IMQ09 - IBM MQ V9 for z/OS Wildfire Workshop

L14 – Introduction to the CICS MQ Trigger Monitors

Version V6.0

Septembery 2018

Wildfire
Technical Hands-On Workshops

IBM WSC Wildfire Team
IBM z Systems

Table of Contents

Exercise Objectives

The objective of this exercise is to gain experience with configuring and securing access to CICS using CICS MQ trigger monitors. In this exercise you will configure the MQ resources (queues and processes) and CICS resources required for using MQ messages to trigger CICS applications. Security resources will also be configured. Once all of the resources have been defined a simple Liberty application will be used to test the configuration changes.

General Exercise Information and Guidelines

- Information required to complete this exercise will be provided on a 'worksheet' prior to the start of this exercise. Refer to this worksheet for which user identity and password are to be used and for other values, for example:

  - ✓ This exercise requires using TSO user *USER1* the localzos system.
  - ✓ The TSO password for this exercise will be provided by the lab instructor.
  - ✓ As a reminder, if a value from your worksheet should be used, the values in the instructions will be in red rather than black.
  - ✓ ***Bold italicized*** text indicates values that need to be entered on a screen.
  - ✓ *Italicized* text indicates values that are constants or names that appear on a screen.
  - ✓ **Bold** text indicates the name of buttons or keyboard keys that need to be pressed.

Part 1 – Introduction to the CICS MQ Trigger Monitor

The CICS MQ trigger monitor is provided by CICS to allow the use of MQ messages for sending requests to and receiving responses from CICS applications. The trigger monitor, as its name implies, actively monitors designated queues for the arrival of messages. When a message does arrive on a designated queue, the trigger monitor invokes, via a CICS transaction, either a CICS or user provided program for processing the message.

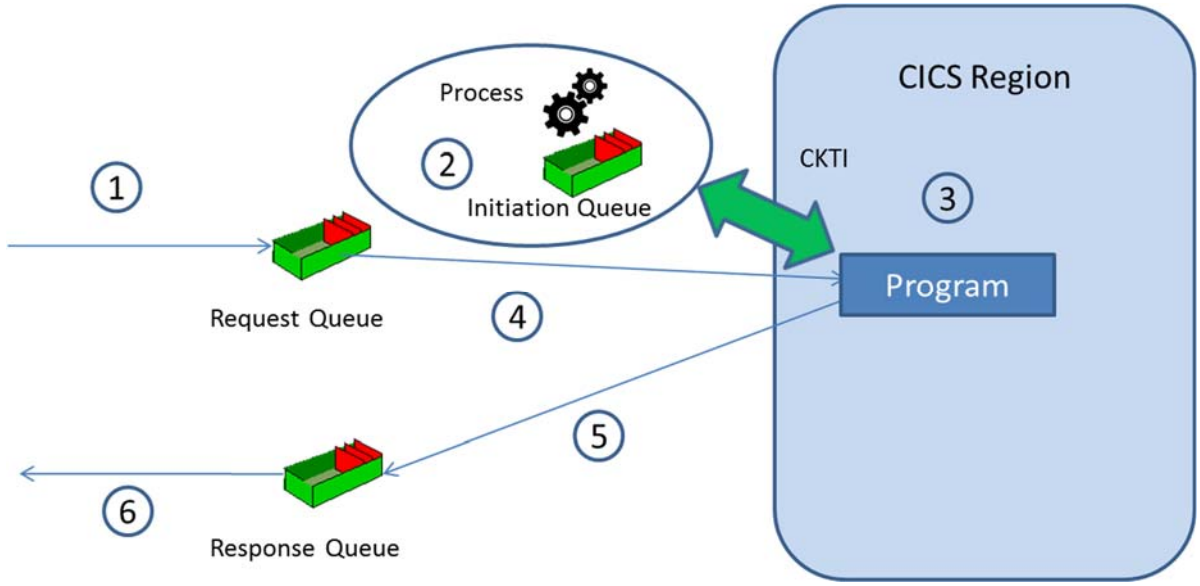
The CICS program started by the trigger monitor has to be MQ aware. This is because this application needs to retrieve the request message from a *request queue* and return the response message in a *response queue*.

The CICS Bridge is a specialized usage of the trigger monitor. The programs started by CICS Bridge are provided by CICS and processes the message in a *request queue* and then links to a user program and pass the contents of the messages either in a CICS COMMAREA or in a CICS container in a channel. The user program processes the contents of the message and returns the results to the CICS Bridge program in a COMMAREA or container in the channel. The CICS Bridge program then puts the response as a message in the *response queue*. The advantage of using the CICS Bridge is that the target user program does not have to be MQ aware and therefore can be invoked from many different types of CICS client applications (e.g. web services, 3270 BMS programs, Java clients, EXCI clients, CTG clients, and so on). An existing user application can easily be accessed using MQ as the transport without requiring any modifications.

In order for a queue to be monitored for either the trigger monitors the queue has to have triggering properties configured, e.g. an *initiation queue* and a *process definition* configured. The former is used to send notification messages or events to CICS and the later identifies the CICS transaction to be invoked when a message arrives as well as certain CICS execution or runtime properties (e.g. security, unit of work, transactionality, etc.). We will see examples of these all of these resources interact in this exercise.

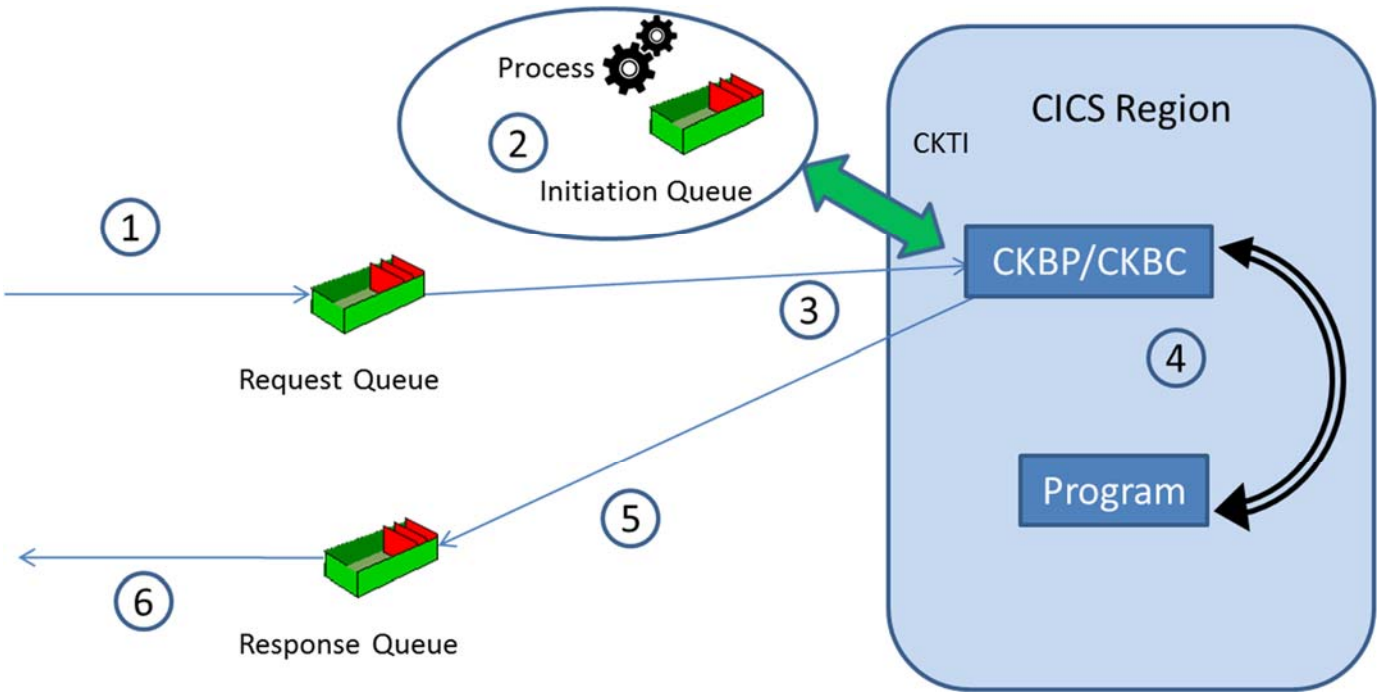
Before we begin, let's explore the trigger monitors by using diagrams and to detail the MQ resources required and the steps involved in sending a request message and receiving a response message from a CICS application.

The diagram below shows the resources and flow of a basic CICS trigger monitor request. A *request queue* is configured along with an associated *initiation queue* and *process definition*. Note that prior to the fist message, a CICS *CKTI* transaction is started and monitors the *initiation queue* for the arrival of messages.



1. A message is placed on the *request queue*.
2. The queue manager notes the arrival of the message on the *request queue* and then uses the information from the *process definition* to determine when to place a notification message on the *initiation queue.* The initiation queue is used for sending a notification event to one or more CICS regions which are waiting for messages to arrive on the initiation queue.
3. The long running CICS *CKTI* task retrieves the notification message from the initiation queue and starts the CICS program associated with the process *definition's Application ID*.
4. The target user program starts and retrieves the message directly from the *request queue.*
5. The target user program places a response message directly in the *response queue*.
6. The response from the CICS application program is obtained by retrieving the message from the *response queue.*

The CICS Bridge Monitor is a special case of triggering where the CICS transaction and MQ aware program are provided by CICS. As before, a *request queue* is configured along with an associated *initiation queue* and a *process definition*. The latter provides the initial CICS transaction, either *CKBR* or *CKBC* along with other information. Again, note that prior to the first message a CICS *CKTI* transaction is started that monitors the *initiation queue* for the arrival of messages.



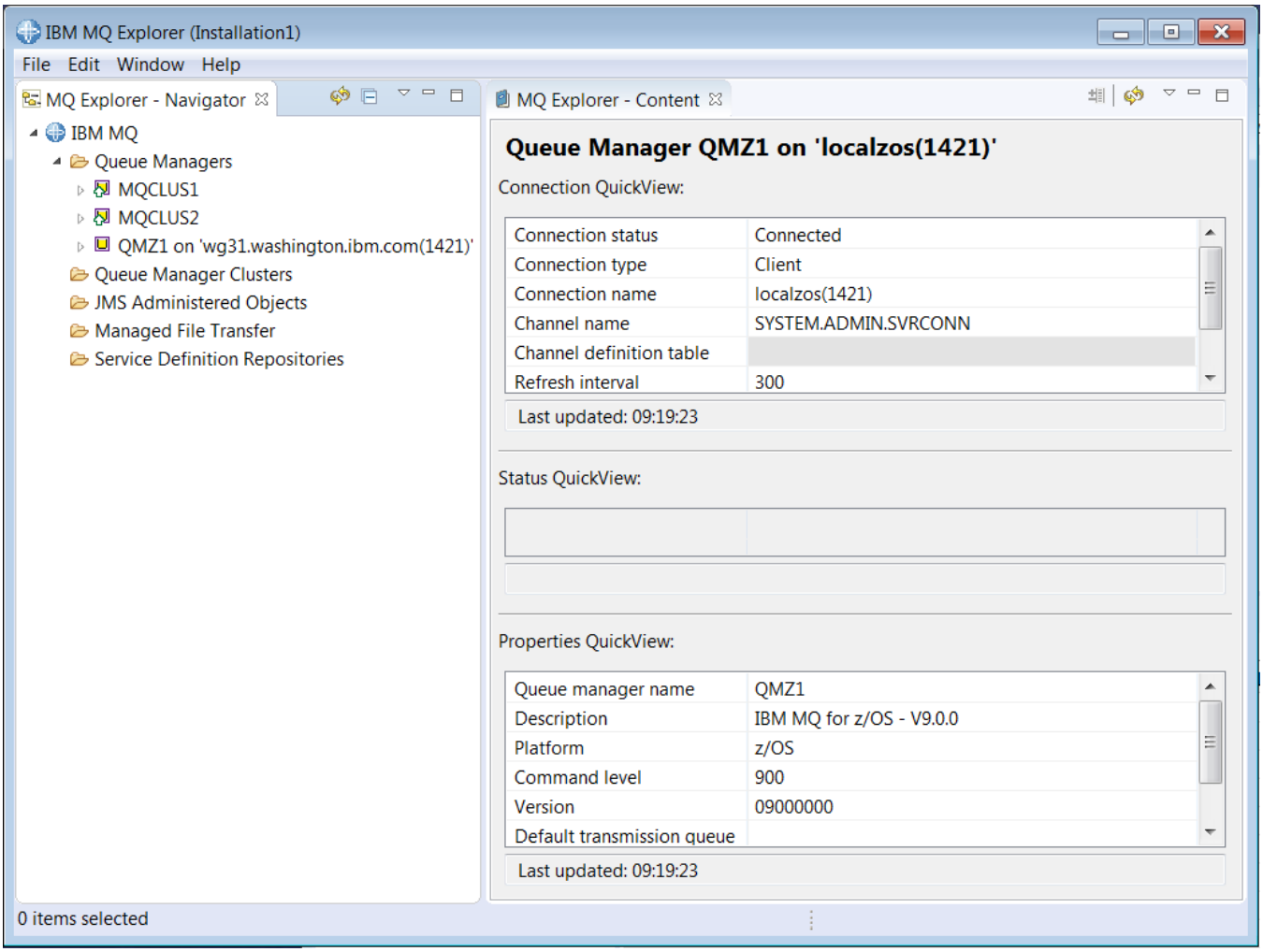
1. A message is placed on the request queue.
2. The queue manager notes the arrival of the message on the *request queue* and then uses the information from the *process definition* to determine when to place a notification message on the *initiation queue.* The initiation queue is used for sending a notification event to one or more CICS regions which are waiting for messages to arrive on the initiation queue.
3. The long running CICS *CKTI* task retrieves the notification message from the *initiation queue* and starts the CICS Bridge monitor program, per the *Application ID* (*CKBR*) in the *process definition*. Information derived from the MQCIH header (e.g. transaction identifier (CKBC or CKBP) and program name, etc.) is used to do a CICS link to the target program. The link request passes the message contents in either a CICS common area (CKBP) or in a single container in a channel (CKBC). The target program executes and returns the response using the same means (COMMREA or CONTAINER) in which the request arrived.
4. The CICS bridge program places the response from the user application in the *response queue*.
5. The response from the CICS application program is obtained by retrieving the message from the *response queue.*

Part 2 – Configuring MQ Trigger Resources

In this part of the exercise the steps required to define the MQ resources for triggering to CICS will be performed using the MQ Explorer.

