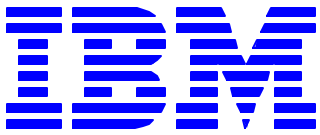


Lab 3
WebSphere Message Broker V6
Flow Debug
December, 2005

Lab Document
V1.1

Last updated: 25 Nov 2005

Table of Contents

1. Objectives

This exercise walks through the process of debugging three possible routes through a simple message flow.

The exercise uses a project named LAB3_Debug_MFP containing LAB3_Debug.msgflow and LAB3_Debug.esql (under the default schema).

NOTE: This lab was built using beta code. Certain screen shots may differ somewhat from what you see due to differences in GA code, i.e., references to version “6.B” rather than version “6.0”.

2. Environment Set-up

In order to complete this exercise the following should be installed and configured in a Windows 2000 or Windows XP environment:

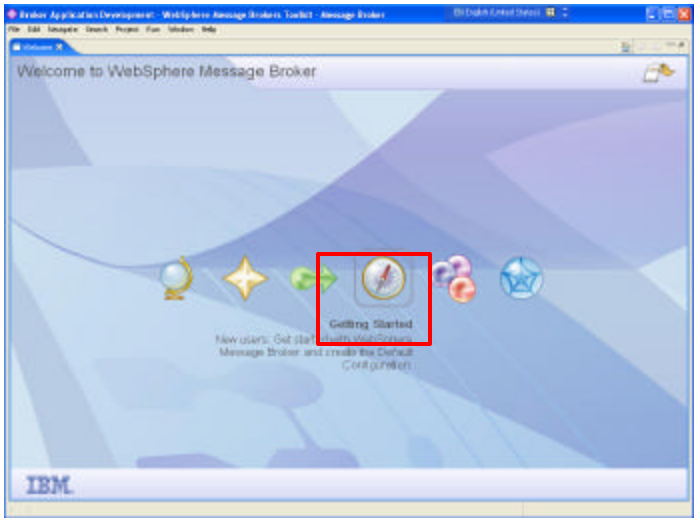
- ✓ WMBv6 Runtime containing:
  - o *WBRK6_DEFAULT_CONFIGURATION_MANAGER* or other configuration manager
  - o *WBRK6_DEFAULT_BROKER* or other broker
  - o *default* execution group
  - o no flows running

- ✓ WMBv6 Toolkit containing:
  - o *LAB3_Debug_MFP* containing *LAB3_Debug.msgflow* and *LAB3_Debug.esql* (under the default schema).

- ✓ WMQ queues:
  - o *LAB3_Debug_IN*
  - o *LAB3_Debug_OUT1*
  - o *LAB3_Debug_OUT2*
  - o *LAB3_Debug_FAILURE*

To establish this environment:

1. Open the WMBv6 Toolkit and when prompted for the workspace select <Lab Home>\workspace, i.e., c:\student\workspace. If it doesn't already exist and you prefer to use the default configuration, launch the *Default Configuration* wizard found under *Getting Started*. Follow the steps to complete it. If you already completed the Introduction lab this would have been done.



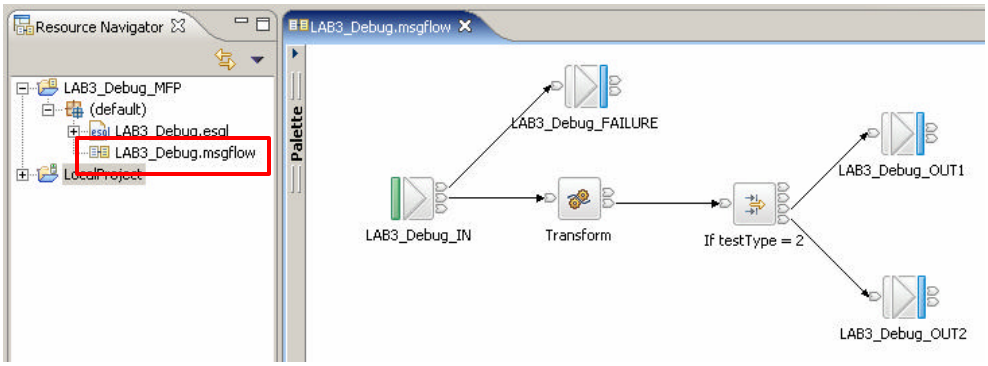
2. Import the message flow project for this lab from <Lab Home>\Lab3\install directory.

3. Set up the exercise queue by:
    a. Open a dos command prompt.
    b. Cd to <Lab Home>\Lab3\install directory.
    c. Run *Lab3mq*.

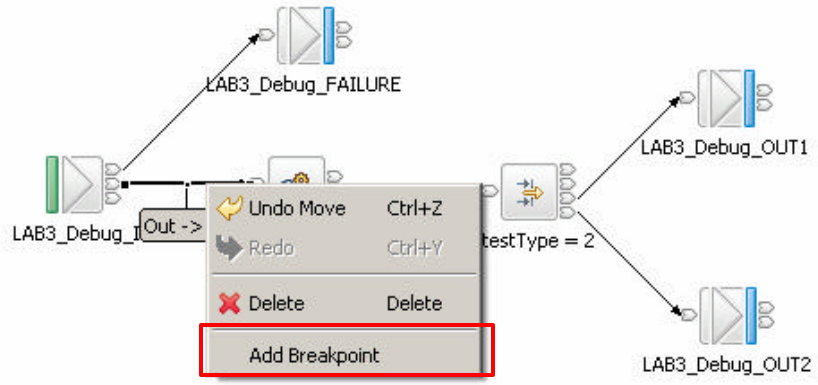
3. Setting Break Points

3.1 Message Flow Break Points

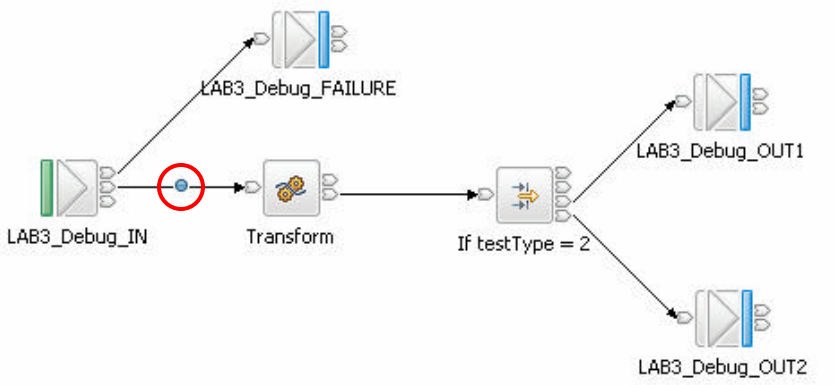
4. Within the Broker Application Development perspective open the LAB3_Debug message flow.



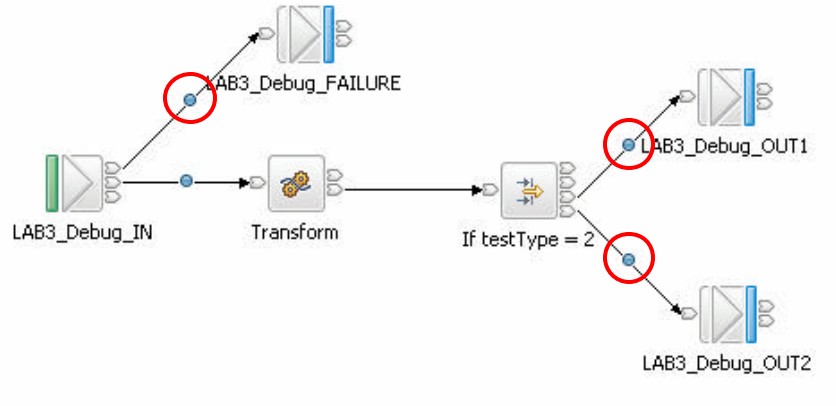
5. To add the first message flow break point right click on the connection between the Out terminal of the LAB3_Debug_IN node and the In terminal of the Transform node. From the drop down select Add Breakpoint.



The break point should then be shown on this connection (highlighted below).

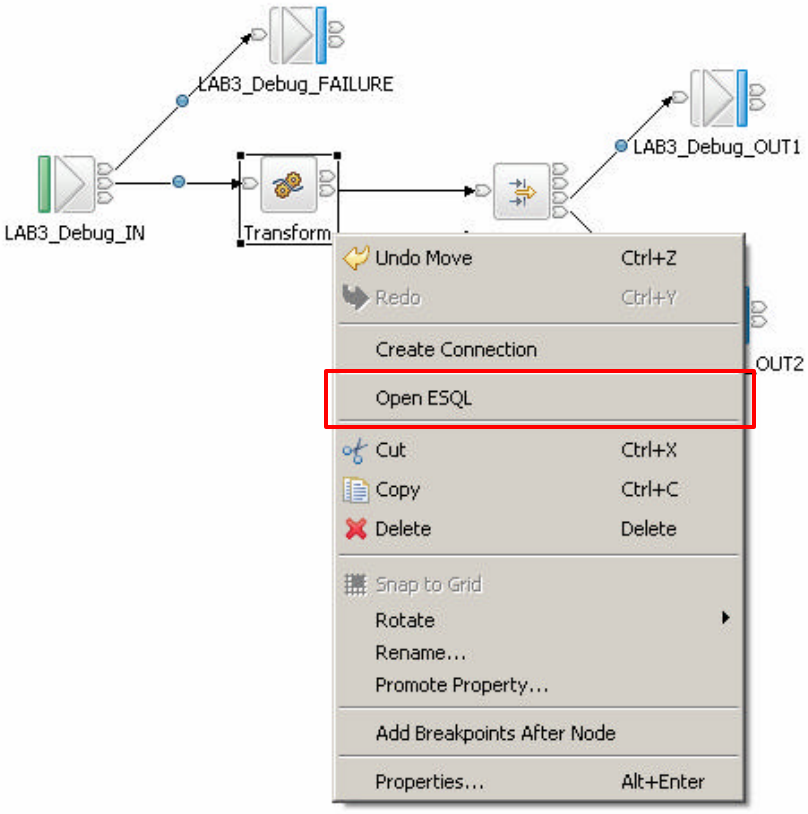


6. Add three further break points on the connections to each of the MQ Output nodes as illustrated below.

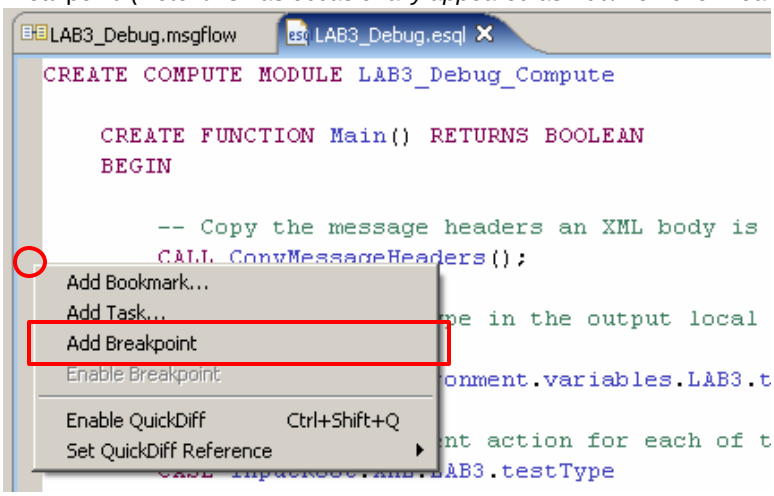


3.2 Setting the ESQL Break Points

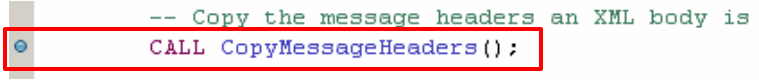
7. To add an ESQL break point first open the ESQL file by right clicking on the Transform compute node and select Open ESQL.



