desktop.

- Check that “ExecFlag=env” is set in “iselmqapiexit.def” file:

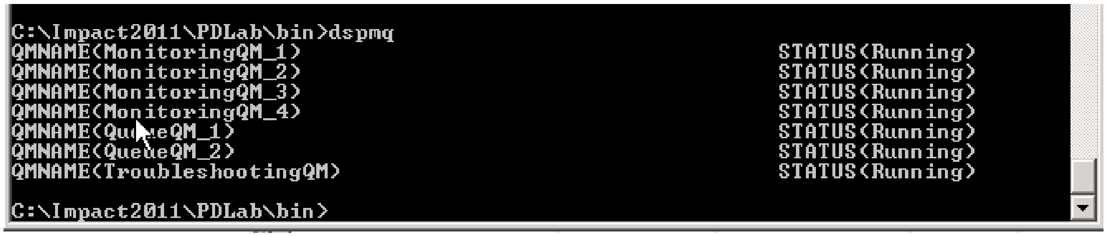


```
# ExecFlag: (case sensitive)
#    Switch to enable/disable whole trace functions.
#==========================================================================
#    Valid values  : yes, no, env (controlled by MA0W env. value)
#    Default value : yes
#==========================================================================
# ExecFlag = yes
# ExecFlag = no
ExecFlag = env
```

- After making the necessary modifications to this file, save and close it.

- Go to the command prompt

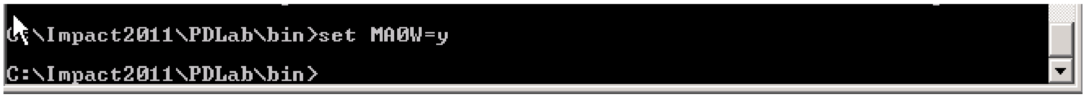
- Start queue manager issuing “strmqm TroubleshootingQM” and Check if is running:



```
C:\Impact2011\PDLab\bin>dspmq
QMNAME(MonitoringQM_1)                                    STATUS(Running)
QMNAME(MonitoringQM_2)                                    STATUS(Running)
QMNAME(MonitoringQM_3)                                    STATUS(Running)
QMNAME(MonitoringQM_4)                                    STATUS(Running)
QMNAME(QueueQM_1)                                         STATUS(Running)
QMNAME(QueueQM_2)                                         STATUS(Running)
QMNAME(TroubleshootingQM)                                 STATUS(Running)

C:\Impact2011\PDLab\bin>
```

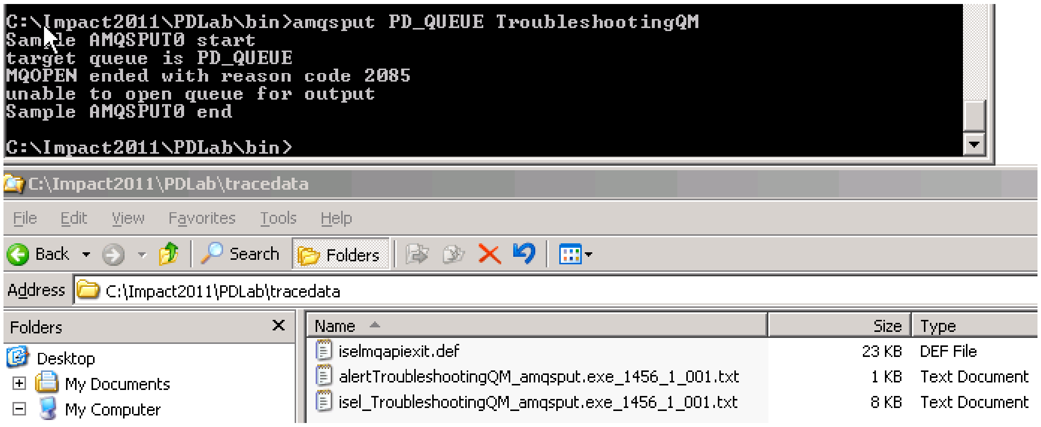
- Open a new command prompt and change directory to C:\Impact 2011\PDLab\bin.

- Set environment variable “set MA0W=y”



```
C:\Impact2011\PDLab\bin>set MA0W=y

C:\Impact2011\PDLab\bin>
```

- To create a trace file invoke “amqsput PD_QUEUE TroubleshootingQM” and observe the directory: “C:\IMPACT2011\PDLab\tracedata” in Windows Explorer



```
C:\Impact2011\PDLab\bin>amqsput PD_QUEUE TroubleshootingQM
Sample AMQSPUT0 start
target queue is PD_QUEUE
MQOPEN ended with reason code 2085
unable to open queue for output
Sample AMQSPUT0 end

C:\Impact2011\PDLab\bin>
```

C:\Impact2011\PDLab\tracedata

Address C:\Impact2011\PDLab\tracedata

Name	Size	Type
iselmqapiexit.def	23 KB	DEF File
alertTroubleshootingQM_amqsput.exe_1456_1_001.txt	1 KB	Text Document
isel_TroubleshootingQM_amqsput.exe_1456_1_001.txt	8 KB	Text Document

Outcome:

From the output you can see that some mistake has been done whilst issuing the command “amqsput”. During runtime the PID of “amqsput” was 1456 (the PID will be different for you). By using this tracing method with the environment variable, the process / application to be traced could be dynamically chosen without having to trace all processes connecting to that queue manager

Analysis:

Try checking the newly created trace files in the directory “C:\IMPACT2011\PDLab\tracedata” and try to identify the mistake and correct it.