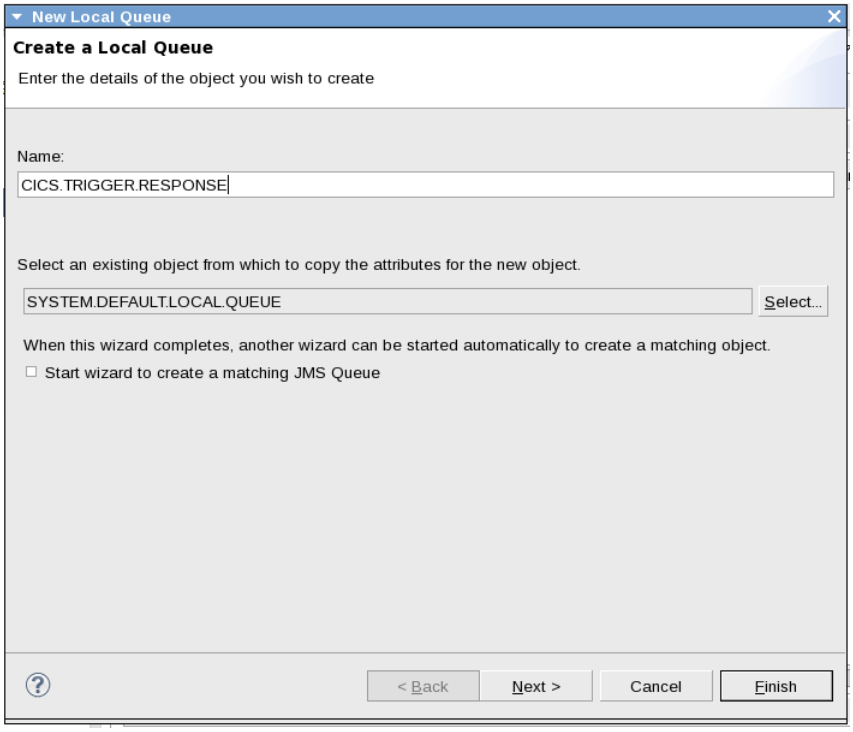
____1. Start MQ Explorer if not already active and connect to queue manager *QMZ1* on host ***wg31.washington.ibm.com*** on port ***1421***.

____2. Once connected, select *Queues* to display a list of queues defined in this Queue Manager.

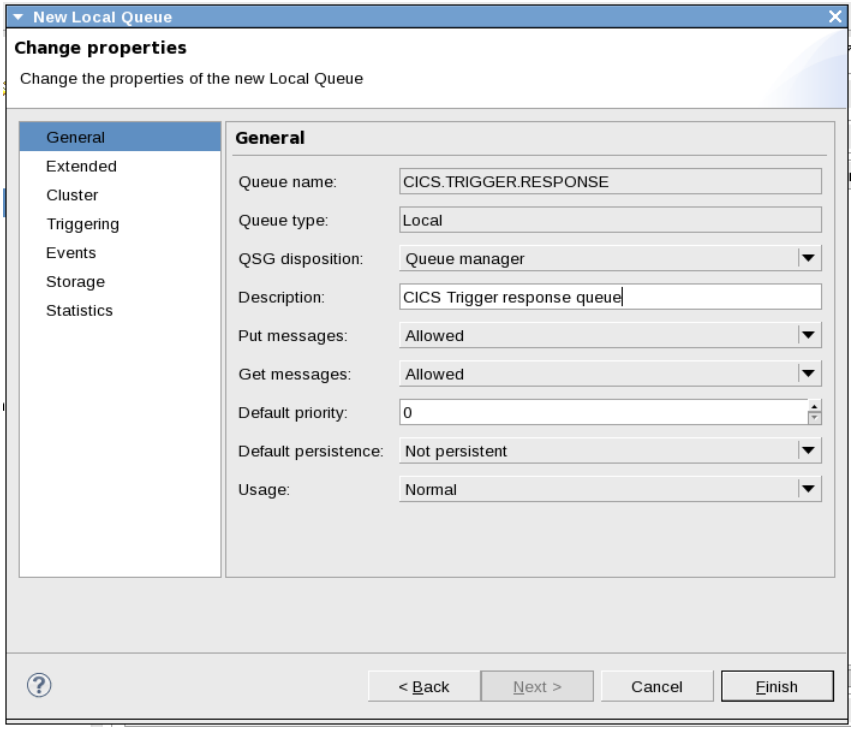


Tech-Tip: A filter of ***CICS.**** was being used in these screen shots to reduce the number of queues displayed.

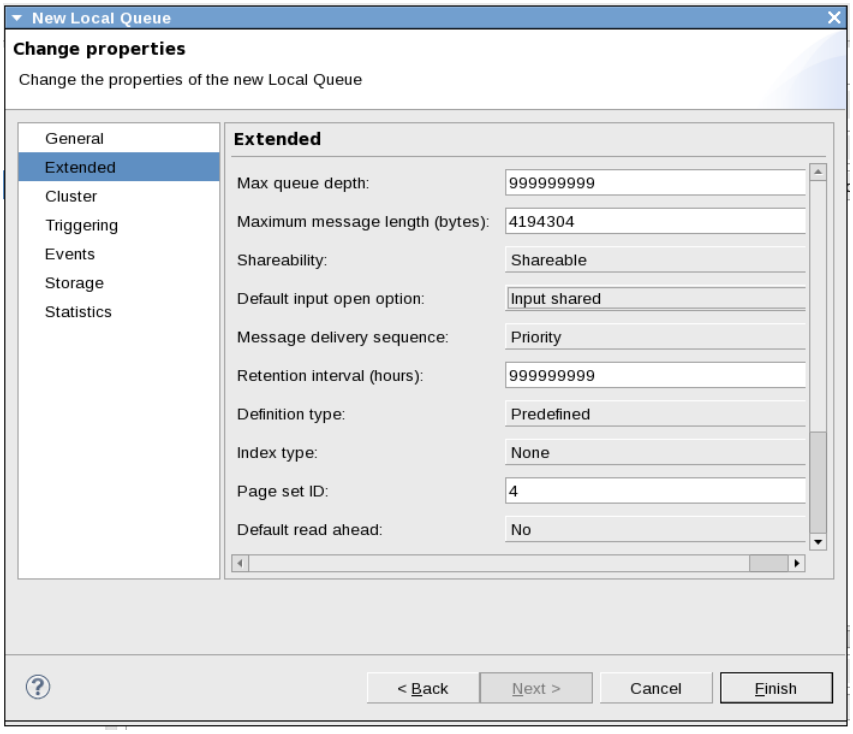
____3. Right mouse button click on *Queues* and select New –> *Local Queue*. This will open a *Create a Local Queue* pane (see below). Enter ***CICS.TRIGGER.RESPONSE*** for the *Name* of the response queue. This queue will be where the response from the CICS program can be found. Click **Next** to continue.



____4. This will open the *Change properties* window (see below). Enter a meaningful description and then select *Extended* on the left hand side of the pane.

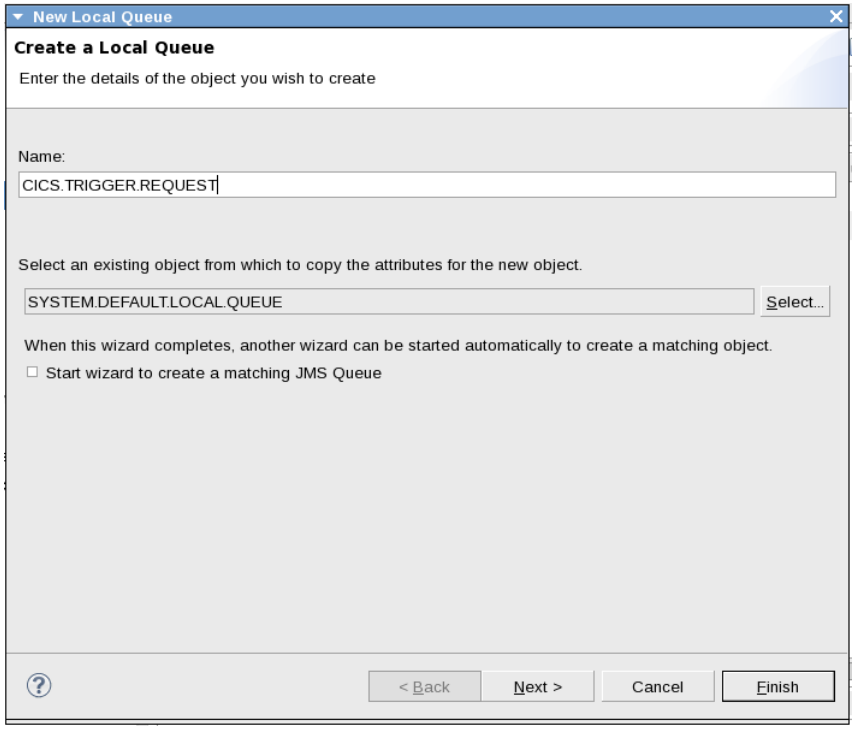


____5. On the *Extended* window use the arrows beside *Shareability* and *Default input open option* properties to make the queue *Sharable* and *Input shared*. Click **Finish** to continue.

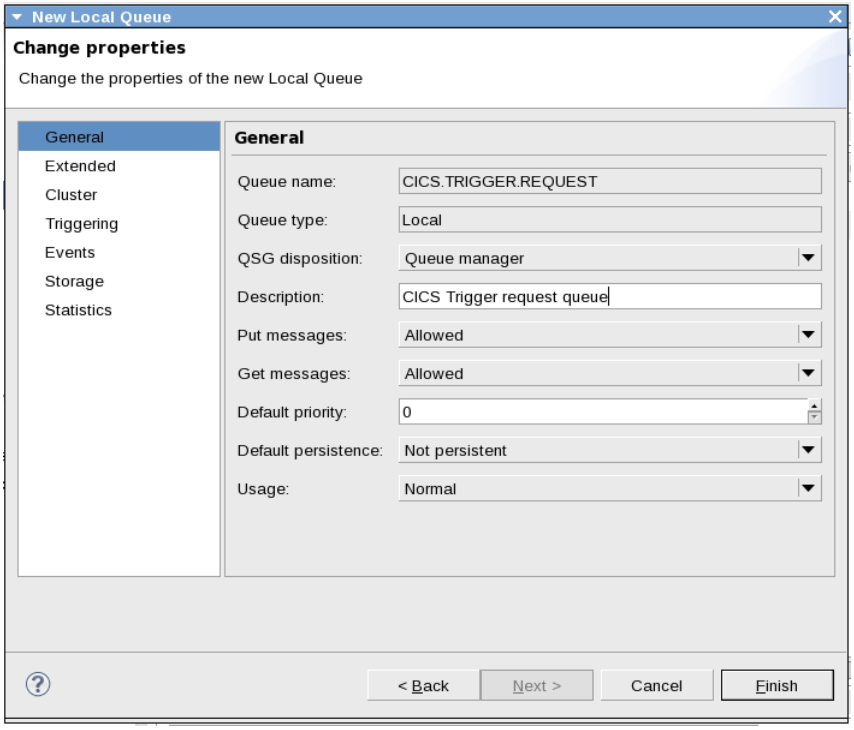


____6. Repeat Step 3 through 5 and create another local queue with the name ***CICS.TRIGGER.INITQ***. An instance of a CICS *CKTI* will be started for this queue and this queue will be monitored for notification or event messages, i.e., a message has arrived on the request queue.