8. To add a break point in the ESQL right click on the grey margin left of the "CALL CopyMessageHeaders();" line. From the drop down menu that appears select Add Breakpoint. *(Note: this has occasionally appeared as Add/Remove Breakpoint)*

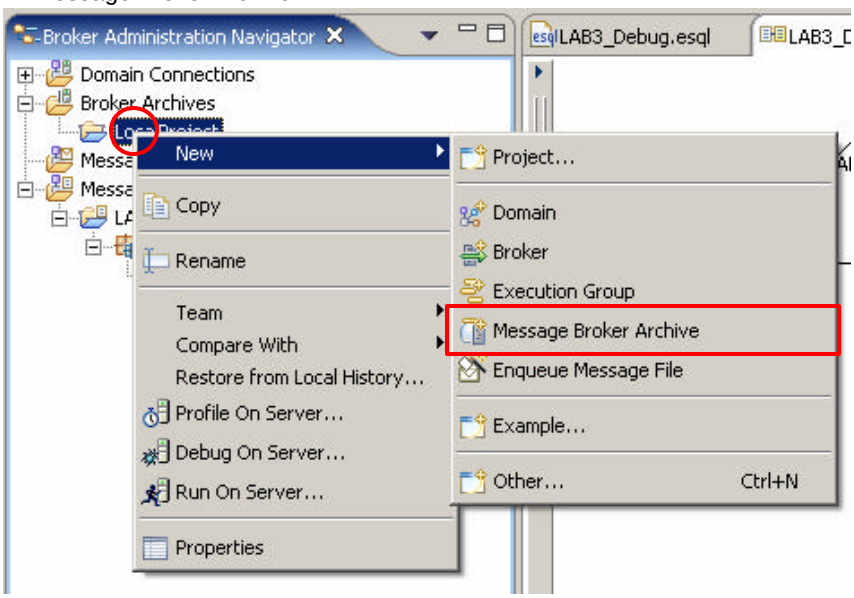


The ESQL break point should then appear for this line.

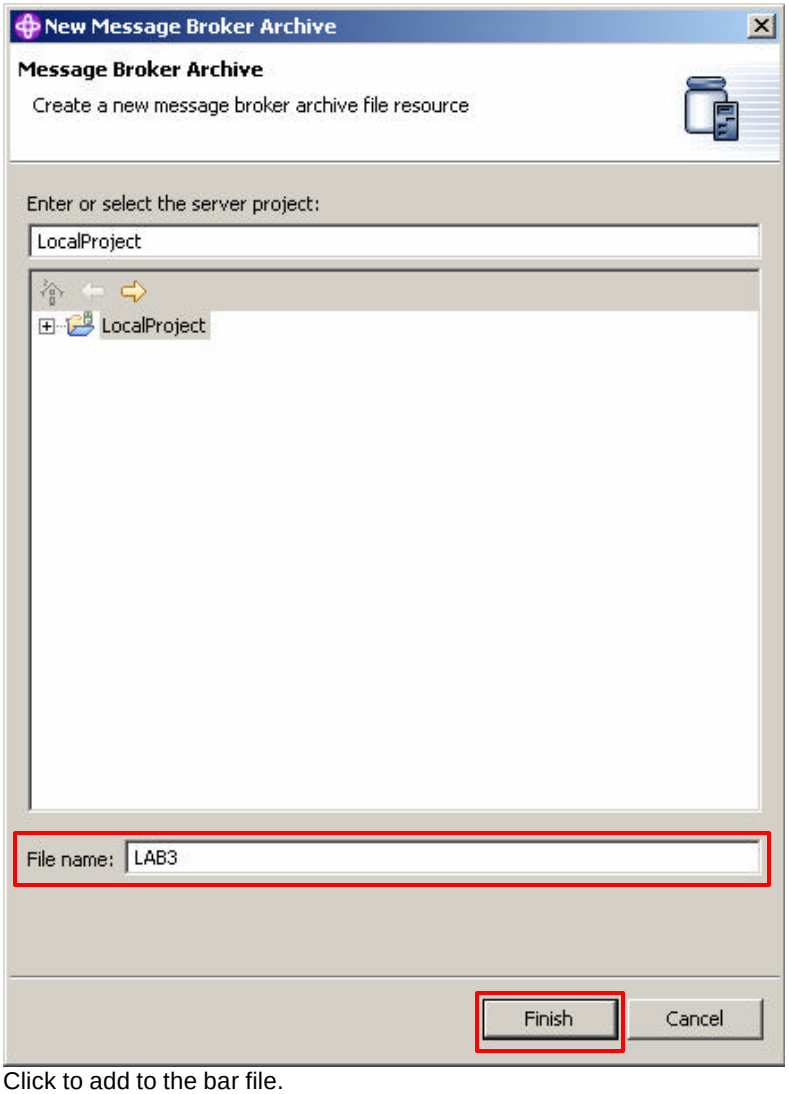


4. Deploying the flow

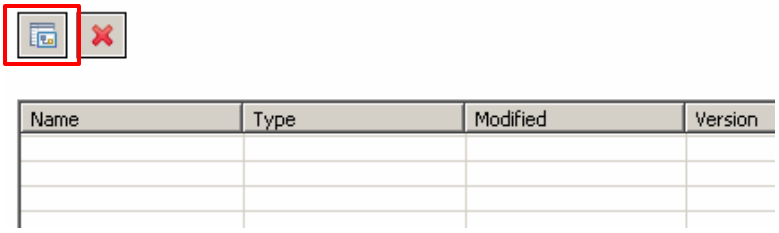
9. Within the Broker Administration perspective right click on the LocalProject and select New → Message Broker Archive



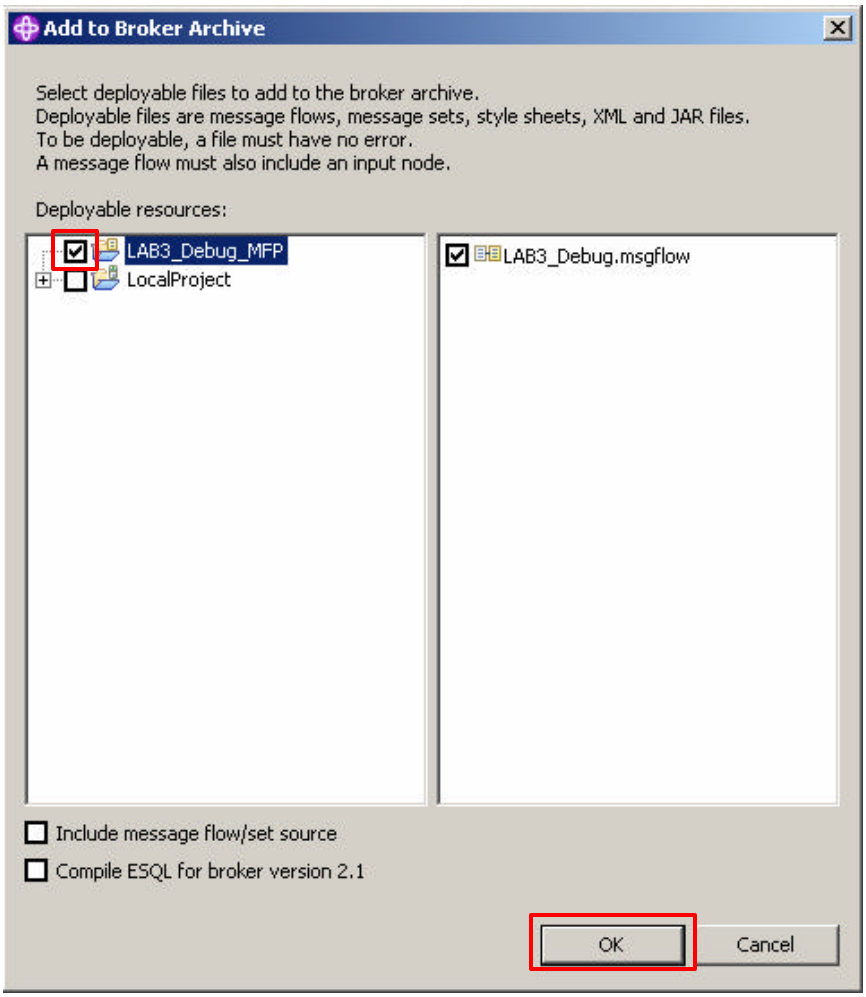
10. Enter the name LAB3 for the bar file and click Finish.



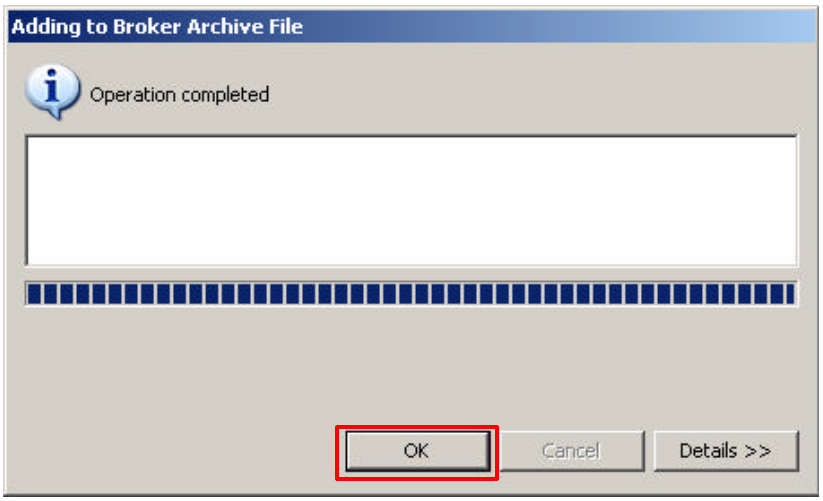
11. Click to add to the bar file.