Hint:

You might want to check either using “runmqsc” or “WMQ Explorer” to look at the objects belonging to the specific queue manager. Be reminded that if you start “runmqsc” from the same shell as before, you will also trace that application, as the environment variable “set MA0W=y” is still set!

(solution in Appendix 1)

4.1.2 Trace particular active MQ process during specific time frame

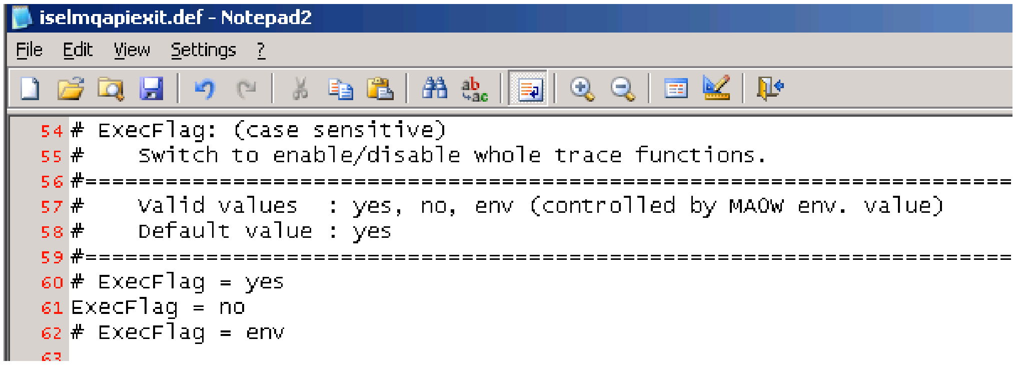
This tracing method is used when a specific application (with respective known PID) needs to be traced, even if general tracing has been deactivated during queue manager start.

- Open a new command prompt

- Stop queue manager issuing “endmqm TroubleshootingQM”, wait for queue manager to end normally:

```
> endmqm TroubleshootingQM
```

- Check that “ExecFlag=no” is set in “iselmqapiexit.def” file:

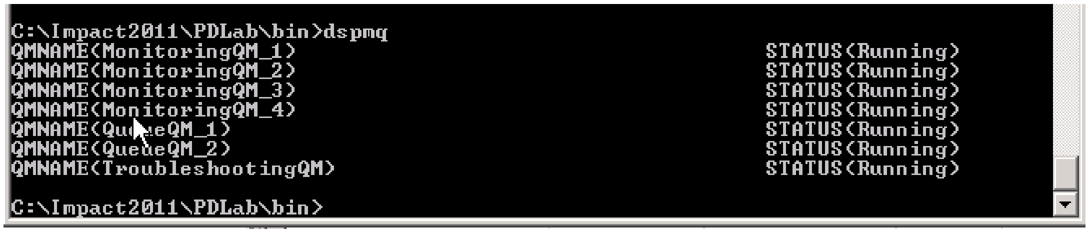


```
# ExecFlag: (case sensitive)
#    Switch to enable/disable whole trace functions.
#=====================================================================
#    Valid values  : yes, no, env (controlled by MAOW env. value)
#    Default value : yes
#=====================================================================
# ExecFlag = yes
ExecFlag = no
# ExecFlag = env
```

- Save and close this file.
- Open a new command prompt
- Start the queue manager issuing “strmqm TroubleshootingQM”:

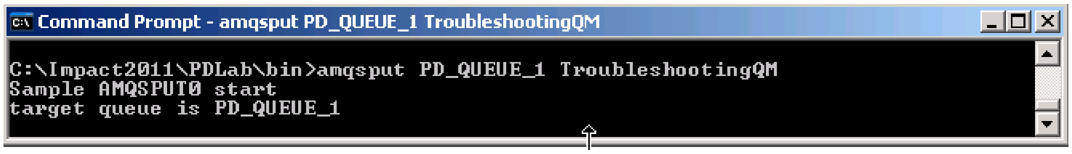
```
> strmqm TroubleshootingQM
```

- Check the queue manager is running with the “dspmq” command or through the WebSphere MQ Explorer:



```
C:\Impact2011\PDLab\bin>dspmq
QMNAME(MonitoringQM_1)                                    STATUS(Running)
QMNAME(MonitoringQM_2)                                    STATUS(Running)
QMNAME(MonitoringQM_3)                                    STATUS(Running)
QMNAME(MonitoringQM_4)                                    STATUS(Running)
QMNAME(QueueQM_1)                                         STATUS(Running)
QMNAME(QueueQM_2)                                         STATUS(Running)
QMNAME(TroubleshootingQM)                                 STATUS(Running)

C:\Impact2011\PDLab\bin>
```