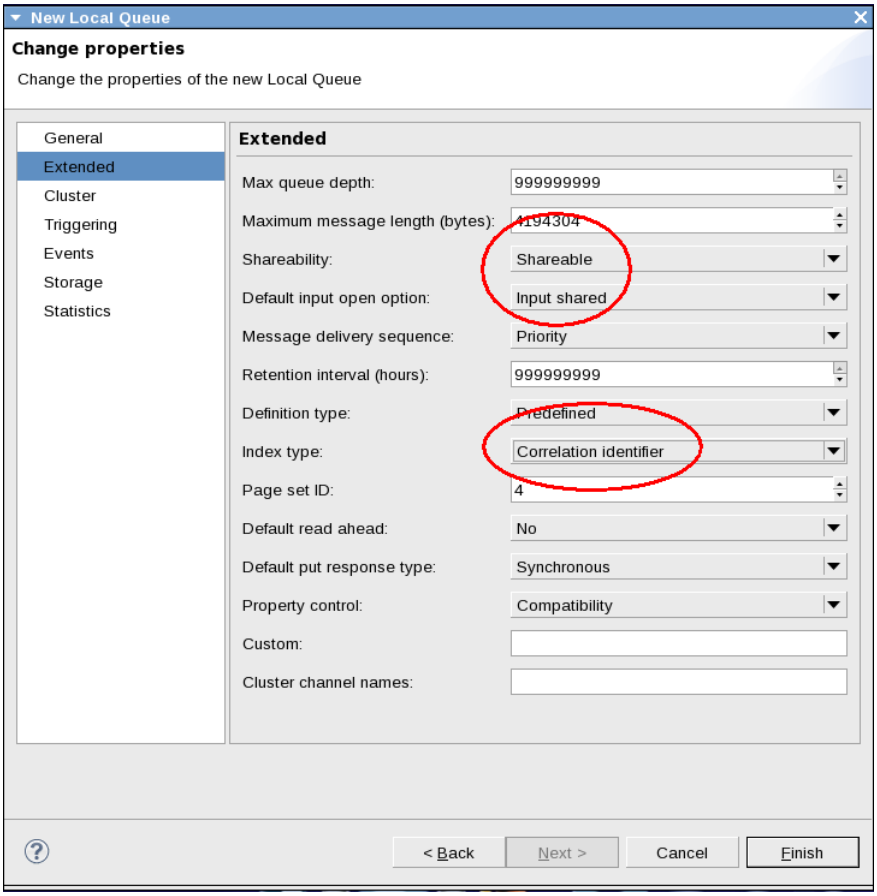
____7. Right mouse button click on *Queues* and select New –> *Local Queue*. This will open a *Create a Local Queue* pane (see below). Enter ***CICS.TRIGGER.REQUEST*** for the *Name* of the request queue. This queue will be where the request to be passed to the CICS program will be placed. Press the **Next** to continue.



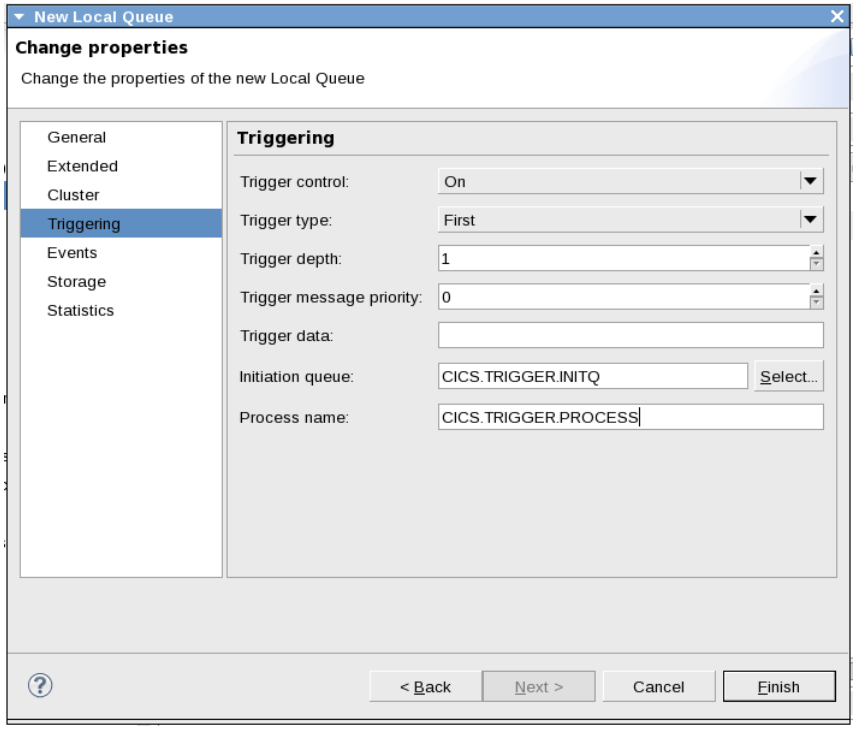
____8. This will open the *Change properties* window (see below). Enter a meaningful description and description and then select *Extended* on the left hand side of the pane.



____9. Use the pull-down arrows to set the *Shareability* property to *Sharable*, the *Default input open option* property to *Input shared* and to select an *Index type* of *Correlation identifier*. Then select *Triggering* on the left-hand side of the pane.



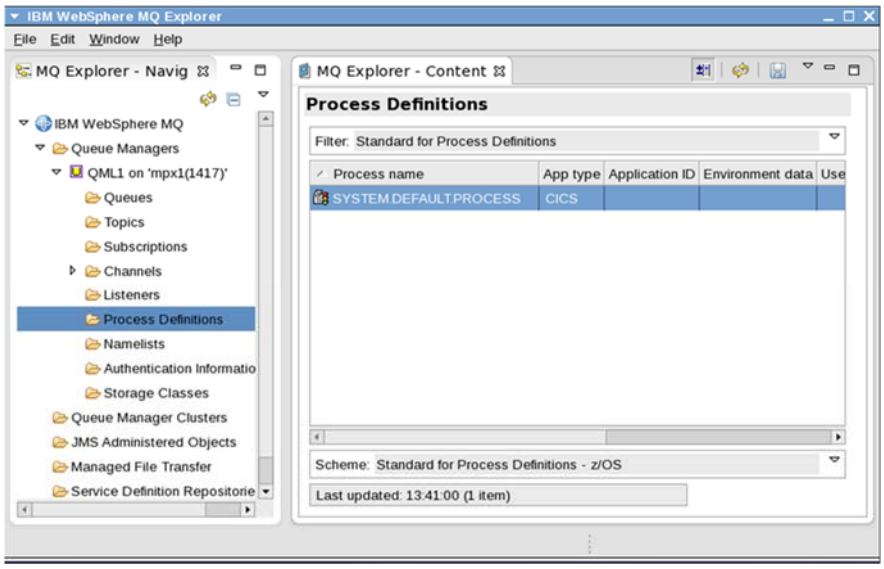
____10. Use the pull down arrow to select ***On*** for *Trigger control*, enter (or use the **Select** button to select) ***CICS.TRIGGER.INITQ*** for the *Initiation Queue* and enter ***CICS.TRIGGER.PROCESS*** for the *Process name*. Press the **Finish** button to complete the creation of this queue.



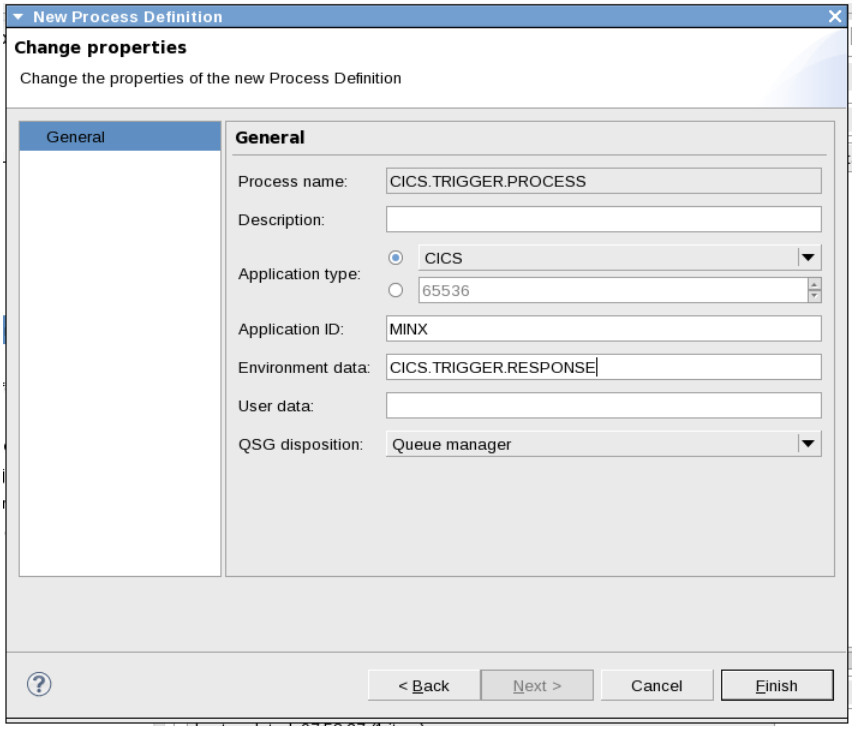
Tech-Tip: Not specifying On for **Trigger control** is the most common cause for problems when executing the clients later in this lab.

Tech-Tip: The *Trigger Type* property is how you control when the trigger event occurs. We are taking the default of *First* (trigger the bridge program when the first message arrives, with the expectation that the bridge program will process multiple messages), but you could also select *Every* (to start multiple instances of the bridge programs), *Depth* (to defer triggering until a certain number of messages have arrived on the request queue) or *None* (to stop triggering).

____11. Back on the IBM MQ Explorer main panel right mouse button click on *Process Definitions* and then select option *New -> Process Definition.*



____12. Enter ***CICS.TRIGGER.PROCESS*** for the *Process name* on the *New Process Definition – Create a Process Definition* and press **Next** to continue. On *the Process Definition – Change Properties* window use the pull down arrow to select *CICS* as the *Application type* and enter ***MINX*** (the user CICS transaction) as the *Application ID* (see below). We want to pass information to the target program that will be used to identify the queue where the response messages should be placed. This information is entered in the area beside *Environment data*. In this area enter ***CICS.TRIGGER.RESPONSE***. Press the **Finish** button to add this definition.



Tech-Tip: CICS transaction *MINX* is configured to start CICS program *ATSCMINX*. This program obtains the name of the response queue by retrieving the *Environment data* configured in the PROCESS resource passed with the request using the code below:

```
          01  MQM-TRIGGER-MESSAGE.
           COPY CMQTMV.


     *    Retrieve trigger message
          EXEC CICS RETRIEVE INTO(MQTM) END-EXEC.
          MOVE MQOT-Q            TO MQOD-OBJECTTYPE.
          MOVE MQTM-QNAME        TO MQOD-OBJECTNAME.
          MOVE MQTM-ENVDATA      TO RESPONSE-QUEUE-NAME.
```

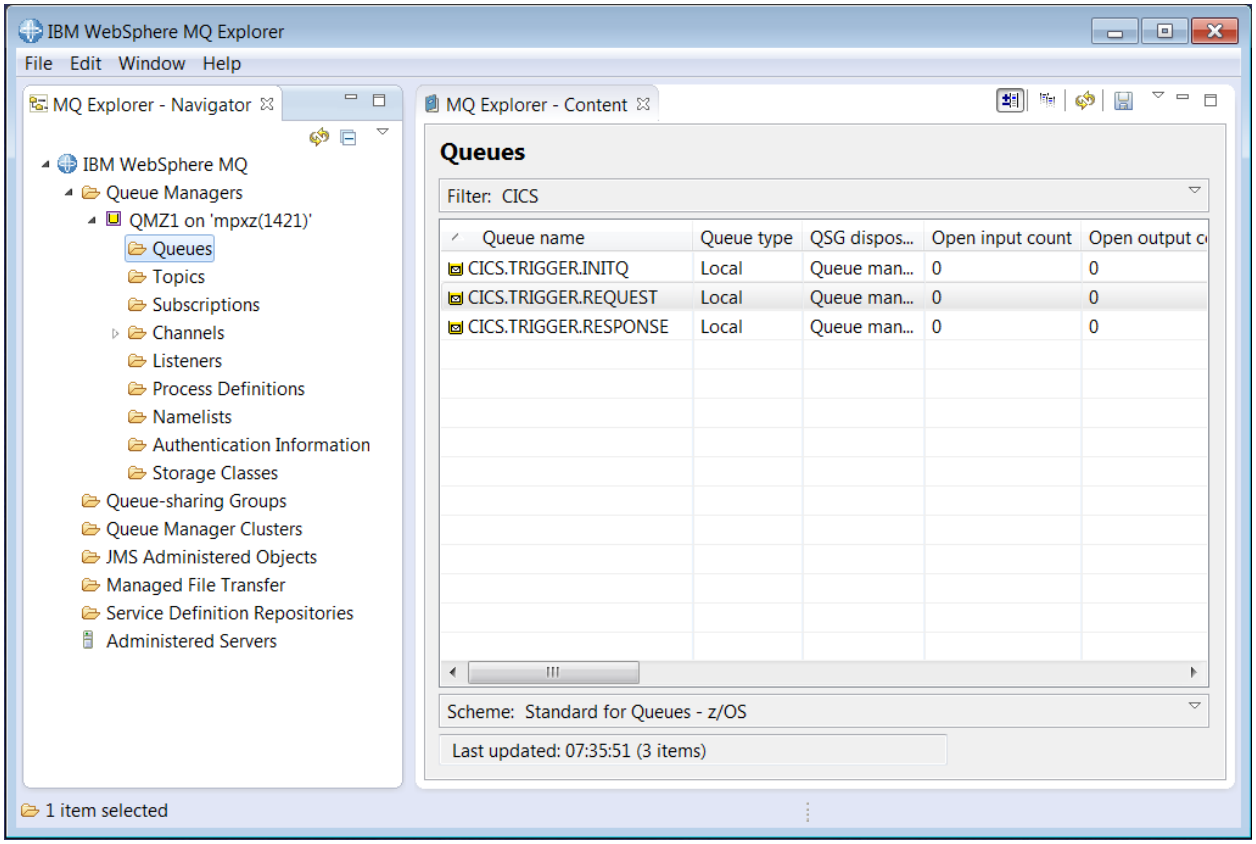
This completes the configuration of MQ Trigger resources.

Part 3 – Configuring MQ Bridge Resources

In this part of the exercise the steps required to define the MQ resources required for using the CICS Bridge will be performed using the MQ Explorer.