12. Select the contents of the LAB3_Debug_MFP project to add the message flow, then click OK.

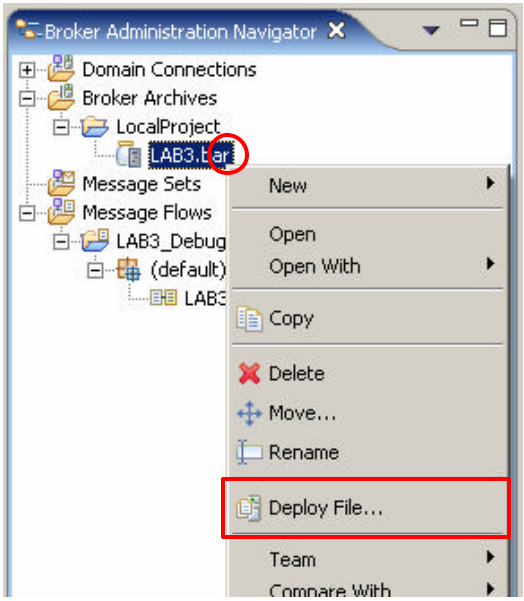


13. Select OK once the files have been added.

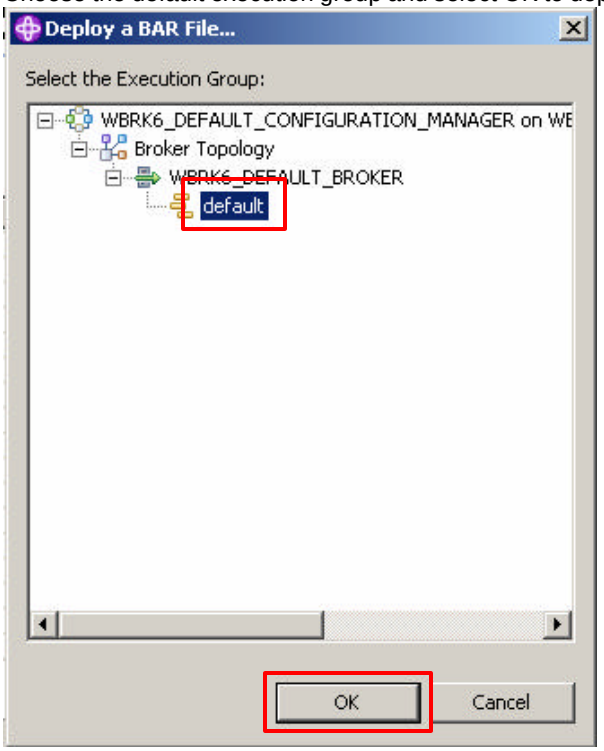


Save the bar file (Ctrl+S)

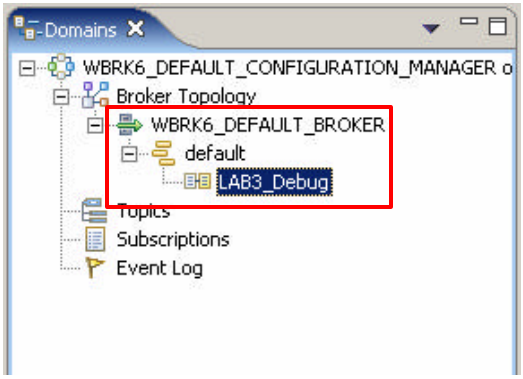
14. Deploy the bar file by right clicking on the bar file and selecting Deploy File… from the drop down list.



15. Choose the default execution group and select OK to deploy.



16. Once the deployment has completed the default execution group should show LAB3_Debug running within it.

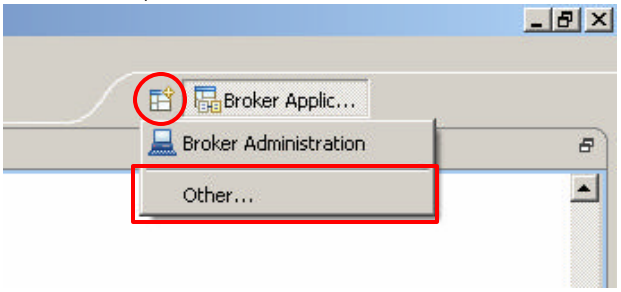


Check the Event log for the following messages.

Message	Source
BIP4040I	WBRK6_DEFAULT_BROKER
BIP2056I	WBRK6_DEFAULT_BROKER

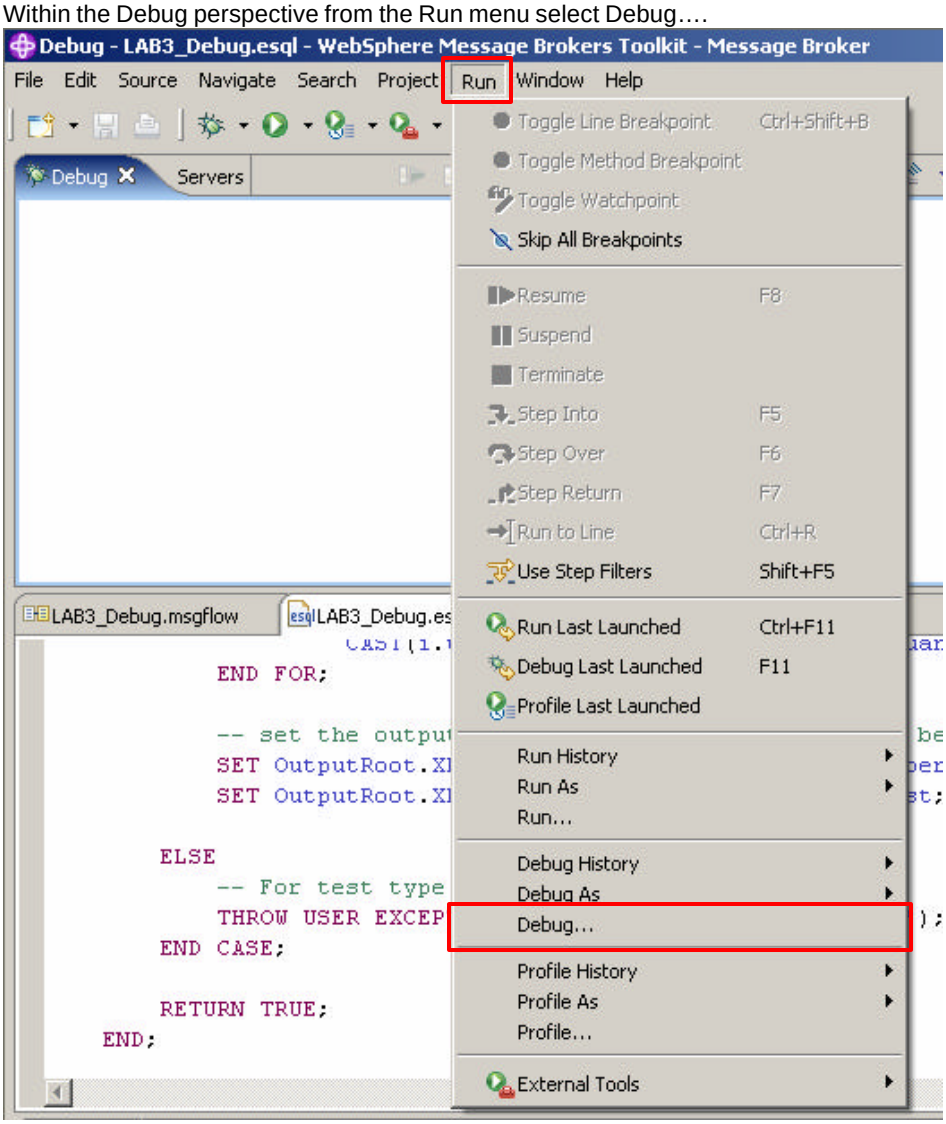
5. Connecting to the Broker

17. Connecting to the broker requires the debug perspective. This is available under the button circled below, select Other.