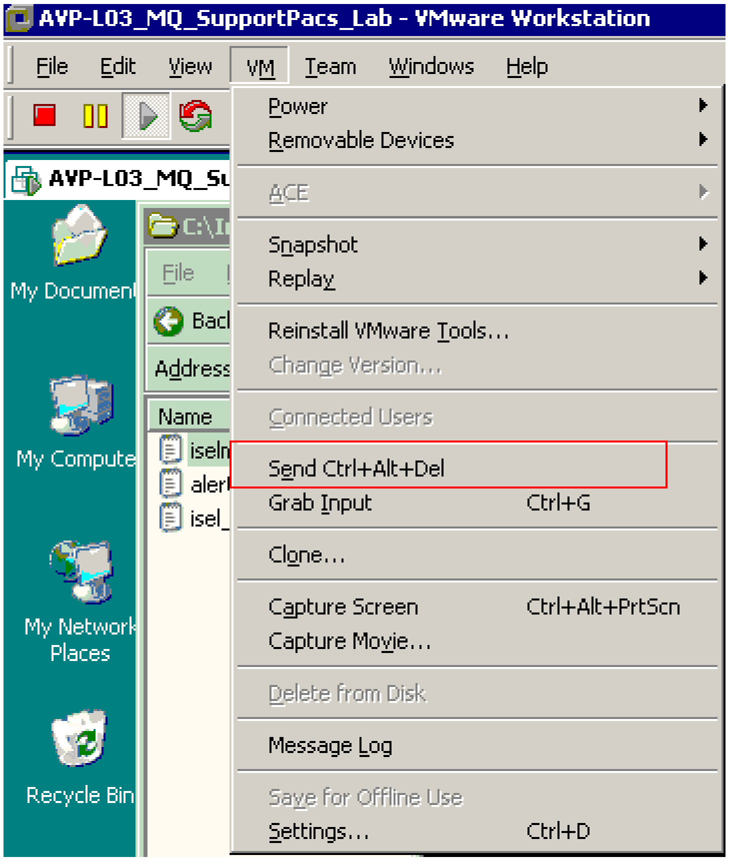
- Invoke “amqsput PD_QUEUE_1 TroubleshootingQM” and observe the directory: “C:\IMPACT2011\PDLab\tracedata”



```
C:\Impact2011\PDLab\bin>amqsput PD_QUEUE_1 TroubleshootingQM
Sample AMQSPUT0 start
target queue is PD_QUEUE_1
```

NOTE: At this stage no trace files are created, as we explicitly requested tracing to be deactivated

- To identify the process and correspondent PID to trace, open the Windows Task Manger. To open the Windows Task Manager select “VM → Send ctrl+alt+Del and then select “Task Manager”



- look for “amqsput.exe” in the task list. (to display the PID go to View→Select Columns and select PID)

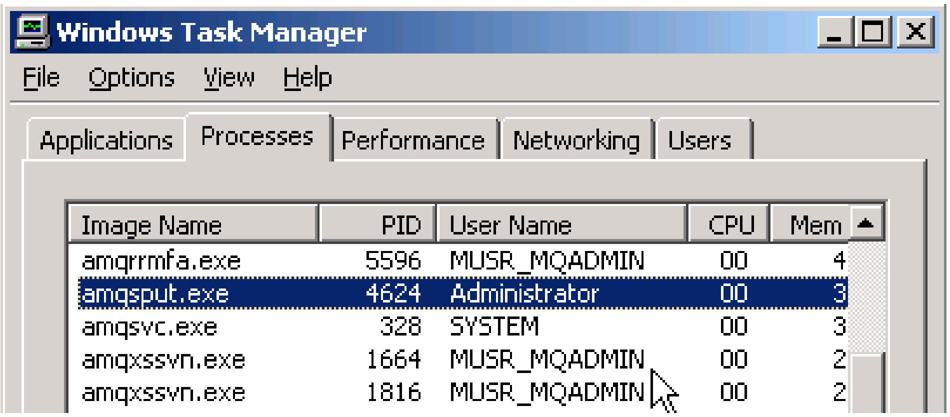
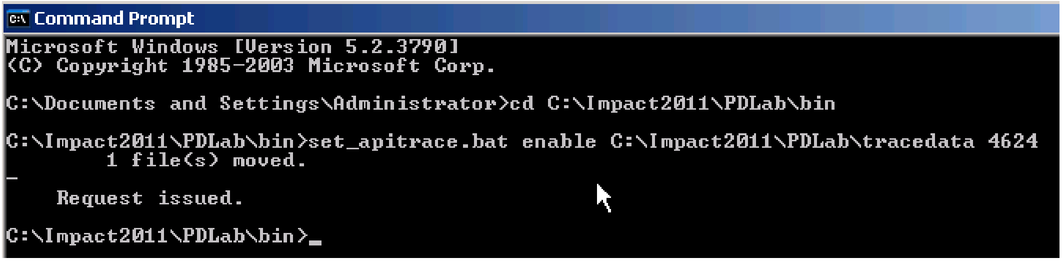