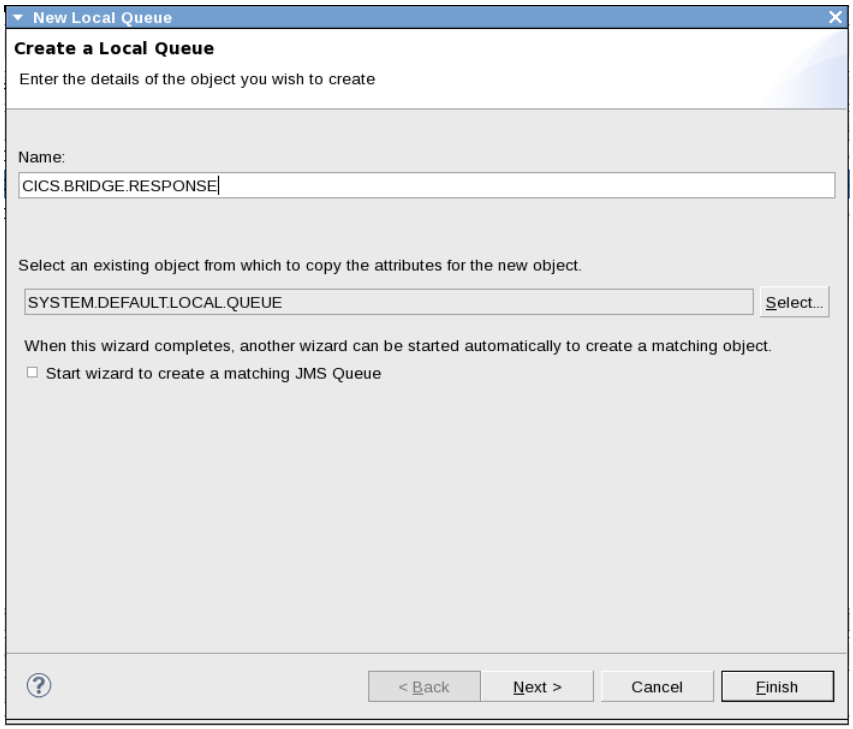
____1. Start MQ Explorer if not already active and connect to queue manager *QMZ1*.

____2. Once connected, select *Queues* to display a list of queues defined in this Queue Manager.

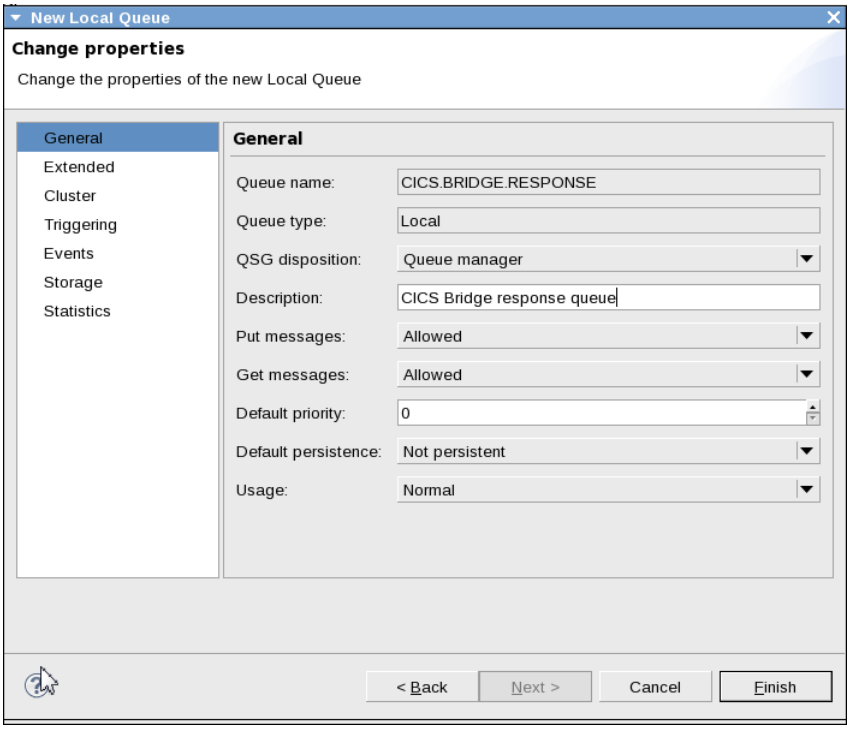


Tech-Tip: A filter of ***CICS.**** was being used in these screen shots to reduce the number of queues displayed.

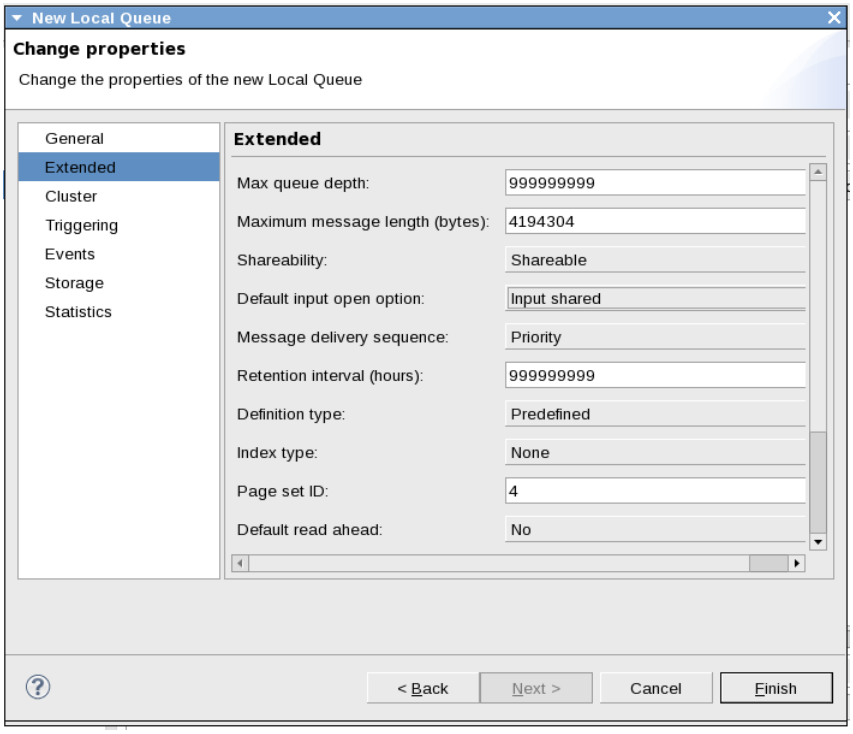
____3. Right mouse button click on *Queues* and select New –> *Local Queue*. This will open a *Create a Local Queue* pane (see below). Enter ***CICS.BRIDGE.RESPONSE*** for the *Name* of the response queue. This queue will be where the response from the CICS program can be found. Click **Next** to continue.



____4. This will open the *Change properties* window (see below). Enter a meaningful description and then select *Extended* on the left hand side of the pane.

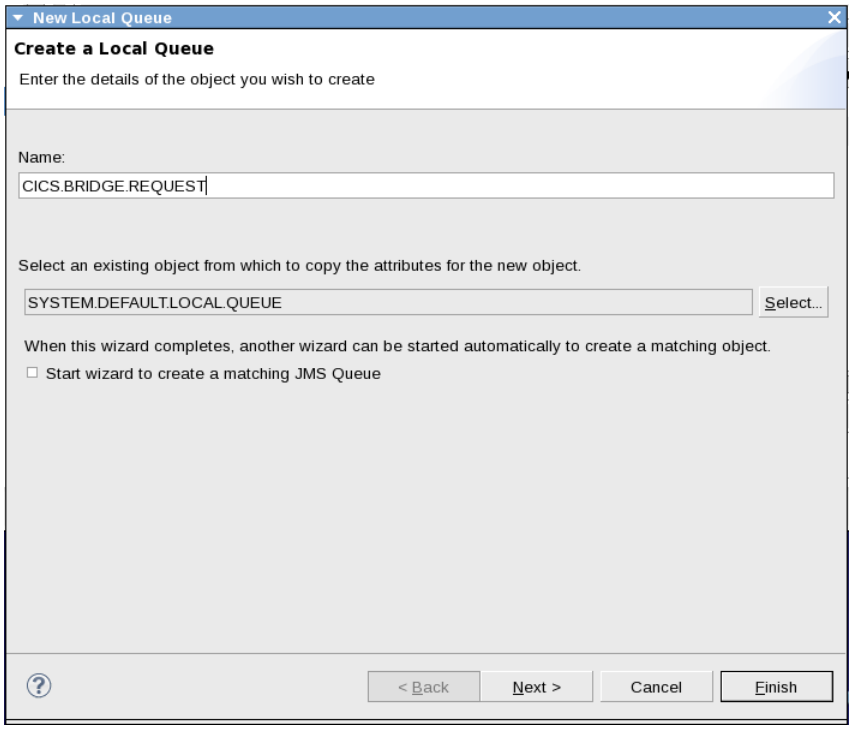


____5. On the *Extended* window use the arrows beside *Shareability* and *Default input open option* properties to make the queue *Sharable* and *Input shared*. Click **Finish** to continue.

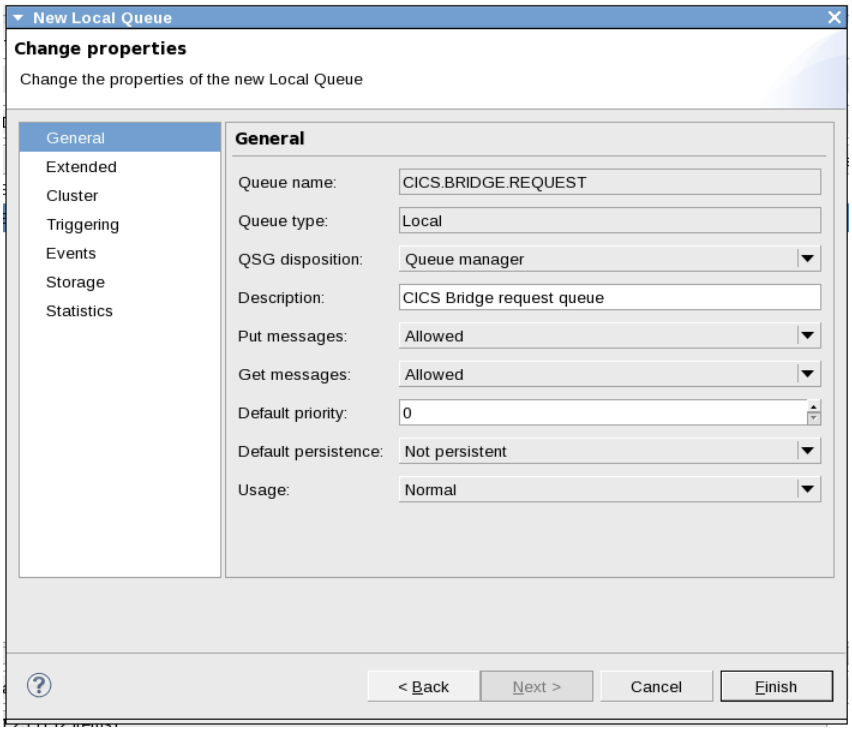


____6. Repeat Step 3 through 5 and create another local queue with the name ***CICS.BRIDGE.INITQ***. An instance of a CICS *CKTI* will be started for this queue and this queue will be monitored for notification or event messages, i.e., a message has arrived on the request queue.

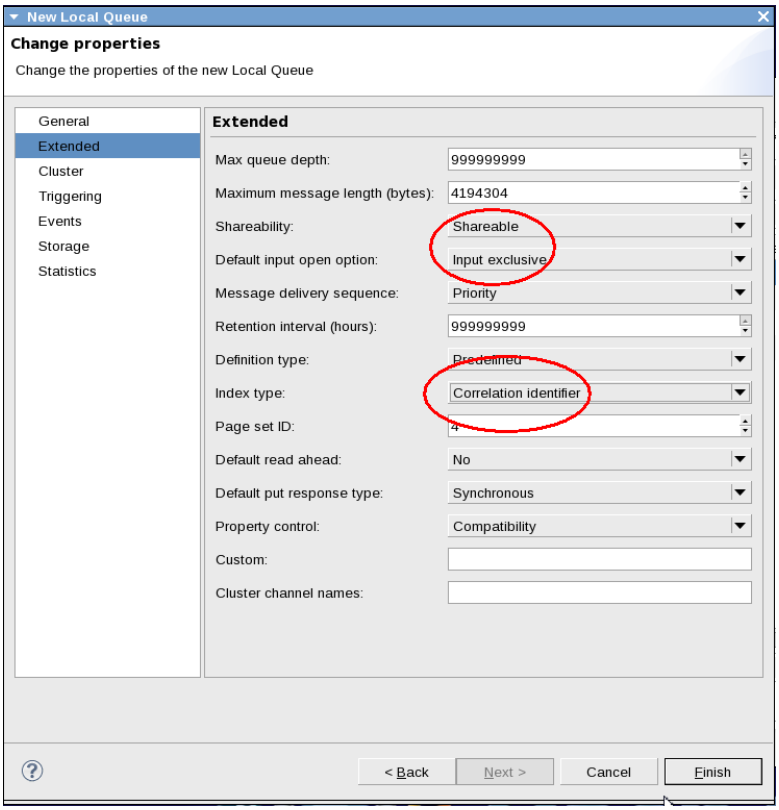
____7. Right mouse button click on *Queues* and select New –> *Local Queue*. This will open a *Create a Local Queue* pane (see below). Enter ***CICS.BRIDGE.REQUEST*** for the *Name* of the request queue. This queue will be where the request to be passed to the CICS program will be placed. Press the **Next** to continue.