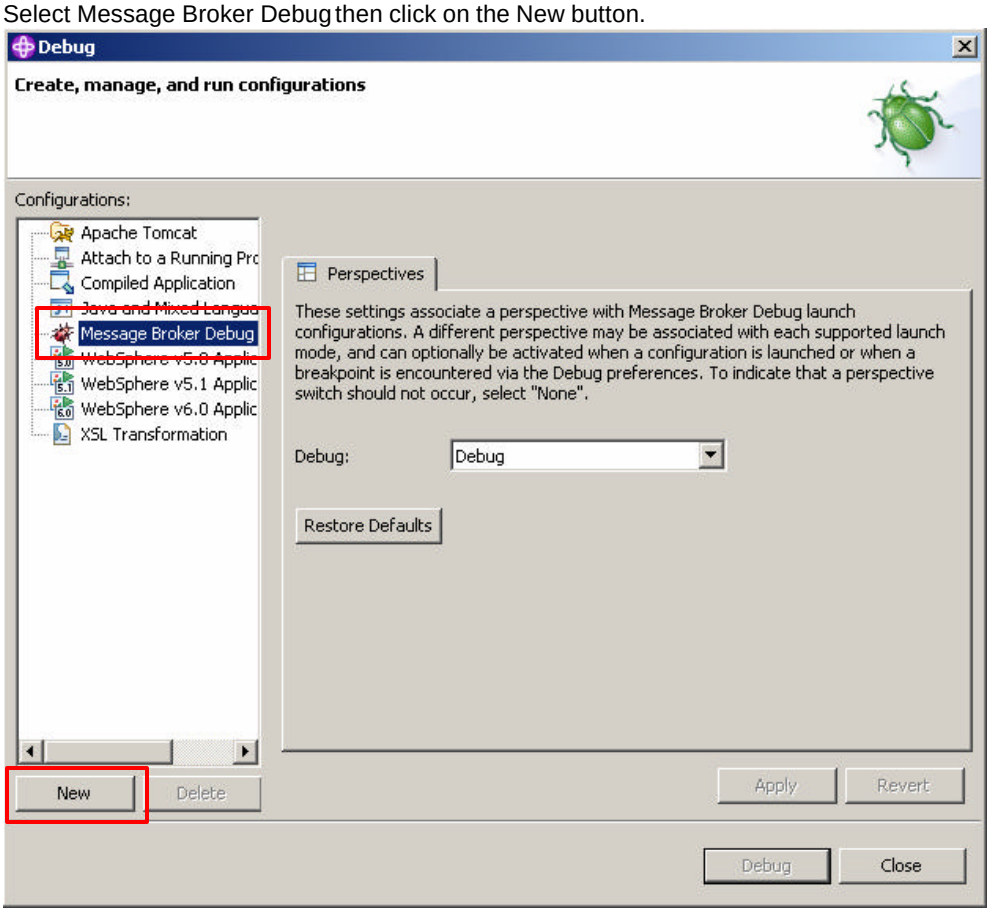


18. From the Select Perspective dialog choose Debug and click OK.

19. Within the Debug perspective from the Run menu select Debug….

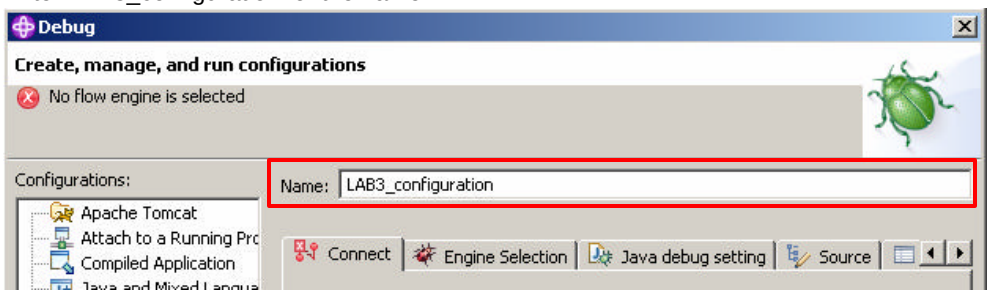


20. Select Message Broker Debug then click on the New button.

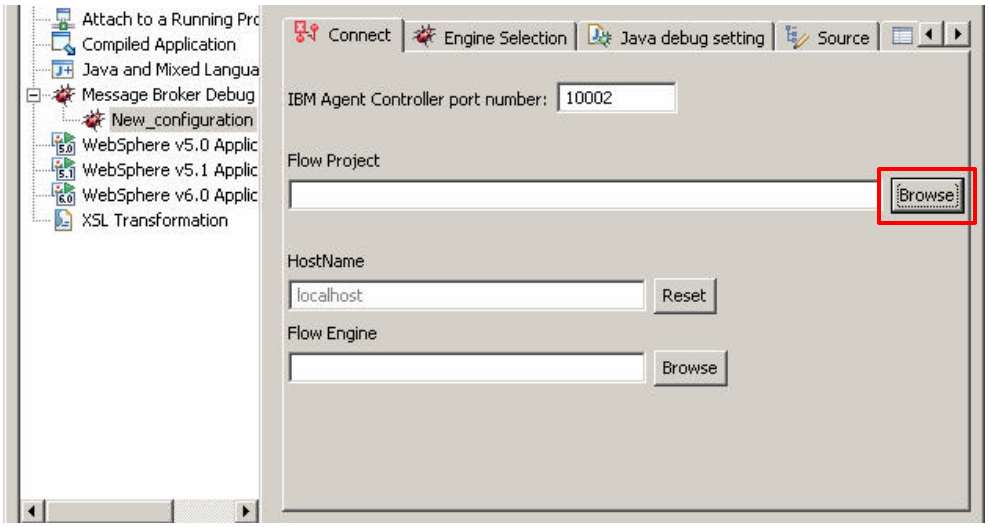


21. Fill in the details for this configuration.

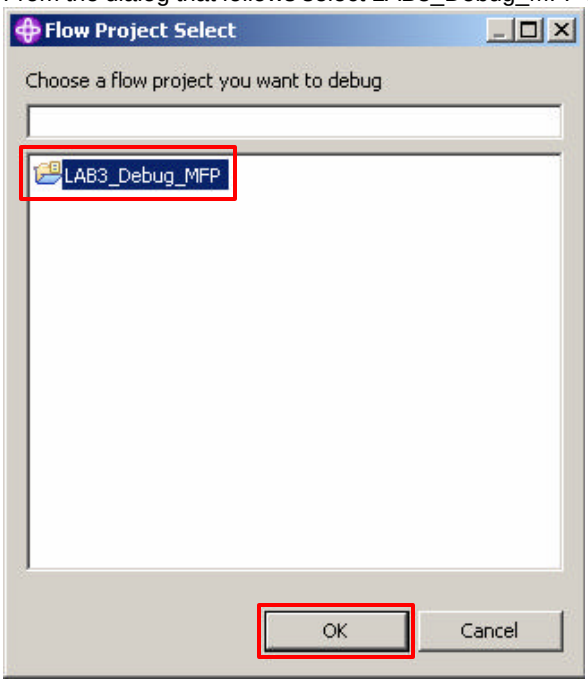
Enter LAB3_configuration for the name.



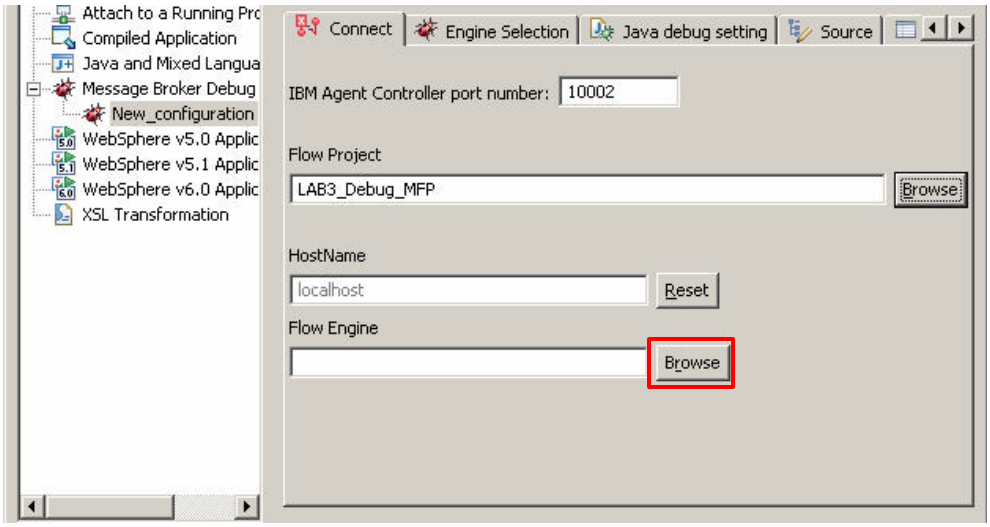
22. Select Browse to use the content assist for the Flow Project to be debugged.



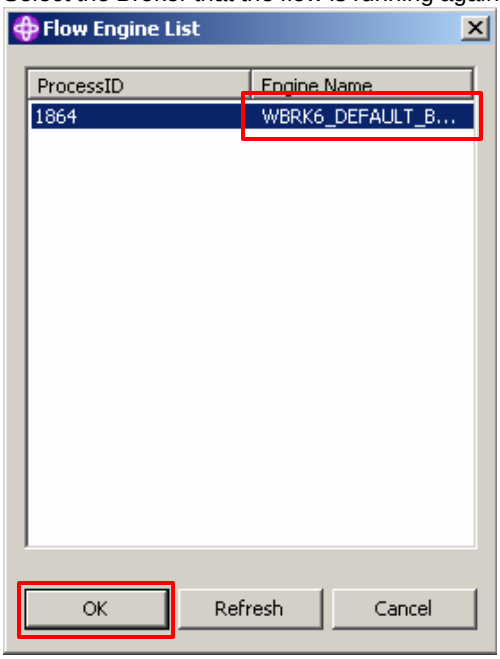
23. From the dialog that follows select LAB3_Debug_MFP and then click OK.



24. Select Browse this time to use the content assist for the Flow Engine to be debugged.

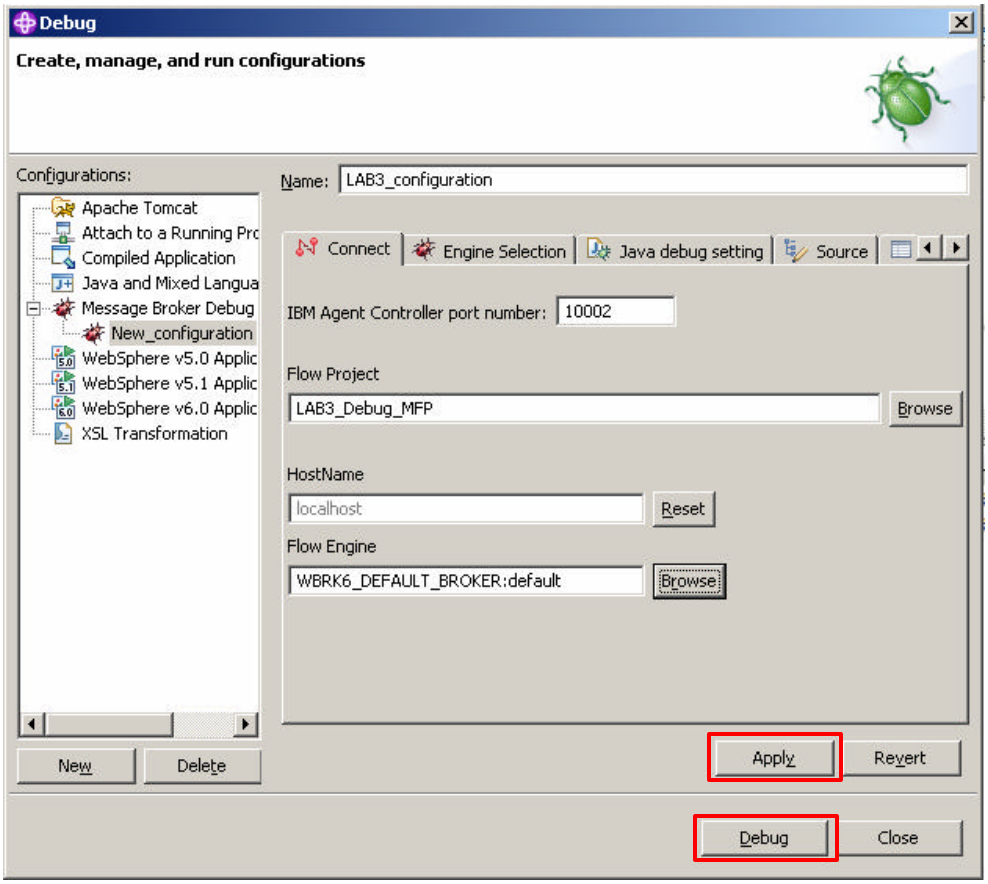


25. Select the Broker that the flow is running against and click OK.

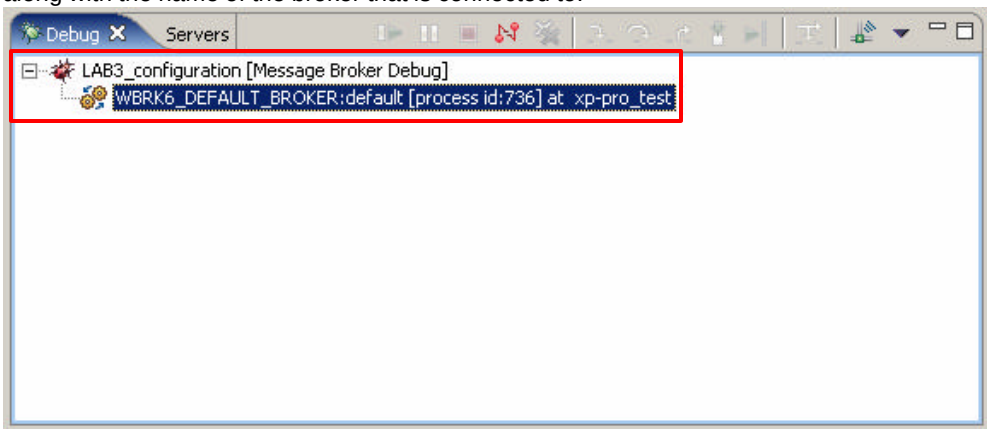


Note: The ProcessID will vary.