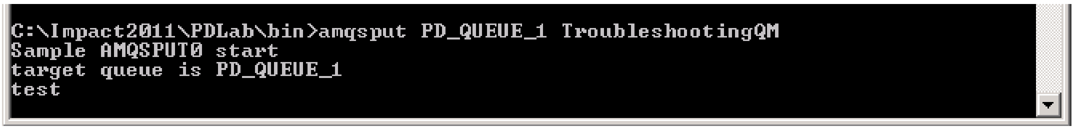
Image Name	PID	User Name	CPU	Mem
amqrrmfa.exe	5596	MUSR_MQADMIN	00	4
amqsput.exe	4624	Administrator	00	3
amqsvc.exe	328	SYSTEM	00	3
amqxssvn.exe	1664	MUSR_MQADMIN	00	2
amqxssvn.exe	1816	MUSR_MQADMIN	00	2

- Open a new command prompt, and activate tracing for process and correspondent PID by issuing the following commands (replacing 4624 with the pid of your amqsput process) :



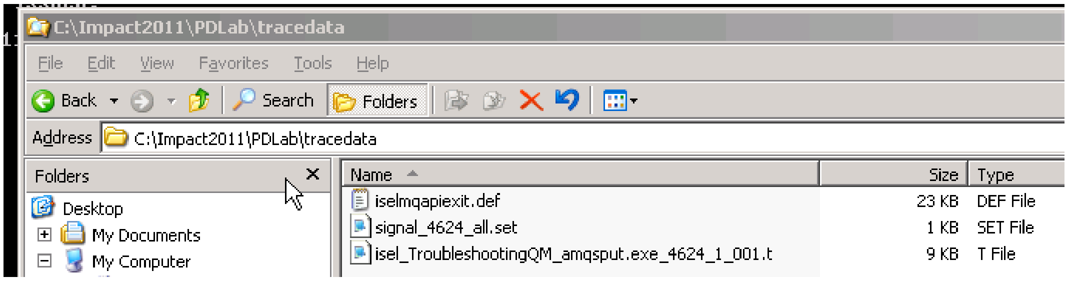
```
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\Documents and Settings\Administrator>cd C:\Impact2011\PDLab\bin

C:\Impact2011\PDLab\bin>set_apitrace.bat enable C:\Impact2011\PDLab\tracedata 4624
        1 file(s) moved.
_
    Request issued.

C:\Impact2011\PDLab\bin>_
```

- Go back to the command window which is still running the amqsput command. The trace file gets created when the first API call is issued by the traced process, so let's put a message into the queue "PD_QUEUE_1" by entering "test" into the amqsput session and pressing "ENTER"



```
C:\Impact2011\PDLab\bin>amqsput PD_QUEUE_1 TroubleshootingQM
Sample AMQSPUT0 start
target queue is PD_QUEUE_1
test
```

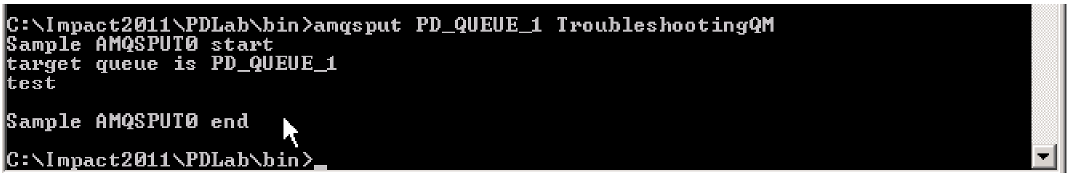
- Observe a trace file getting created for our process and correspondent PID, whilst the trace file is being written to it has the suffix ".t"