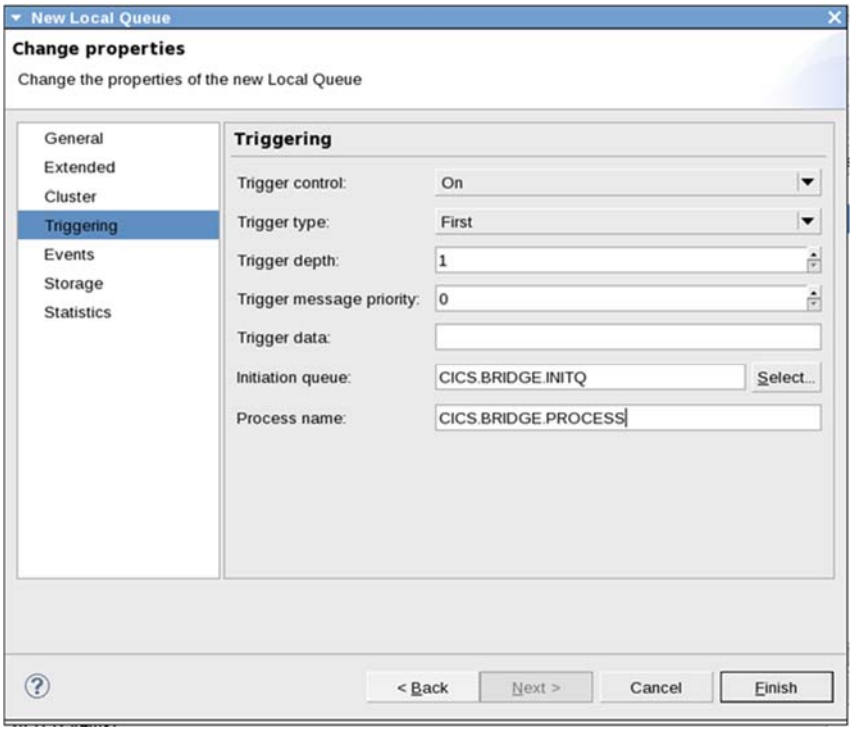
____8. This will open the *Change properties* window (see below). Enter a meaningful description and description and then select *Extended* on the left-hand side of the pane.



____13. Use the pull-down arrows to set the *Shareability* property to *Sharable*, the *Default input open option* property to *Input shared* and to select an *Index type* of *Correlation identifier*. Then select *Triggering* on the left-hand side of the pane.



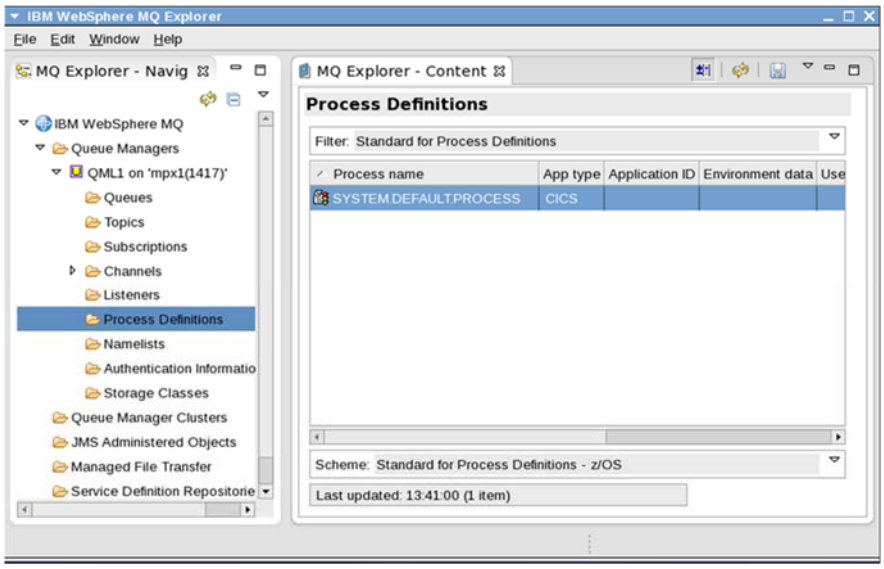
____9. Use the pull down arrow to select ***On*** for *Trigger control*, enter (or use the **Select** button to select) ***CICS.BRIDGE.INITQ*** for the *Initiation Queue* and enter ***CICS.BRIDGE.PROCESS*** for the *Process name*. Press the **Finish** button to complete the creation of this queue.



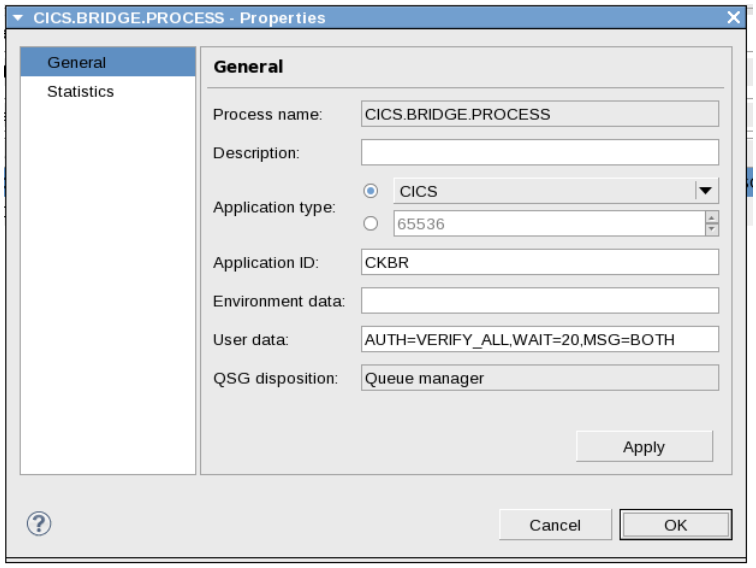
Tech-Tip: Not specifying On for **Trigger control** is the most common cause for problems when executing the clients later in this lab.

Tech-Tip: The *Trigger Type* property is how you control when the trigger event occurs. We are taking the default of *First* (trigger the bridge program when the first message arrives, with the expectation that the bridge program will process multiple messages), but you could also select *Every* (to start multiple instances of the bridge programs), *Depth* (to defer triggering until a certain number of messages have arrived on the request queue) or *None* (to stop triggering).

____10. Back on the IBM MQ Explorer main panel right mouse button click on *Process Definitions* and then select option *New -> Process Definition.*



____11. Enter ***CICS.BRIDGE.PROCESS*** for the *Process name* on the *New Process Definition – Create a Process Definition* and press **Next** to continue. On *the Process Definition – Change Properties* window use the pull down arrow to select *CICS* as the *Application type* and enter ***CKBR*** as the *Application ID* (see below). We want to pass information to the CICS Bridge that will be used to control the invocation of the target user program. This information is entered in the area beside *User data*. In this area enter ***AUTH=VERIFY_ALL,WAIT=20,MSG=BOTH***. Press the **Finish** button to add this definition.



This user data configures the CICS Bridge to:

- Extract the user identity and password from the request and for each message verify the user identity and password with the external security manager.
- Wait *20* seconds until timing out when waiting for subsequent requests when processing a unit of work that involves multiple programs.
- And to write CICS Bridge messages to *both* the CICS job log and master terminal.

Tech-Tip: We will be invoking a COMMAREA enabled user application so the MQCIH transaction will be either ***CKBP*** or another transaction code that invokes program *DFHMPBP0*.

If we were invoking a channel/container enabled user application we would have set the MQCIH transaction to ***CKBC*** or another transaction code that invokes program *DFHMQBP3*. The CICS Bridge creates a channel named DFHMQBR_CHANNEL and the containers are limited to one inbound container named DFHREQUEST and one outbound container named DFHRESPONSE.

This completes the configuration of MQ Bridge resources.

Part 4 – Configuring CICS MQMonitor resources

In this part of the exercise the CICS MQMonitor resources will be defined for both the *CICS.BRIDGE.INITQ* and *CICS.TRIGGER.INITQ* initiation queues. These resources will be installed into the CICS region and can be managed using the CEMT transaction.

Start at Step 6 if exercise *L13 – Implementing z/OS Queue Manager Security* has not been completed.

____1. Use RACF command *RDEFINE* to define a profile for these new queue giving *START3* update access.

RDEFINE MQQUEUE QMZ1.CICS.** OWNER(SYS1)

____2. *Use the RACF PERMIT* to reset all current access to null.

PERMIT QMZ1.CICS.** CLASS(MQQUEUE) RESET

____3. Use the RACF *PERMT* command to give UPDATE access to this queue to group *MQSTC* and *CICSSTC* and the CICS default user (SIT parameter DFLTUSER=CICSUSER).

PERMIT QMZ1.CICS.** CLASS(MQQUEUE)
ID(MQSTC,CICSSTC,CICSUSER,MQUSERS) ACC(UPDATE)

> **Tech-Tip:** In this workshop we do not use mixed case MQ resources so the case of the queues, etc. name is not important in this workshop. If you are using mixed case for your MQ resources there are corresponding RACF resources to the one used in this workshop. For example MXQUEUE is the mixed case equivalent of MQQUEUE and MXADMIN is the mixed case equivalent of MQADMIN.

____4. Refresh the RACF instorage profiles will command RACF command *SETROPTS.*

SETROPTS RACLIST(MQQUEUE) REFRESH

____5. Refresh the queue manager's RACF information with MVS command

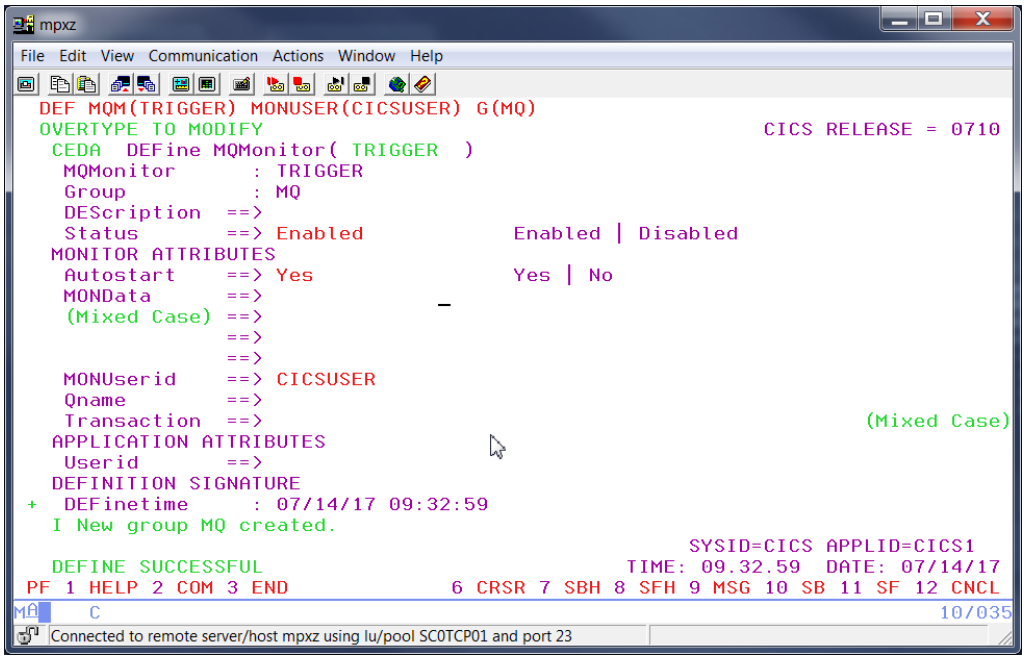
QMZl REFRESH SECURITY(MQQUEUE)

You should see these messages in the console.

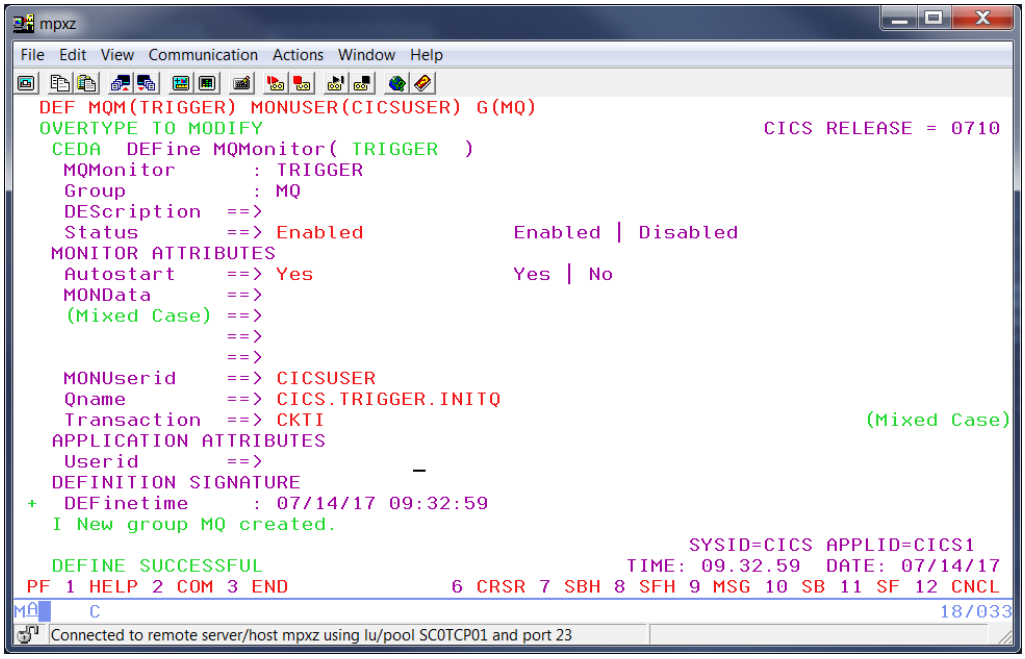
```
QMZ1 REFRESH SECURITY(MQQUEUE)
CSQH001I QMZ1 CSQHCHK4 Security using uppercase classes
CSQ9022I QMZ1 CSQHSREF ' REFRESH SECURITY' NORMAL COMPLETION
```

____6. Start a 3270 session with CICS region CICS1.

____7. Clear the screen and enter ***CEDA DEF MQMonitor(TRIGGER) MONUSER(CICSUSER) GROUP(MQ)*** in the 3270 session. Press **Enter** to continue. In the 3270 session the message *New group MQ created* should be displayed (see below).