26. With all the settings made click Apply to save this configuration and then Debug to start the debug session.



The name of the Debug session that has just been created should appear in the Debug view along with the name of the broker that is connected to.

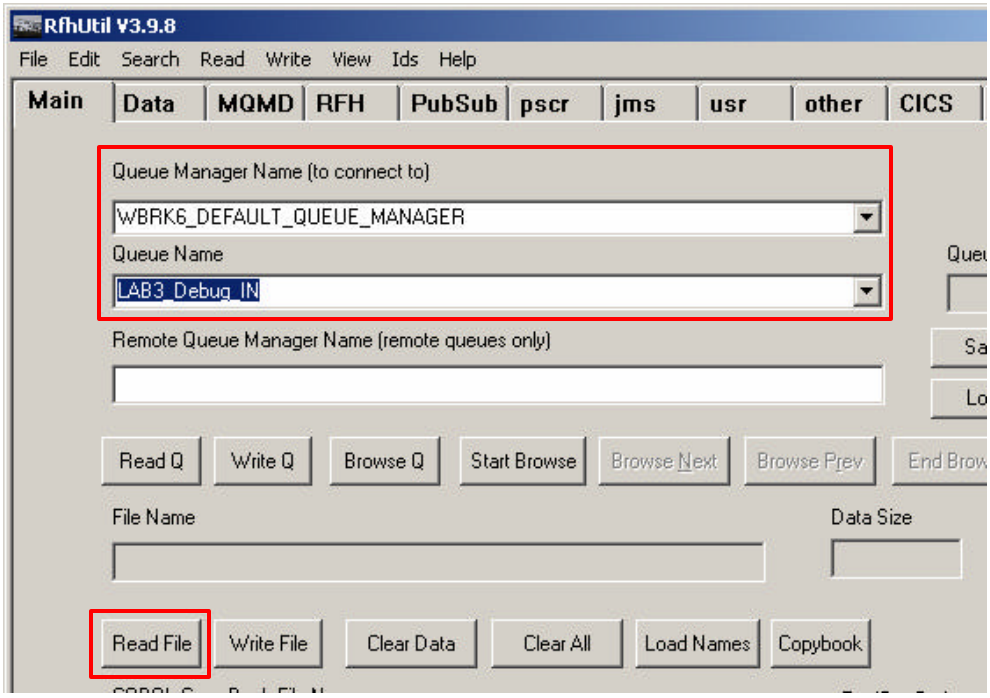


The session is ready to debug the flow.

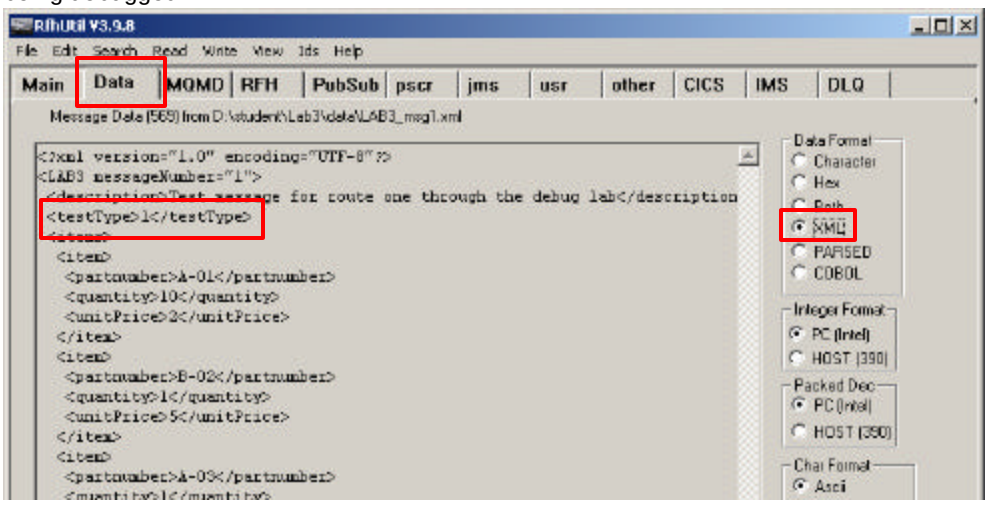
6. Debugging the first route through the Message Flow

27. With RFHUTIL (Start by clicking RFHUtil on the desktop) connect to the queue manager and the input queue LAB3_Debug_IN used by the message flow.

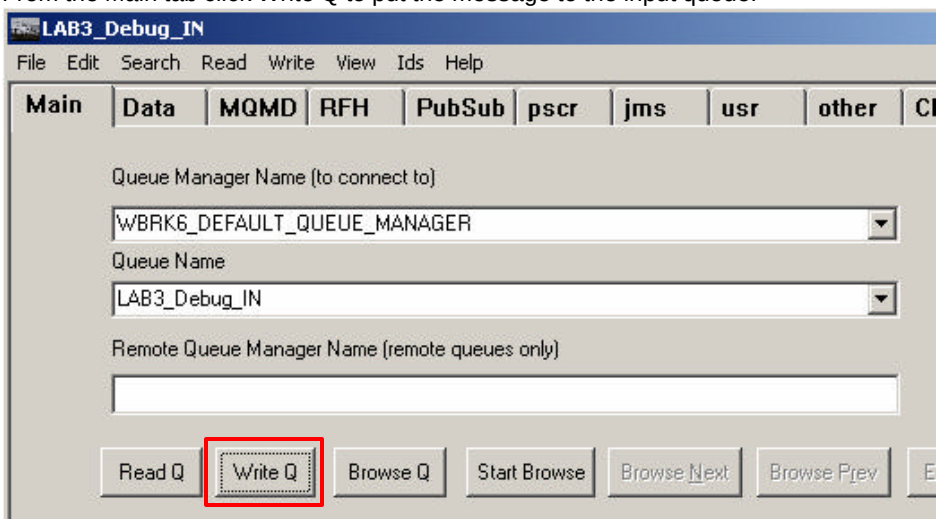
    Click read file and open C:\student\Lab3\data\LAB3_msg1.xml



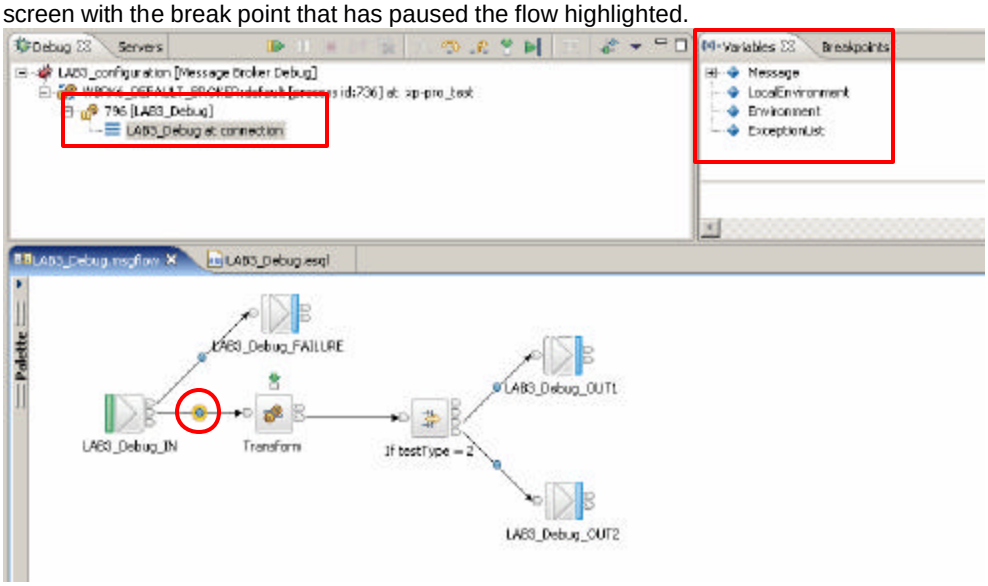
28. Switch to the Data tab and using the XML data format observe that the message contains testType=1. The testType element is used to control the routes through the message flow being debugged.



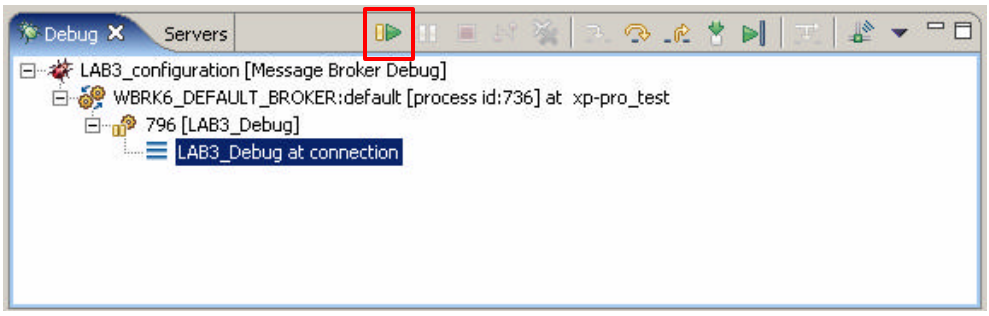
29. From the Main tab click Write Q to put the message to the input queue.