Address C:\Impact2011\PDLab\tracedata

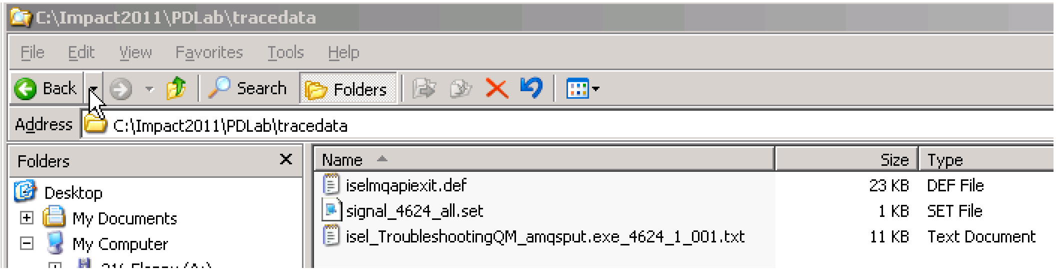
Name	Size	Type
iselmqapiexit.def	23 KB	DEF File
signal_4624_all.set	1 KB	SET File
isel_TroubleshootingQM_amqsput.exe_4624_1_001.t	9 KB	T File

i. We end the process / application "amqsput" by hitting "ENTER"…



```
C:\Impact2011\PDLab\bin>amqsput PD_QUEUE_1 TroubleshootingQM
Sample AMQSPUT0 start
target queue is PD_QUEUE_1
test

Sample AMQSPUT0 end

C:\Impact2011\PDLab\bin>_
```

ii. … and observe that the trace file is being finalized. Now it has the suffix "txt". This indicates that the trace process has been terminated.



Address C:\Impact2011\PDLab\tracedata

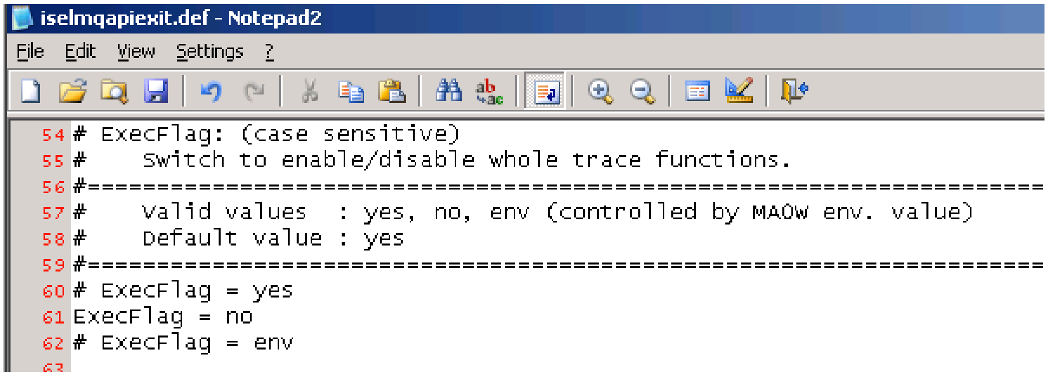
Name	Size	Type
iselmqapiexit.def	23 KB	DEF File
signal_4624_all.set	1 KB	SET File
isel_TroubleshootingQM_amqsput.exe_4624_1_001.txt	11 KB	Text Document

i. You can now analyze the trace file. Try finding out where our test message has been placed and what API calls have been invocated.

(solution in Appendix 1)

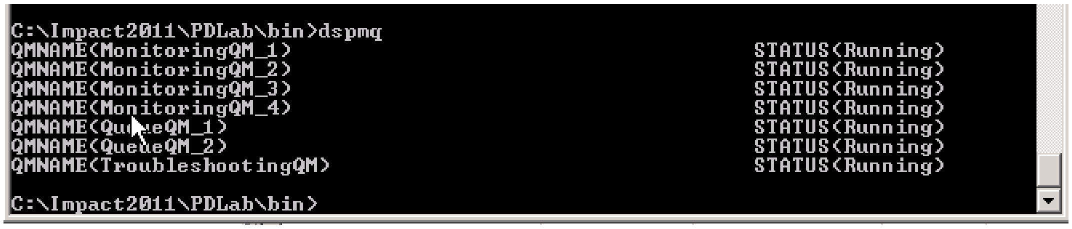
4.1.3 Trace all MQ processes during specific time frame

- Open a command prompt if you don’t already have one open
- Open Windows Explorer in the directory C:\Impact2011\PDLab\TraceData. There is a shortcut to the PDLab directory on the desktop. “ExecFlag=no” should be set in “iselmqapiexit.def” file:



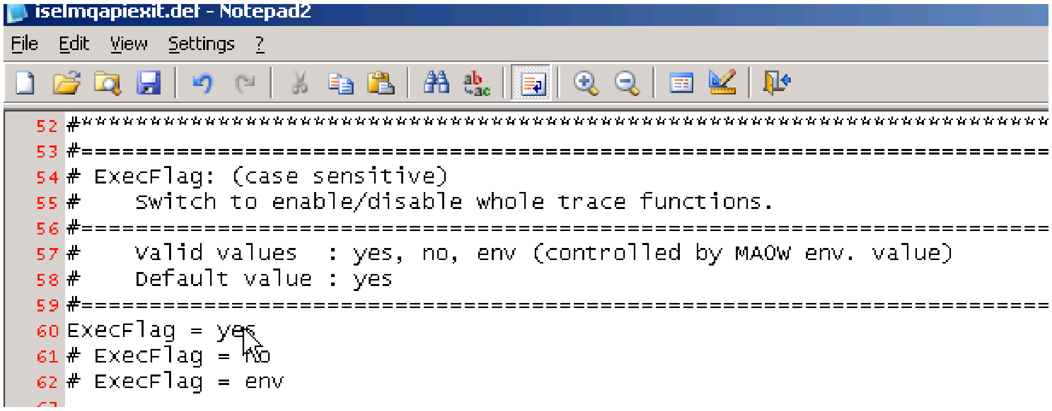
```
# ExecFlag: (case sensitive)
#    Switch to enable/disable whole trace functions.
#======================================================================
#    Valid values  : yes, no, env (controlled by MAOW env. value)
#    Default value : yes
#======================================================================
# ExecFlag = yes
ExecFlag = no
# ExecFlag = env
```