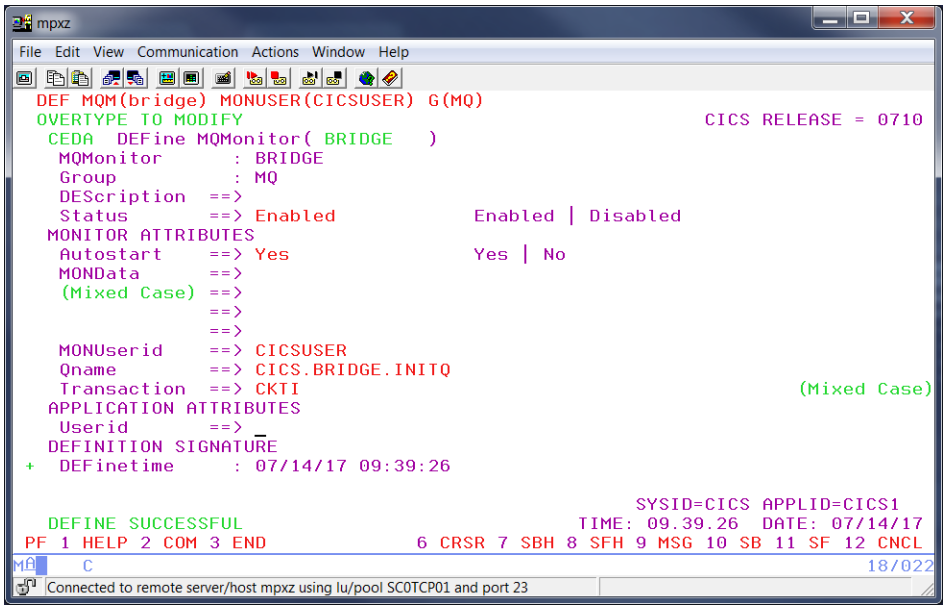


____8. Enter ***CICS.TRIGGER.INITQ*** in the area beside *Qname* and ***CKTI*** in the area beside *Transaction* session. Press **Enter** to continue.



Tech-Tip: Setting the *AutoStart* attribute to ***Yes*** will automate the starting of the required CKTI task for this initiation queue.

____9. Repeat these steps to define an MQMonitor named BRIDGE for queue ***CICS.BRIDGE.INITQ***.



____10. Next use the CICS transaction ***CEDA IN G(MQ)*** to install these new resources.

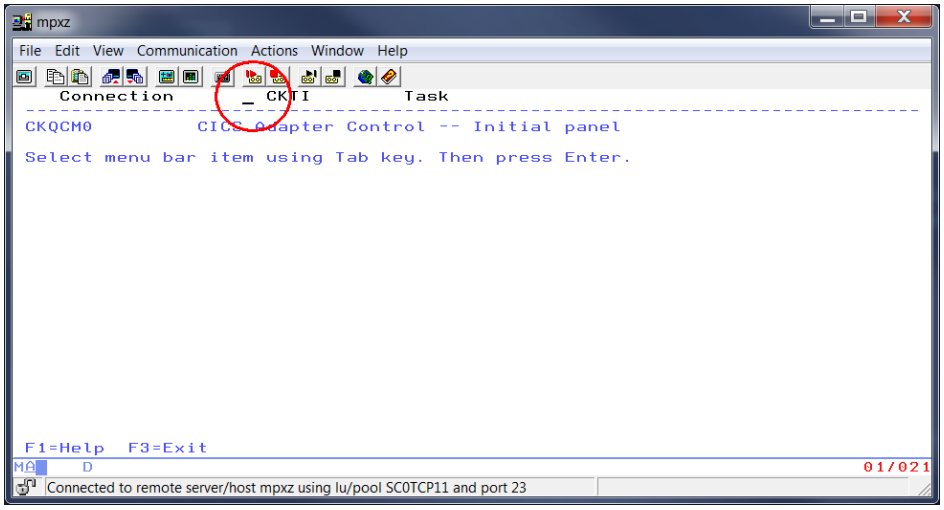
____11. Clear the screen and use the CICS transaction CEMT to start these monitors, first the TRIGGER monitor, e.g.:

CEMT SET MQM(TRIGGER) START

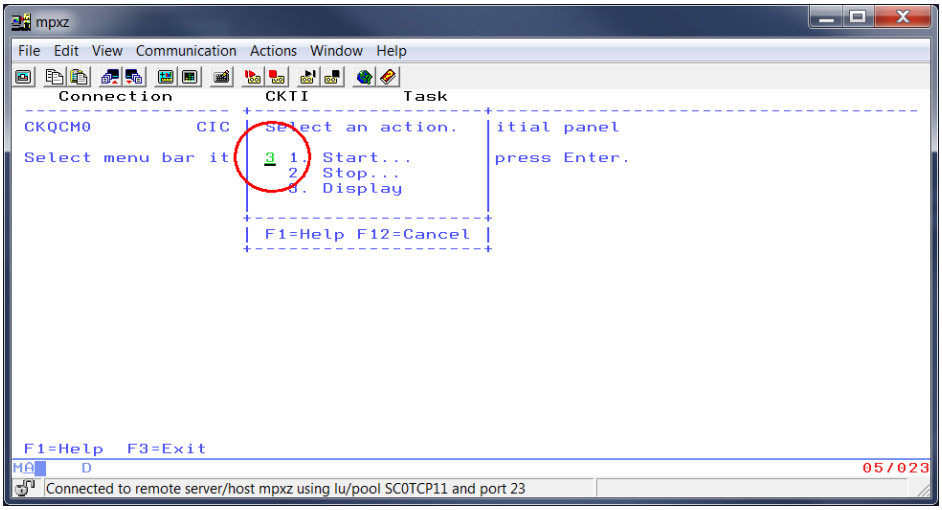
Then clear the screen again and start the BRIDGE monitor.

CEMT SET MQM(BRIDGE) START

____12. Clear the 3270-session screen again and enter CICS transaction ***CKQC*** and press **Enter**. This will display the screen below. Use the keyboard's arrow keys to move the cursor to CKTI on the 3270 terminals tool bar line and press **Enter** to continue.



____13. This will display a drop down box with three *CKTI* related options. ***1*** could be used to *Start* a new *CKTI* instance, ***2*** could be used to *Stop* an existing *CKTI* instance and ***3*** could be used to *Display* the currently active *CKTI* instances. Enter ***3*** and press the **Enter** key to continue.



____14. You should see results like what is shown below. There are three *CKTI* tasks or instances active. Two we just started for *ZCONN2.TRIGGER.INITQ* and another for *CICS01.INITQ* which was used in a previous exercise. CKTI task for the MQMonitor resources just added will not be displayed until a the monitors are started.

____15. Become familiar with the options provided by the CKQC transaction/application and when finished use a combination of **F12** keys (*Cancel*) or **F3** keys (*Exit*) to terminate this transaction and leave the 3270 session on a clear screen.

This completes the configuration of CICS resources.

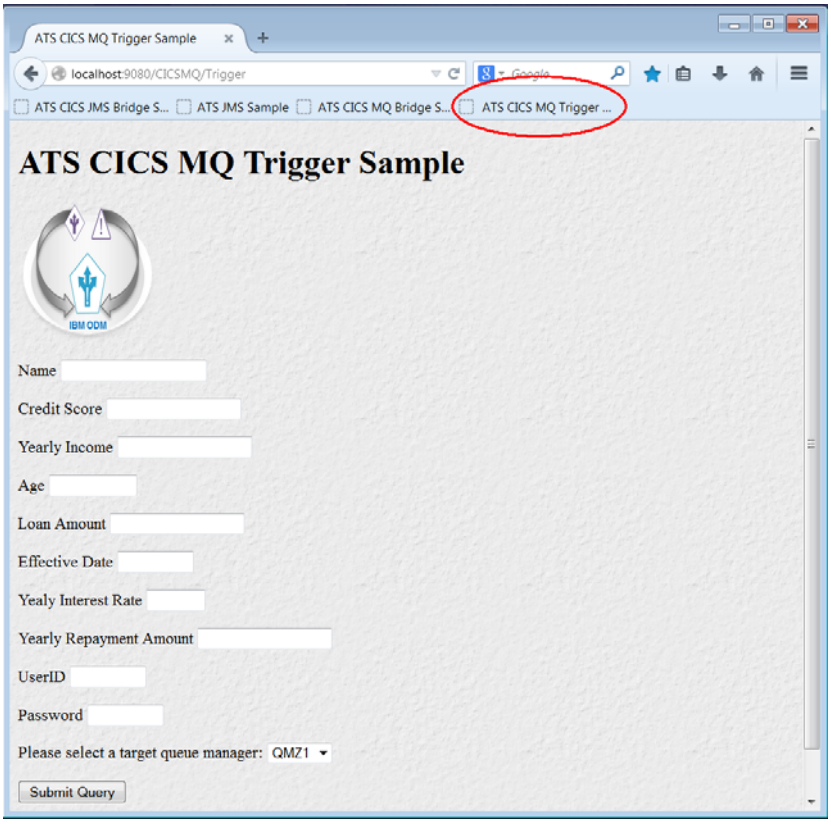
Part 5 – Testing the CICS Trigger

In this part we will test the MQ and CICS configuration completed in the first parts of this exercise. This will have required starting the Liberty server that in running in the Linux portion of the image and using a web browser to access a simple JSP/Server application that is running in Liberty. This application will send requests to a CICS program and display the results.

Testing of this application requires a Channel Authorization Record that allows access to channel Client.to.QueueMgr. Such a record was added in exercise L13 – Implementing z/OS Queue Manager Security. If exercise L13 has not been completed, then Steps 7 through 12 in Part 3 of that exercise should be performed now.

____1. Next open the *Mozilla Firefox* on the desktop and selecting the icon and using the right mouse button to select the *Open* option.

____2. On the *Mozilla Firefox* session you should see an option on the *Bookmarks Toolbar* for *ATS CICS MQ Trigger Sample* (see below). Click this option and you should see the web page below.



This client application that invokes a CICS application that access Operational Decision Management rules for approving loan requests.

The rules for rejecting a loan any one of the following:

- If the credit score of the borrower is less than 300.
- If the yearly repayment amount is more than 30% of the borrower's income.
- If the income of the borrower is less than $24,000.
- If the age of the borrower is more than 65 years.
- The loan amount is more than $1,000,000.

The data entry editing of this client is very simple. So enter numbers for *Credit Score, Yearly Income, Age, Loan Amount, Yearly Interest Rate and Yearly Repayment Amount* with no commas are any other punctuation. *Name* and *Effective Date* can be any values; these fields have no restrictions.

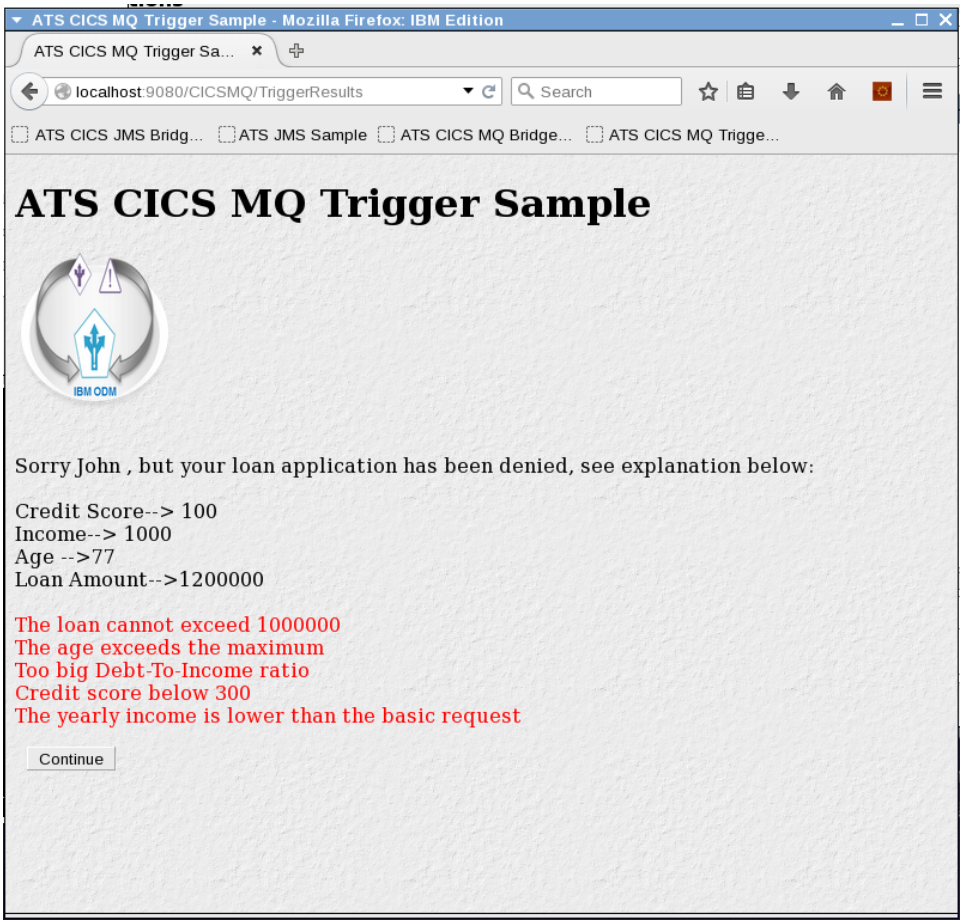
If interested, the 3270 interface to this application can be started with CICS transaction ***LOAN***.