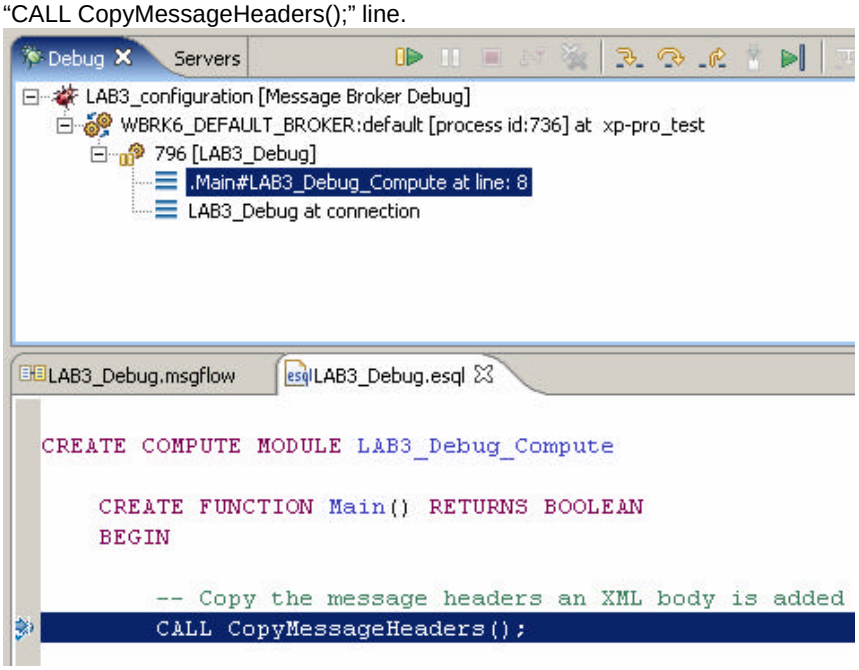
30. Return to the Message Broker Toolkit. The Debug view shows the message flow LAB3_Debug as paused. The Variables view now shows Messages, Local Environment, Environment and the Exception List. The Message flow is displayed in the lower half of the screen with the break point that has paused the flow highlighted.



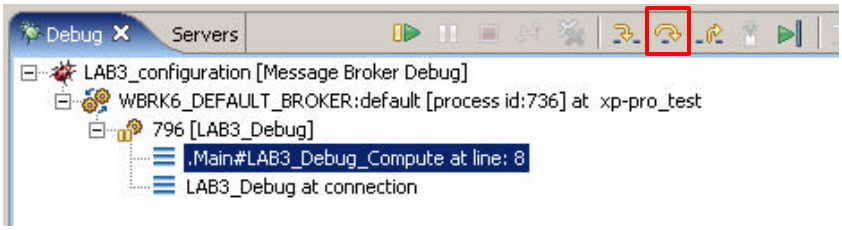
31. Select Resume.



32. The lower half of the screen now changes to the ESQL view showing the flow paused on the "CALL CopyMessageHeaders();" line.

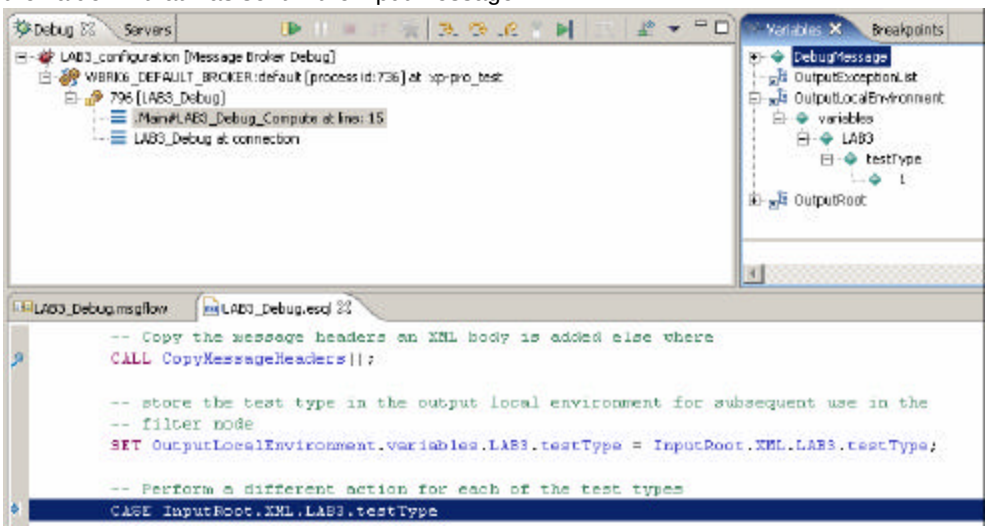


33. Use the "Step Over" button to step over the"CALL CopyMessageHeaders();" line



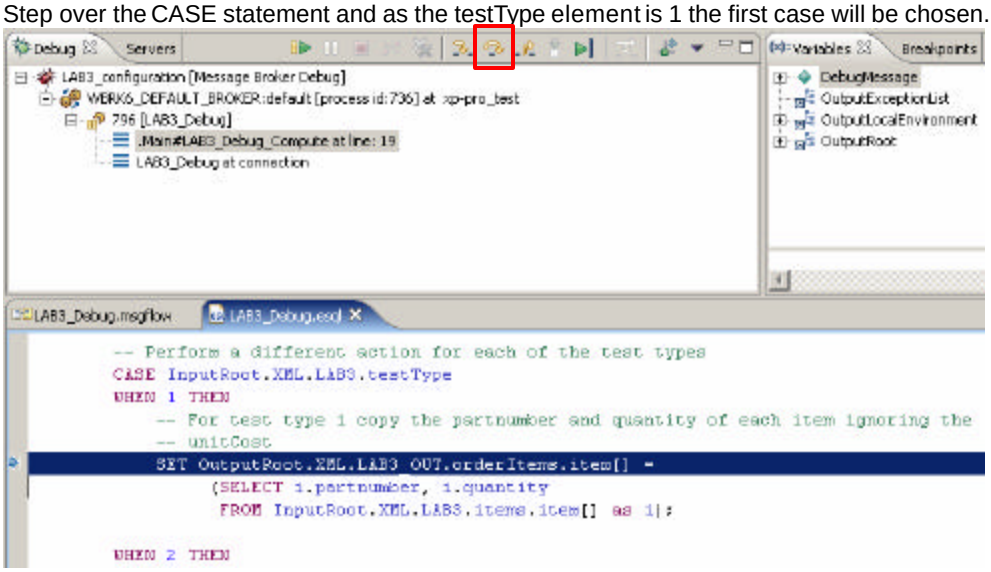
press the "Step Over" button again to step beyond the"SET OutputLocalEnvironment…" line.

34. With the execution now paused on the CASE statement in the variables view expand the OutputLocalEnvironment, variables, LAB3, testType tree to observe that this has been set by the value "1" that was sent in the input message.



This value will be used later to control the output terminal of the filter node.

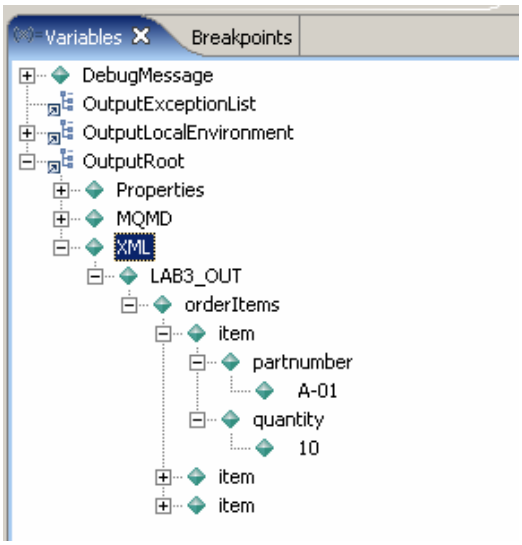
35. Step over the CASE statement and as the testType element is 1 the first case will be chosen.



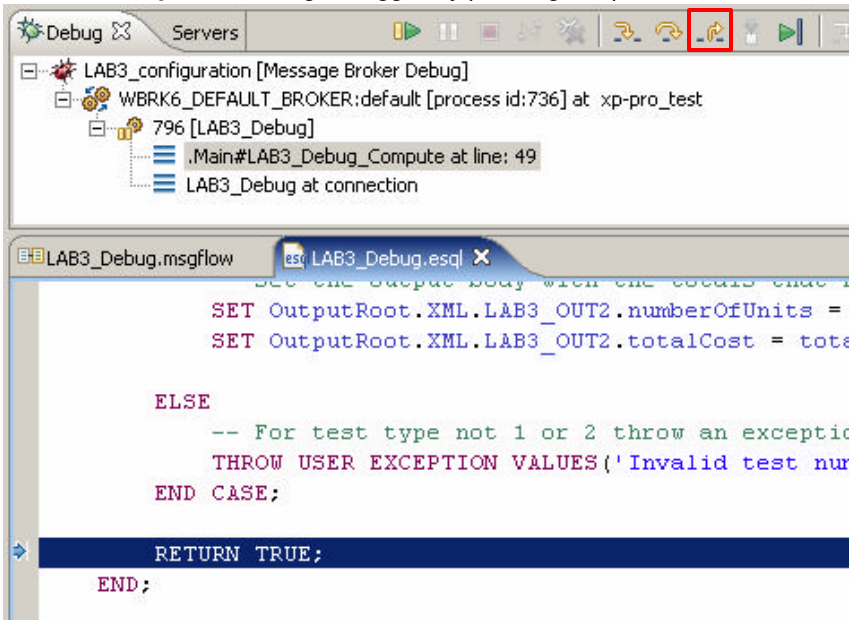
36. Step over the statement which is setting the OutputRoot.XML…

Return to the variables view this time expand the OutputRoot to reveal the message that will be propagated. The screen shot that follows shows the message expanded with the full

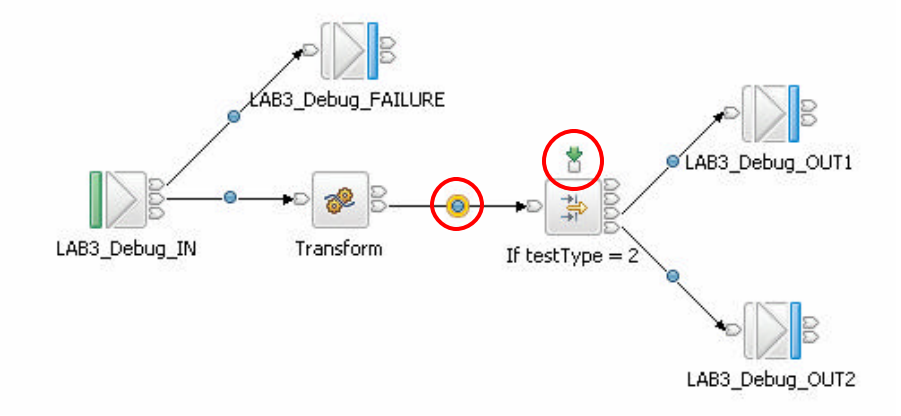
expansion of the first “item” element.