- Open a new command prompt
- Check the queue manager TroubleshootingQM is running using the dspmq command (start it using “strmqm TroubleshootingQM” or the Websphere MQ Explorer if it isn’t:



```
C:\Impact2011\PDLab\bin>dspmq
QMNAME(MonitoringQM_1)                                    STATUS(Running)
QMNAME(MonitoringQM_2)                                    STATUS(Running)
QMNAME(MonitoringQM_3)                                    STATUS(Running)
QMNAME(MonitoringQM_4)                                    STATUS(Running)
QMNAME(QueueQM_1)                                         STATUS(Running)
QMNAME(QueueQM_2)                                         STATUS(Running)
QMNAME(TroubleshootingQM)                                 STATUS(Running)

C:\Impact2011\PDLab\bin>
```

- Set “ExecFlag=yes” in “iselmqapiexit.def” file:

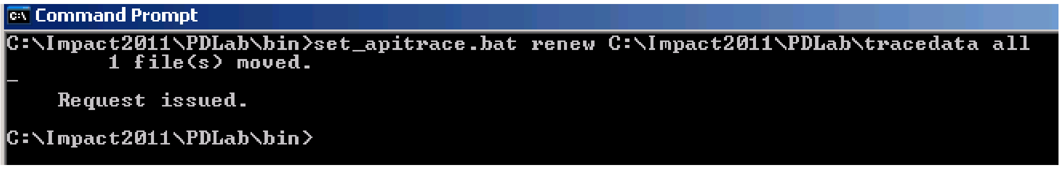


```
#**********************************************************************
#======================================================================
# ExecFlag: (case sensitive)
#    Switch to enable/disable whole trace functions.
#======================================================================
#    Valid values  : yes, no, env (controlled by MAOW env. value)
#    Default value : yes
#======================================================================
ExecFlag = yes
# ExecFlag = no
# ExecFlag = env
```

- Open a command prompt and change directory to C:\Impact2011\PDLab\bin:

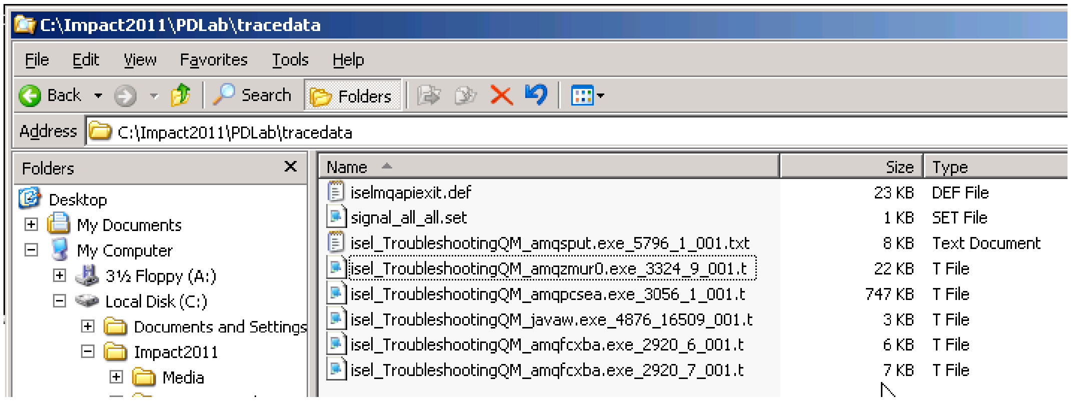
> cd c:\Impact2011\PDLab\bin

- activate tracing by issuing the following command:



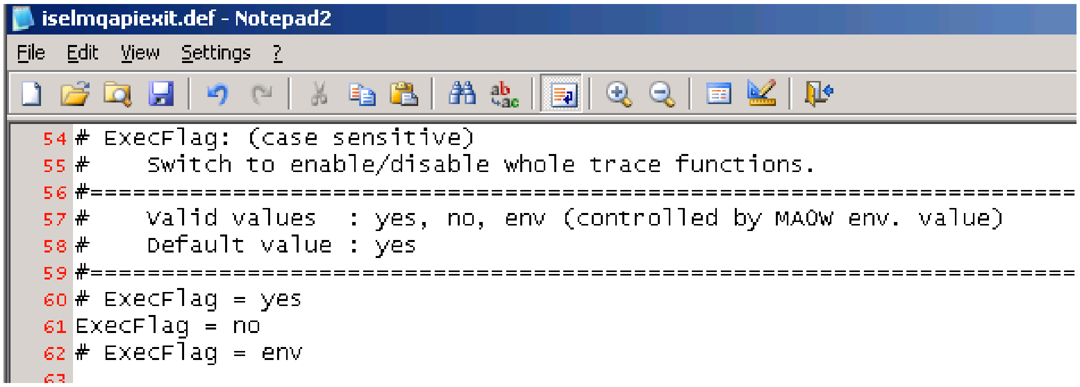
```
C:\Impact2011\PDLab\bin>set_apitrace.bat renew C:\Impact2011\PDLab\tracedata all
        1 file(s) moved.
_
    Request issued.

C:\Impact2011\PDLab\bin>
```

- observe trace files created by **ALL** processes connected to the queue manager



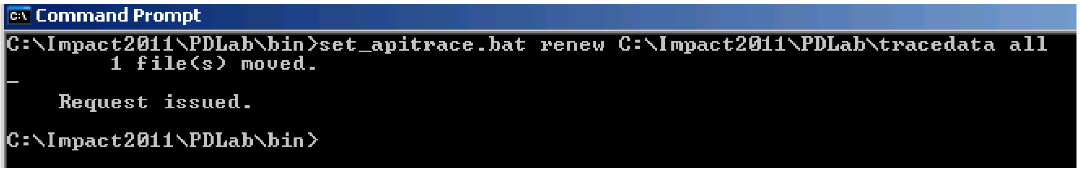
Name	Size	Type
iselmqapiexit.def	23 KB	DEF File
signal_all_all.set	1 KB	SET File
isel_TroubleshootingQM_amqsput.exe_5796_1_001.txt	8 KB	Text Document
isel_TroubleshootingQM_amqzmur0.exe_3324_9_001.t	22 KB	T File
isel_TroubleshootingQM_amqpcsea.exe_3056_1_001.t	747 KB	T File
isel_TroubleshootingQM_javaw.exe_4876_16509_001.t	3 KB	T File
isel_TroubleshootingQM_amqfcxba.exe_2920_6_001.t	6 KB	T File
isel_TroubleshootingQM_amqfcxba.exe_2920_7_001.t	7 KB	T File

- to deactivate tracing after a specific time set “ExecFlag=no” in “iselmqapiexit.def” file:



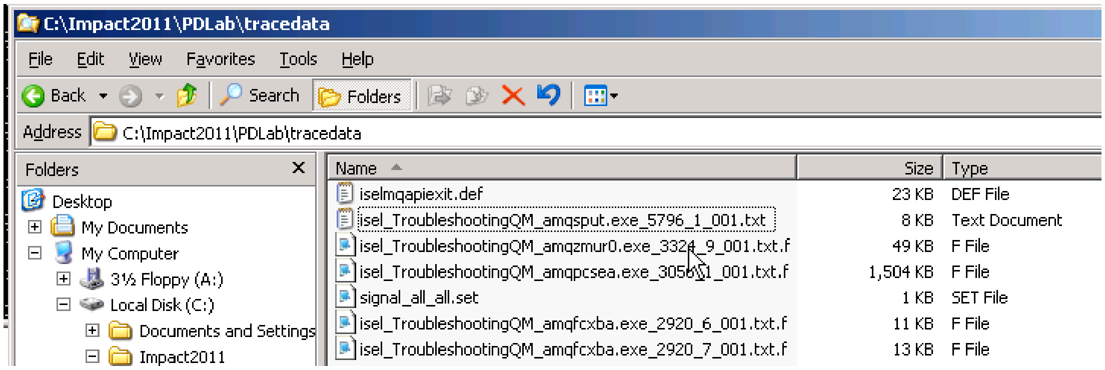
```
# ExecFlag: (case sensitive)
#     Switch to enable/disable whole trace functions.
#=====================================================================
#     Valid values  : yes, no, env (controlled by MAOW env. value)
#     Default value : yes
#=====================================================================
# ExecFlag = yes
ExecFlag = no
# ExecFlag = env
```

- deactivate tracing by issuing the following command:



```
C:\Impact2011\PDLab\bin>set_apitrace.bat renew C:\Impact2011\PDLab\tracedata all
        1 file(s) moved.
_
    Request issued.

C:\Impact2011\PDLab\bin>
```

- observe how trace are finalized by append “.f” to the files. This may take a few minutes to complete.