____3. To get started enter ***John*** for the *Name*, ***100*** for the *Credit Score*, ***1000*** for the *Yearly Income, **77*** for the *Age*, ***1200000*** for the *Loan Amount*, ***12/12/16*** for the *Effective Date*, ***00005*** for the *Yearly Interest Rate, **1000*** for the *Yearly Repayment Amount* and you're your assigned user ID and password. Press the **Submit Query** button to continue.

If exercise *L13 – Implementing z/OS Queue Manager Security* has not been completed, then a channel authentication rule added in that exercise needs to be added now. Locate that exercise and perform Steps 7 through 12 in Part 3 of that exercise to add the required channel authentication rule.

> **Tech-Tip:** The time out value is set relatively low for this application. If the request fails with a 2033 response, retry the re quest.

____4. You should see something like the screen below.



____5. Click the **Continue** button to return to the initial screen.

____6. Enter the same or different information on the screen but this time enter an invalid user ID and/or password and press **Continue**. Note that you must provide values for the numeric fields, otherwise a *NumberFormatException* will occur. You should see a message like below on the screen.

MQJE001: Completion Code '2', Reason '2035'.
null
null
null
null

____7. Use the MVS console to identify the issue.

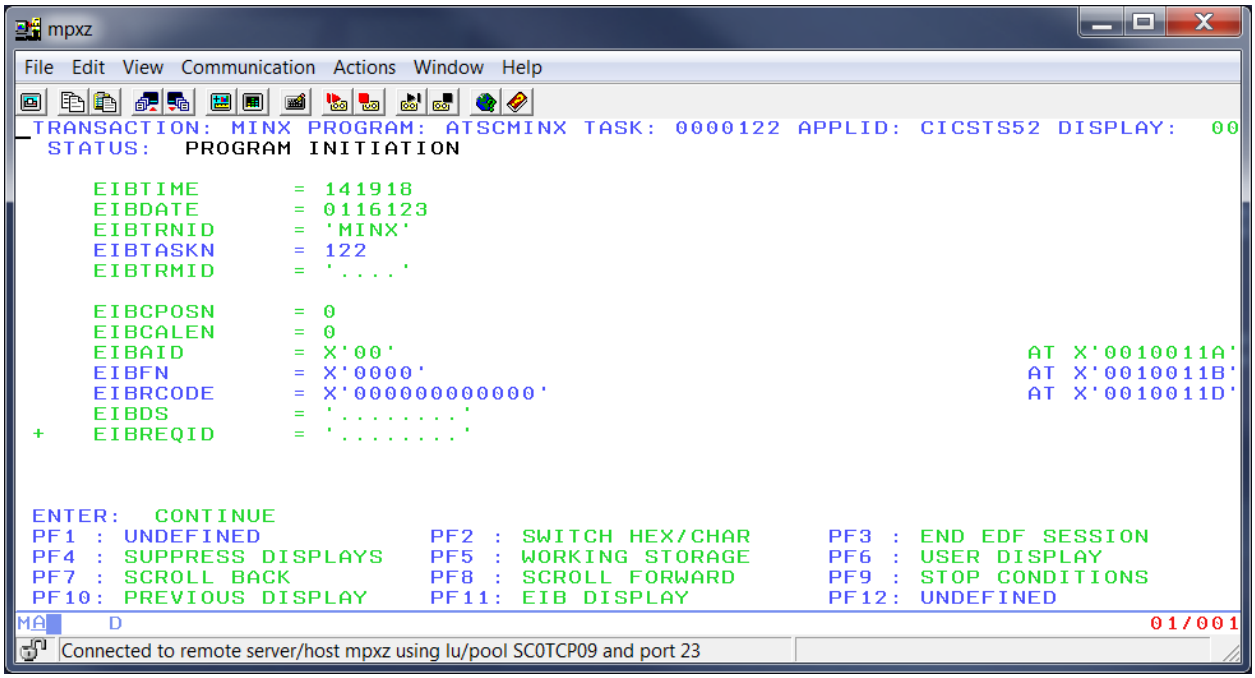
If the user identity was invalid you should see a message like the one below on the console.

IRR012I VERIFICATION FAILED. USER PROFILE NOT FOUND.

If an invalid password was entered you should see a message like the one below on the console.

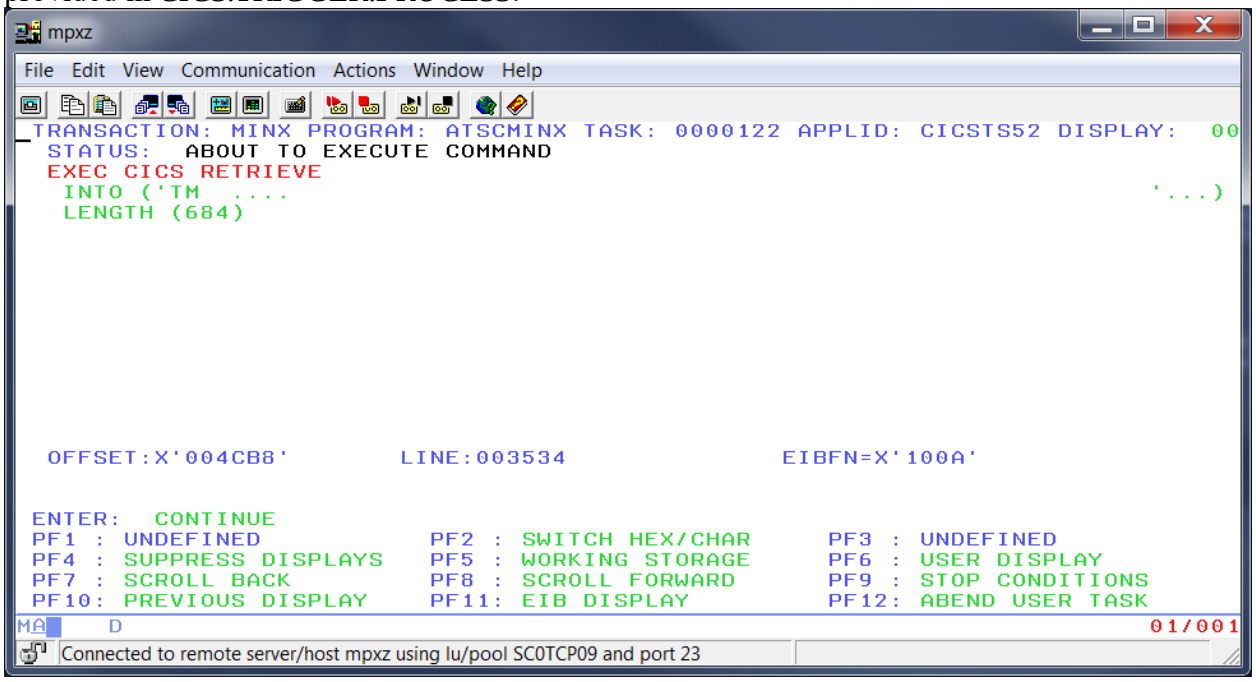
IRR013I VERIFICATION FAILED. INVALID PASSWORD GIVEN.

____8. In a CICS terminal session clear the screen enter CICS transaction ***CEDX MINX*** and then drive another request from the Mozilla session. The screen below should appear in the 3270 session showing the start of the CICS program, note that there is no COMMAREA.

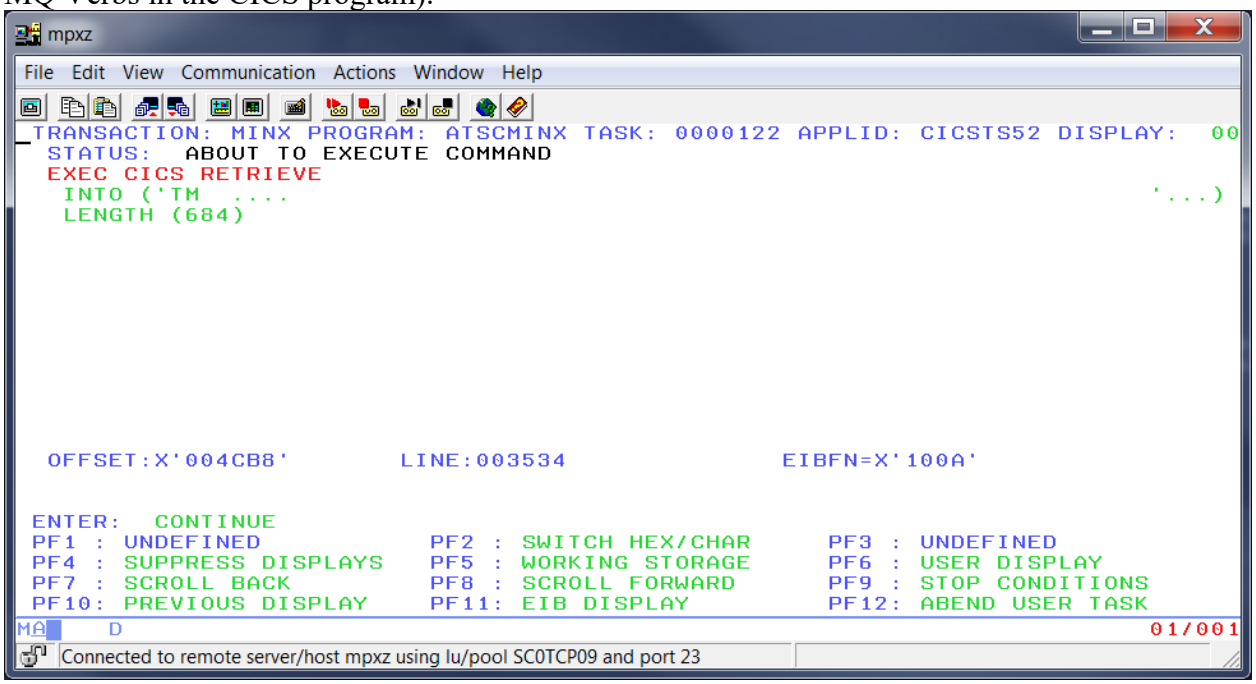


The EDF trace will stop the execution of the request and an error message will appear in the Mozilla session because the request will have timed out waiting for a response.

____9. Continue pressing **Enter** and you will see the *EXEC CICS RETRIEVE* of the *Environment data* provided in *CICS.TRIGGER.PROCESS*.



____10. Continue pressing **Enter** and you will see the calls to the MQM resource manager (these are the MQ Verbs in the CICS program).



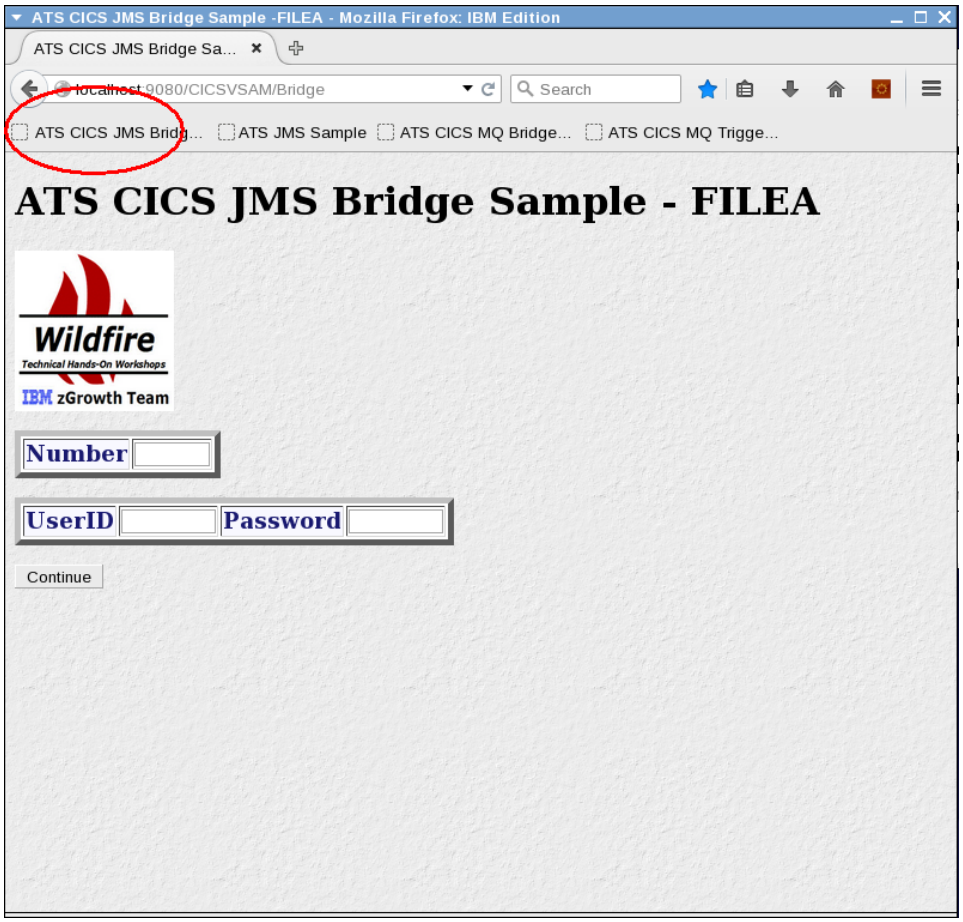
____11. Continue pressing Enter until the program terminates or you can use the **F3** key to terminate the CICS EDF trace.