37. Leave the ESQL that is being debugged by pressing “Step Return”button.

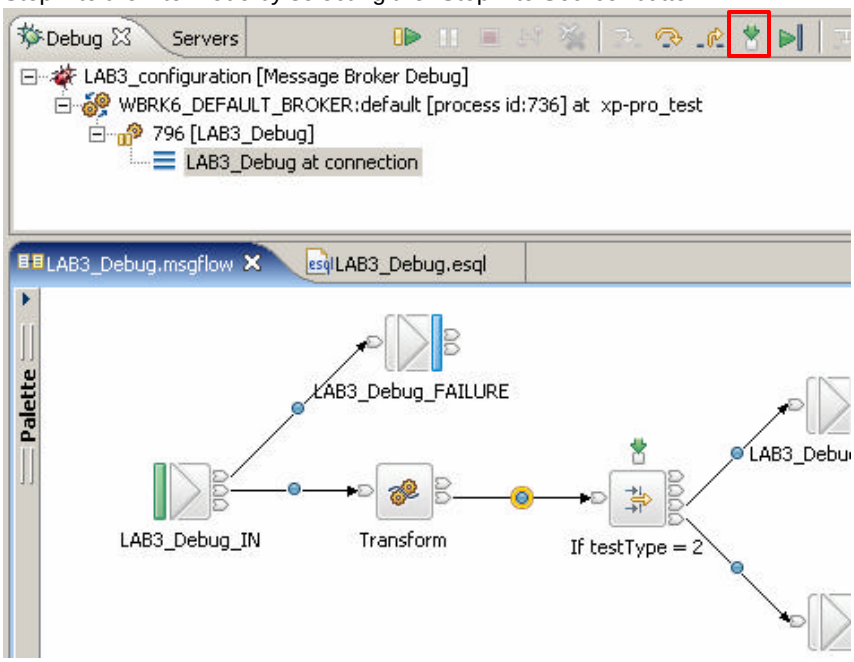


Notice how this takes the view back to the message flow and breaks just beyond the “Transform” Compute node even through a break point was not specified for this connection.



The symbol above the “If testType = 2” filter node identifies this as a node that can be stepped into.

38. Step into the filter node by selecting the “Step into Source” button.

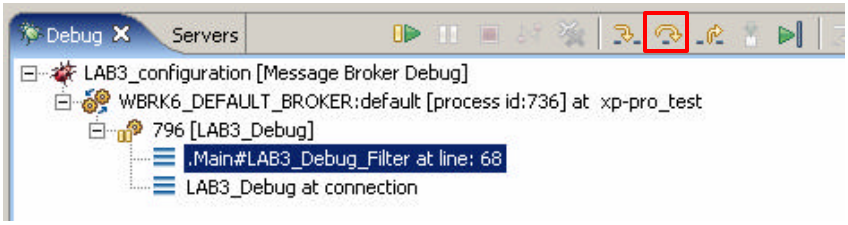


39. The focus returns to the ESQL this time for the filter node source.

```
CREATE FILTER MODULE LAB3_Debug_Filter
    CREATE FUNCTION Main() RETURNS BOOLEAN
    BEGIN
        -- Filter to the "True" terminal for test type 2 only
        IF LocalEnvironment.variables.LAB3.testType = 2 THEN
            RETURN TRUE;
        END IF;

        RETURN FALSE;
    END;
END MODULE;
```

40. Step Over this function and observe that for the testType of 1 that was copied into the Local Environment and that this function returns FALSE.

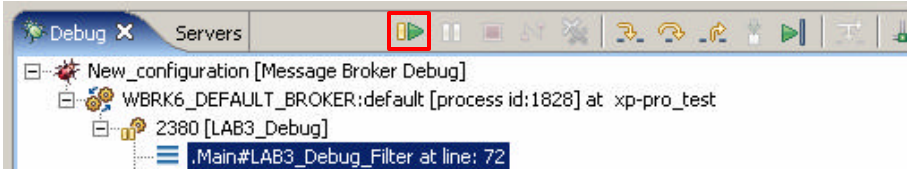


The following shows the flow paused on the “RETURN FALSE;” line.

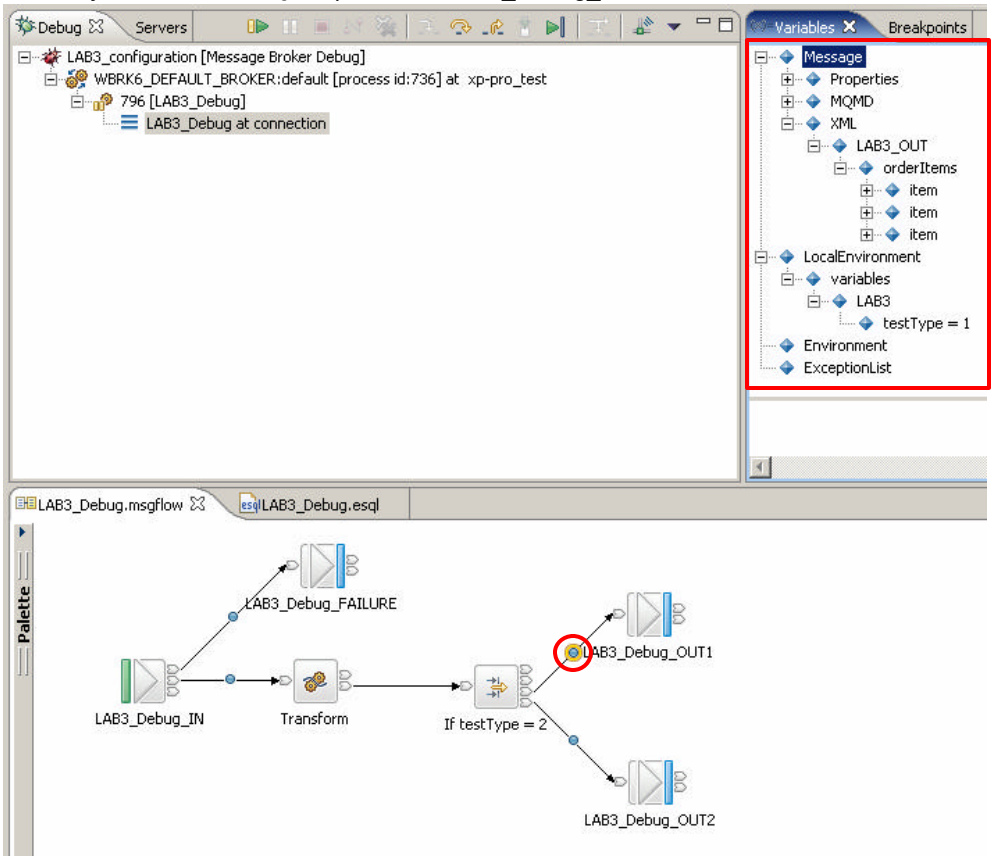
```
CREATE FILTER MODULE LAB3_Debug_Filter
    CREATE FUNCTION Main() RETURNS BOOLEAN
    BEGIN
        -- Filter to the "True" terminal for test type 2 only
        IF LocalEnvironment.variables.LAB3.testType = 2 THEN
            RETURN TRUE;
        END IF;

        RETURN FALSE;
    END;
END MODULE;
```

Press Resume to continue.

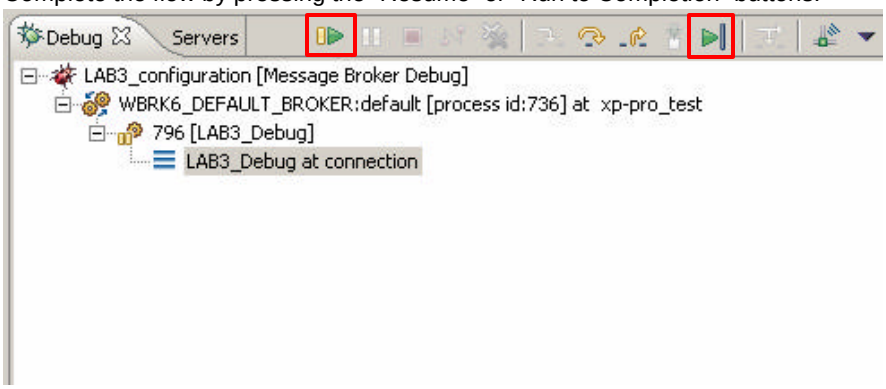


41. When execution returns to the message flow the final break point for this test message halts the flow just before the MQ Output node “LAB3_Debug_OUT1”.



Once again you may wish to explore the Variables view to see the Message that is about to be written.

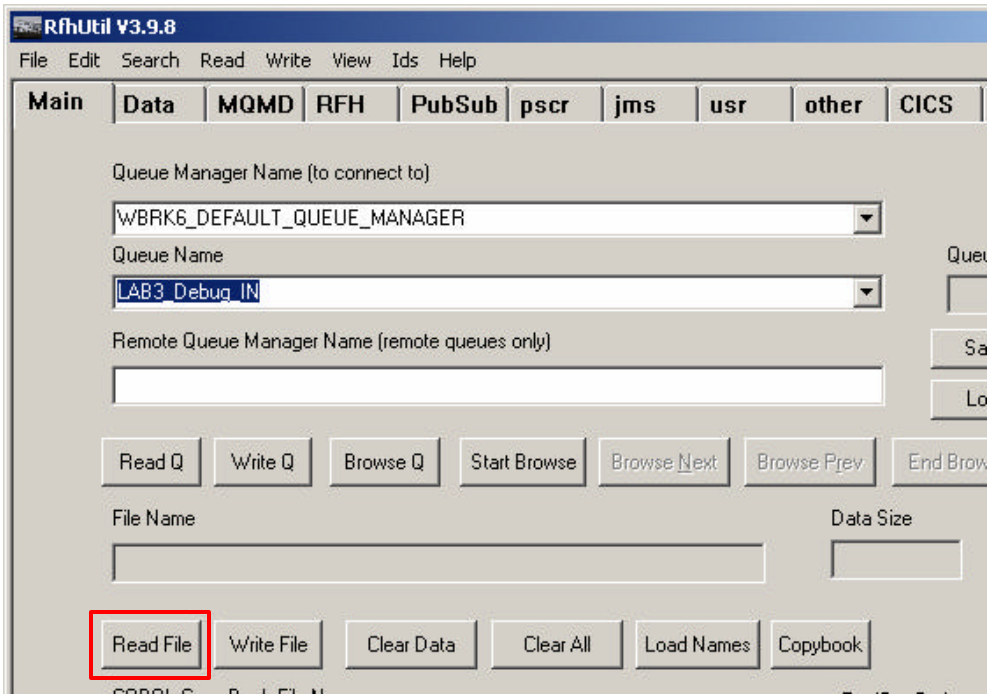
Complete the flow by pressing the “Resume” or “Run to Completion” buttons.