Name	Size	Type
iselmqapiexit.def	23 KB	DEF File
isel_TroubleshootingQM_amqsput.exe_5796_1_001.txt	8 KB	Text Document
isel_TroubleshootingQM_amqzmur0.exe_3324_9_001.txt.f	49 KB	F File
isel_TroubleshootingQM_amqpcsea.exe_3056_1_001.txt.f	1,504 KB	F File
signal_all_all.set	1 KB	SET File
isel_TroubleshootingQM_amqfcxba.exe_2920_6_001.txt.f	11 KB	F File
isel_TroubleshootingQM_amqfcxba.exe_2920_7_001.txt.f	13 KB	F File

End of Lab

Appendix 1 : Solutions to Troubleshooting lab exercises:

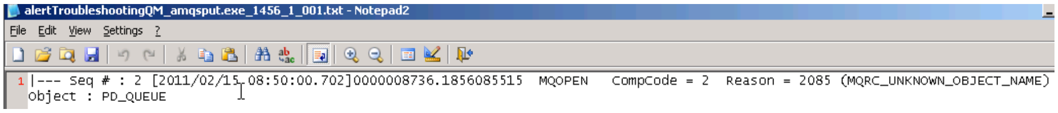
Solution: Trace specific MQ process

When issuing the command "amqsput PD_QUEUE TroubleshootingQM" a wrong queue name was provided:

WRONG: "amqsput PD_QUEUE TroubleshootingQM"

CORRECT: "amqsput PD_QUEUE_1 TroubleshootingQM"

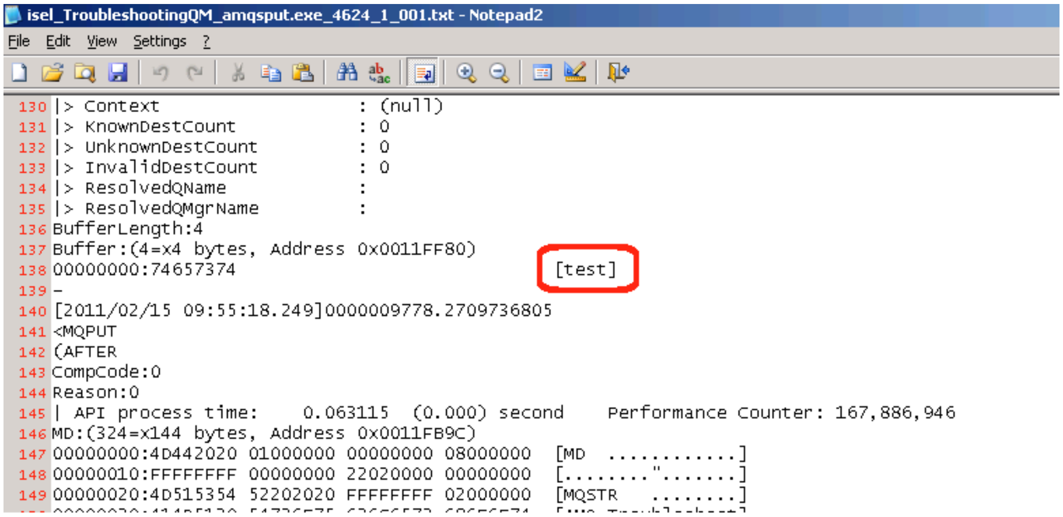
This can very well be seen in both trace files, as the reason code is decoded:


alertTroubleshootingQM_amqsput.exe_1456_1_001.txt - Notepad2
File Edit View Settings ?

```
|--- Seq # : 2 [2011/02/15 08:50:00.702]0000008736.1856085515  MQOPEN   CompCode = 2  Reason = 2085 (MQRC_UNKNOWN_OBJECT_NAME)
Object : PD_QUEUE
```

Solution: Trace particular active MQ process during specific time frame

By search for the string "test" you will find our test message and the commands issued. Look through the trace file and try to understand, what "amqsput" does. The source files of amqsput are available on the system, look for them in the installation directory of WebSphere MQ.


isel_TroubleshootingQM_amqsput.exe_4624_1_001.txt - Notepad2
File Edit View Settings ?

```
|> Context                    : (null)
|> KnownDestCount             : 0
|> UnknownDestCount           : 0
|> InvalidDestCount           : 0
|> ResolvedQName              :
|> ResolvedQMgrName           :
BufferLength:4
Buffer:(4=x4 bytes, Address 0x0011FF80)
00000000:74657374                                   [test]
-
[2011/02/15 09:55:18.249]0000009778.2709736805
<MQPUT
(AFTER
CompCode:0
Reason:0
| API process time:    0.063115  (0.000) second    Performance Counter: 167,886,946
MD:(324=x144 bytes, Address 0x0011FB9C)
00000000:4D442020 01000000 00000000 08000000  [MD  ............]
00000010:FFFFFFFF 00000000 22020000 00000000  [........".......]
00000020:4D515354 52202020 FFFFFFFF 02000000  [MQSTR   ........]
```