This completes the testing of MQ triggering.

Part 6 – Testing the CICS Bridge Monitor

In this part we will test the MQ and CICS configuration completed in the first parts of this exercise. This application will send requests to a CICS program and display the results using a web browser to access a simple JSP/Server application that is running in Liberty.

____1. On the *Mozilla Firefox* session you should see an option on Bookmarks Toolbar for *ATS CICS JMS Bridge* (see below). Click this option and you should see the web page below.



This client application will use CICS MQ triggering to link to CICS program CSCVINQ. This program accepts a COMMAREA and based on the request will either Select (or retrieve), Insert, Update or Delete a record from a key sequenced VSAM file.

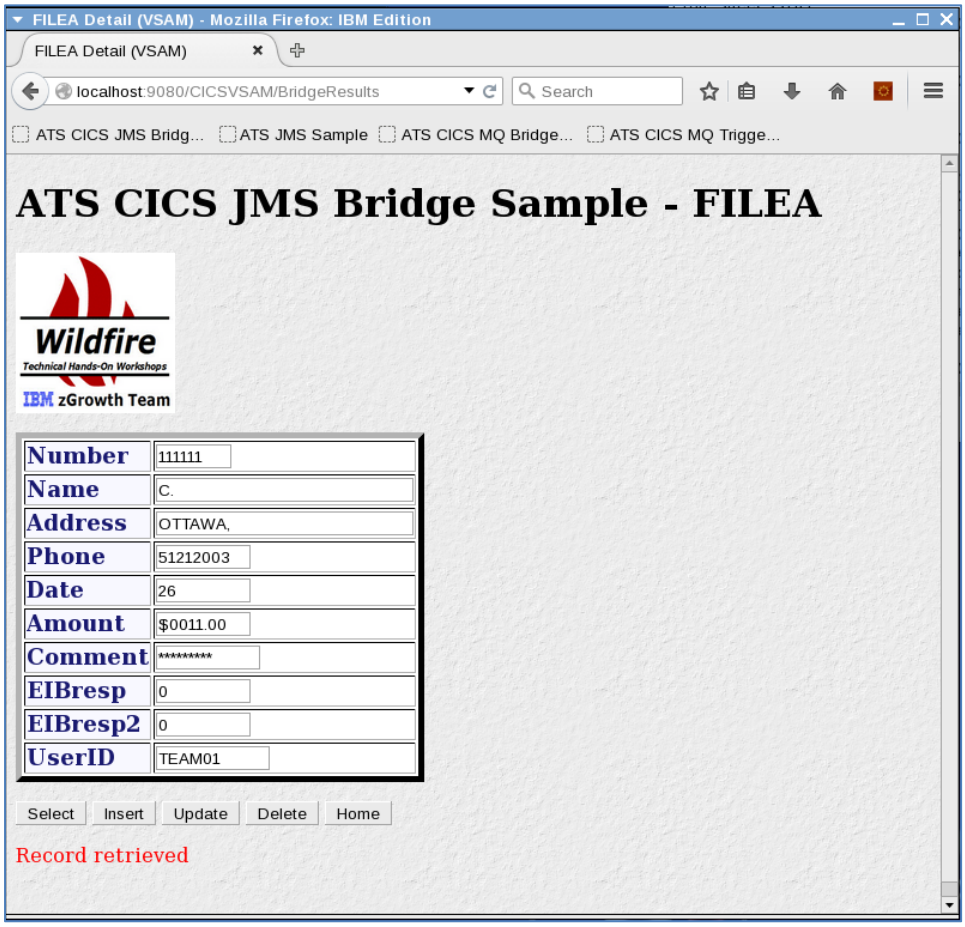
The VSAM file (DD name FILEA) is part of the CICS provided sample application. The VSAM file contains the following records (the key is the first 6 characters of the record).

```
000100S. D. BORMAN        SURREY, ENGLAND     3215677826 11 81$0100.11Y
000102J. T. CZAYKOWSKI    WARWICK, ENGLAND    9835618326 11 81$1111.11Y
000104M. B. DOMBEY        LONDON,ENGLAND      1284629326 11 81$0999.99Y
000106A. I. HICKSON       CROYDON, ENGLAND    1948567326 11 81$0087.71Y
000111ALAN TULIP          SARATOGA,CALIFORNIA 4612075301 02 74$0111.11Y
000762SUSAN MALAIKA       SAN JOSE,CALIFORNIA 2231212101 06 74$0000.00Y
000983J. S. TILLING       WASHINGTON, DC      3451212021 04 75$9999.99Y
001222D.J.VOWLES          BOBLINGEN, GERMANY  7031555110 04 73$3349.99Y
001781TINA J YOUNG        SINDELFINGEN,GERMANY7031999021 06 77$0009.99Y
003210B.A. WALKER         NICE, FRANCE        1234567026 11 81$3349.99Y
003214PHIL CONWAY         SUNNYVALE, CAL.     3411212000 06 73$0009.99N
003890BRIAN HARDER        NICE, FRANCE        0000000028 05 74$0009.99N
004004JANET FOUCHE        DUBLIN, IRELAND     7111212102 11 73$1259.99N
004445DR. P. JOHNSON      SOUTH BEND, S.DAK.  6121212026 11 81$0009.99N
004878ADRIAN JONES        SUNNYVALE, CALIF.   3221212010 06 73$5399.99N
005005A. E. DALTON        SAN FRANCISCO, CA.  0000000101 08 73$0009.99N
005444ROS READER          SARATOGA, CALIF.    6771212020 10 74$0809.99N
005581PETE ROBBINS        BOSTON, MASS.       4131212011 04 74$0259.99N
006016SIR MICHAEL ROBERTS NEW DELHI, INDIA    7033121121 05 74$0009.88Y
006670IAN HALL            NEW YORK, N.Y.      2121212031 01 75$3509.88N
006968J.A.L. STAINFORTH   WARWICK, ENGLAND    5671382126 11 81$0009.88Y
007007ANDREW WHARMBY      STUTTGART, GERMANY  7031100010 10 75$5009.88N
007248M. J. AYRES         REDWOOD CITY, CALF. 3331212111 10 75$0009.88N
007779MRS. A. STEWART     SAN JOSE, CALIF.    4151212003 01 75$0009.88Y
009000P. E. HAVERCAN      WATERLOO, ONTARIO   0987654321 01 75$9000.00Y
100000M. ADAMS            TORONTO, ONTARIO    0341512126 11 81$0010.00Y
111111C. BAKER            OTTAWA, ONTARIO     5121200326 11 81$0011.00Y
200000S. P. RUSSELL       GLASGOW,  SCOTLAND  6373829026 11 81$0020.00Y
222222DR E. GRIFFITHS     FRANKFURT, GERMANY  2003415126 11 81$0022.00Y
300000V. J. HARRIS        NEW YORK, U.S.      6473980126 11 81$0030.00Y
333333J.D. HENRY          CARDIFF, WALES      7849302026 11 81$0033.00Y
400000C. HUNT             MILAN, ITALY        2536373826 11 81$0040.00Y
444444D. JACOBS           CALGARY, ALBERTA    7788982026 11 81$0044.00Y
500000P. KINGSLEY         MADRID, SPAIN       4445464026 11 81$0000.00Y
555555S.J. LAZENBY        KINGSTON, N.Y.      3994442026 11 81$0005.00Y
600000M.F. MASON          DUBLIN, IRELAND     1239878026 11 81$0010.00Y
666666R. F. WALLER        LA HULPE, BRUSSELS  4298384026 11 81$0016.00Y
700000M. BRANDON          DALLAS, TEXAS       5798432026 11 81$0002.00Y
777777L.A. FARMER         WILLIAMSBURG, VIRG. 9187613126 11 81$0027.00Y
800000P. LUPTON           WESTEND, LONDON     2423338926 11 81$0030.00Y
888888P. MUNDY            NORTHAMPTON, ENG.   2369163926 11 81$0038.00Y
900000D.S. RENSHAW        TAMPA, FLA.         3566812026 11 81$0040.00Y
999999ANJI STEVENS        RALEIGH, N.Y.       8459163926 11 81$0049.00Y
```

If interested, the 3270 interface to the sample application can be started with CICS transaction ***AMNU***.

____2. Choose one of the keys from a record above (e.g. ***111111***) and enter in the area beside *Number*. Enter your team identifier and password and press the **Continue** button.

____3. You should see something like the screen below.



Try some of the other functions (e.g., Insert, Update or Delete) with the records provided by the CICS sample or use your own data.

____4. Click the **Home** button to return to the initial screen.

____5. Enter a record key the area beside *Number* but this time enter an invalid user ID and/or password and press **Continue**. You should see a message like below on the screen.

MQJCA1011: Failed to allocate a JMS connection.MQJCA1011: Failed to allocate a JMS connection.com.ibm.msg.client.jms.DetailedJMSSecurityException: JMSWMQ2013: The security authentication was not valid that was supplied for QueueManager 'QMZ1' with connection mode 'Client' and host name 'localzos(1421)'. Please check if the supplied username and password are correct on the QueueManager to which you are connecting

____6. Use the MVS console to identify the issue.

If the user identity was invalid you should see a message like the one below on the console.

CSQX777E QMZ1 CSQXRESP Channel SYSTEM.DEF.SVRCONN from LINUX (192.27.216.40) has been blocked due to USERSRC(NOACCESS), Detail: CLNTUSER(DDD)

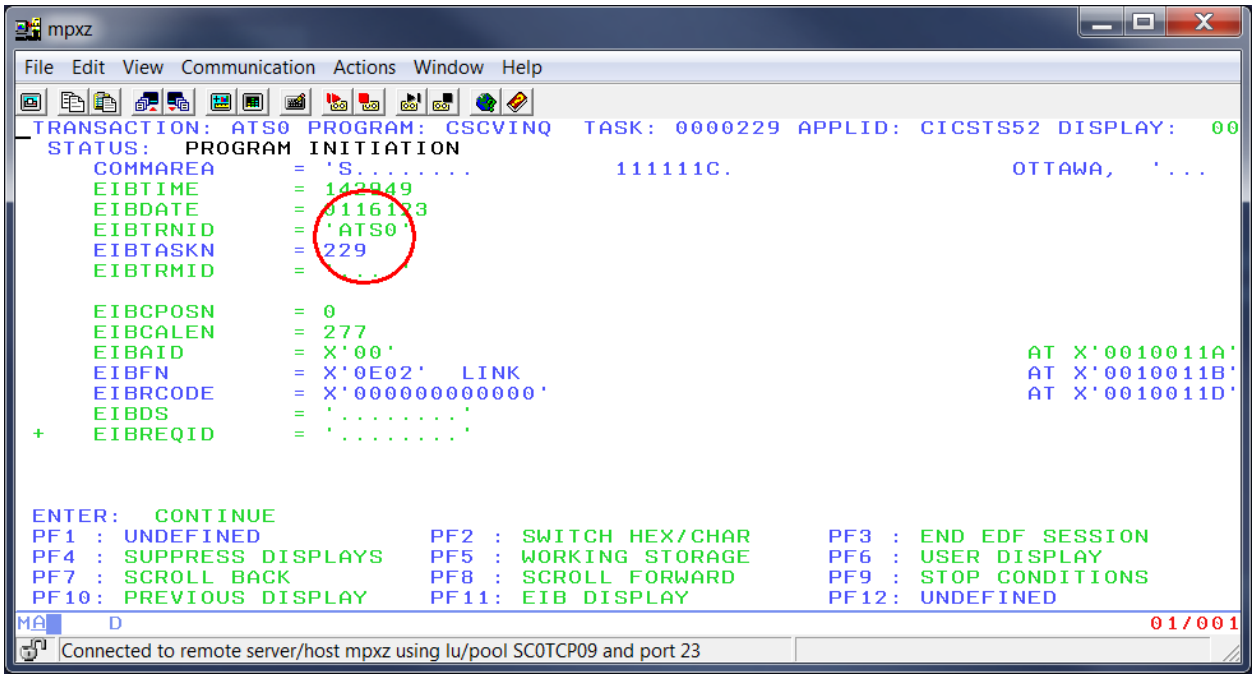
The request was blocked by channel authentication rules.

If an invalid password was entered you should see a message like the one below on the console.

IRR013I VERIFICATION FAILED. INVALID PASSWORD GIVEN.

This was because the specification of *AUTH=VERIFY_ALL* in the *User data* area of the *CICS.BRIDGE.PROCESS* requested the verification of user ID and password for each trigger request.

____7. *The JMS application running in Liberty includes an MQCIH* structure in each message. The MQCHI header has a transaction field which is set to transaction *ATS0*. In a CICS terminal session enter CICS transaction ***CEDX ATS0*** and then drive another request from the Mozilla session. The screen below should appear in the 3270 session showing the start of the CICS program and the COMMAREA being passed from MQ.



The EDF trace will stop the execution of the request and an error message will appear in the Mozilla session because the request will have timed out waiting for a response.

CONGRATULATIONS!!! You have completed this exercise

Summary

In this exercise you define the MQ resources (queues and processes) required for sending messages to CICS and starting CICS applications. Both basic MQ triggering and MQ CICS Bridge resources were defined using the MQ Explorer. The security resources required to protect these queues were defined to RACF and then the CICS CKTI transactions were started to monitor the triggering initiation queues.

A simple JSP/Servlet application running in Liberty was used to test both methods of sending MQ messages to CICS while at the same time exploring the security implications.