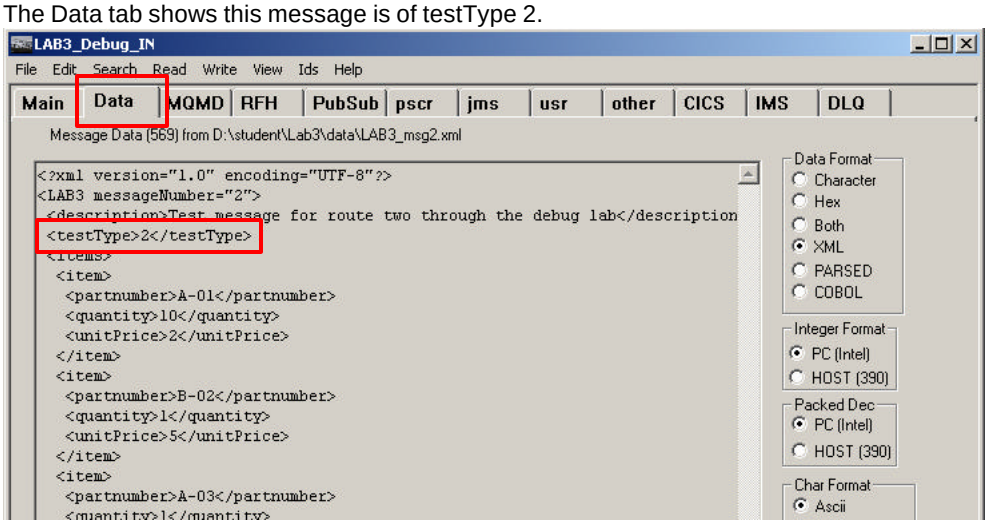
You may wish to confirm that the message is in fact on the output queue using MQ Explorer or RFHUTIL for LAB3_Debug_OUT1.

7. Debugging the second route through the Message Flow

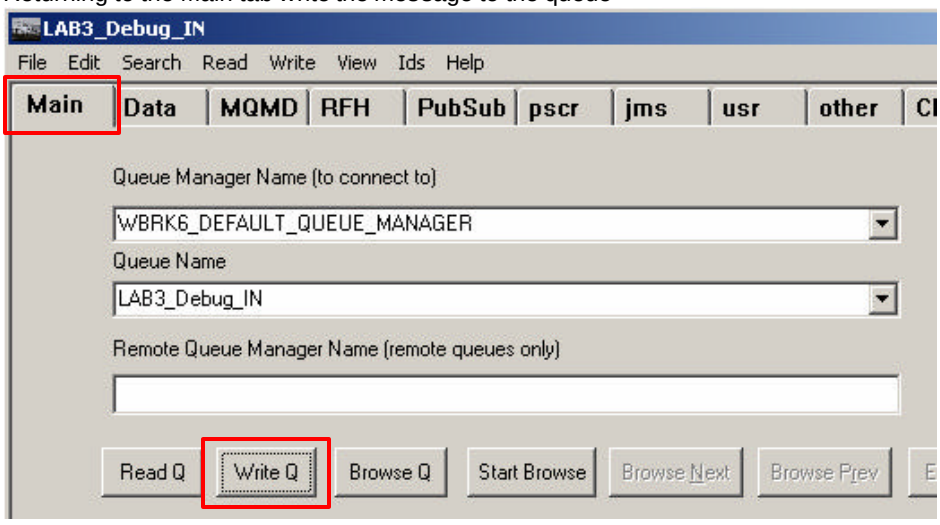
42. Return to RFHUTIL to read the next test message C:\student\Lab3\data\LAB3_msg2.xml



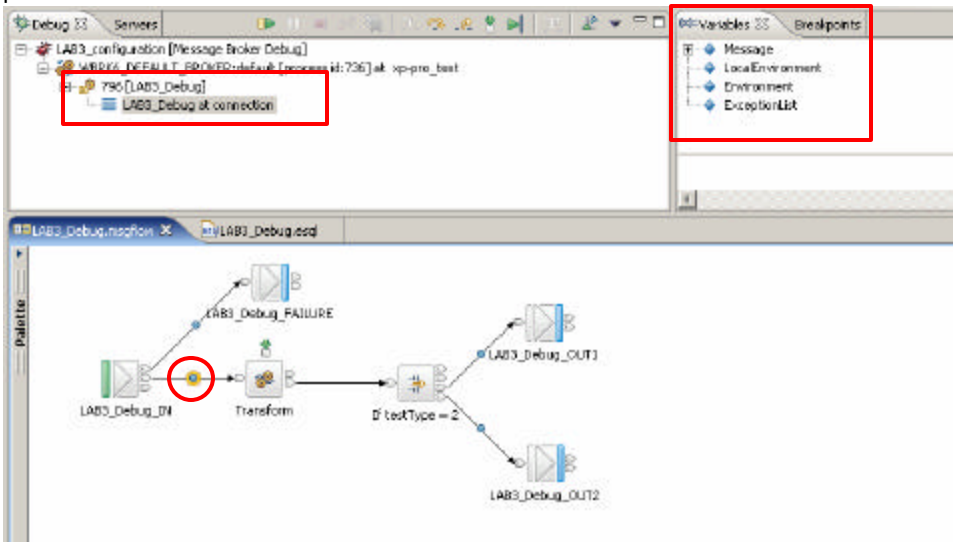
43. The Data tab shows this message is of testType 2.



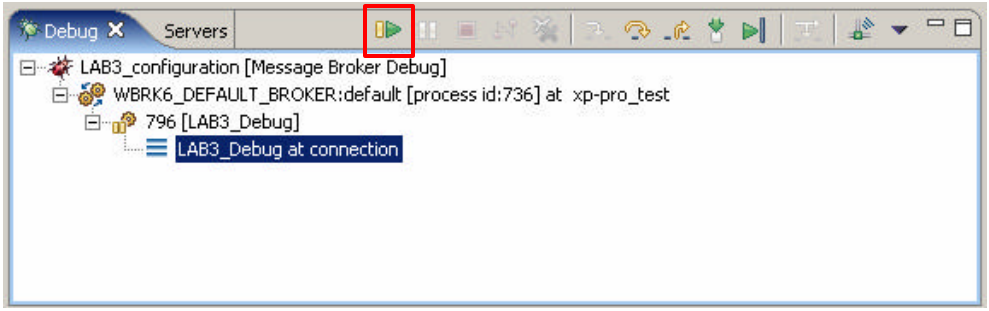
44. Returning to the Main tab write the message to the queue



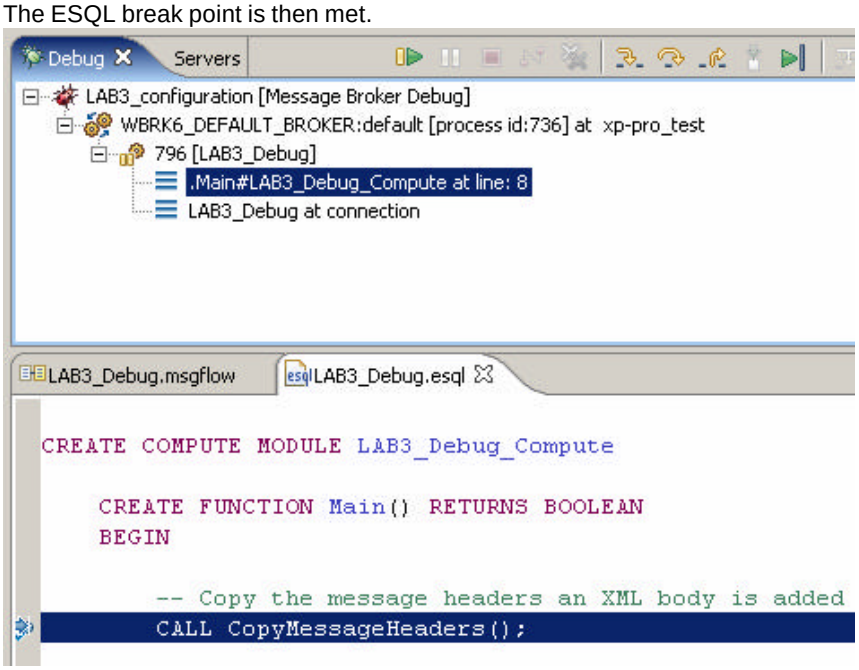
45. Return to the Message Broker Toolkit as before the message is paused by the first break point.



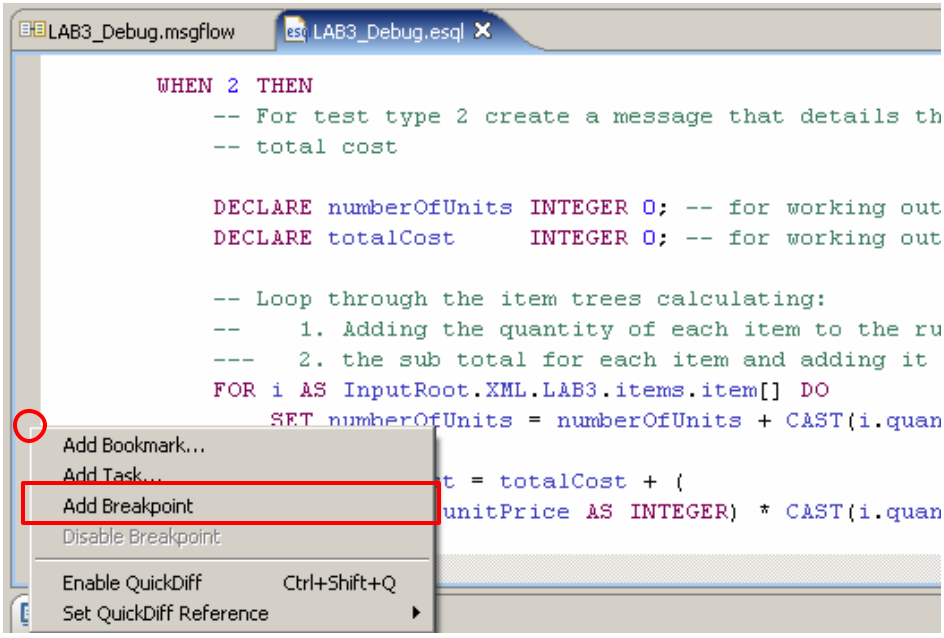
46. Press Resume.



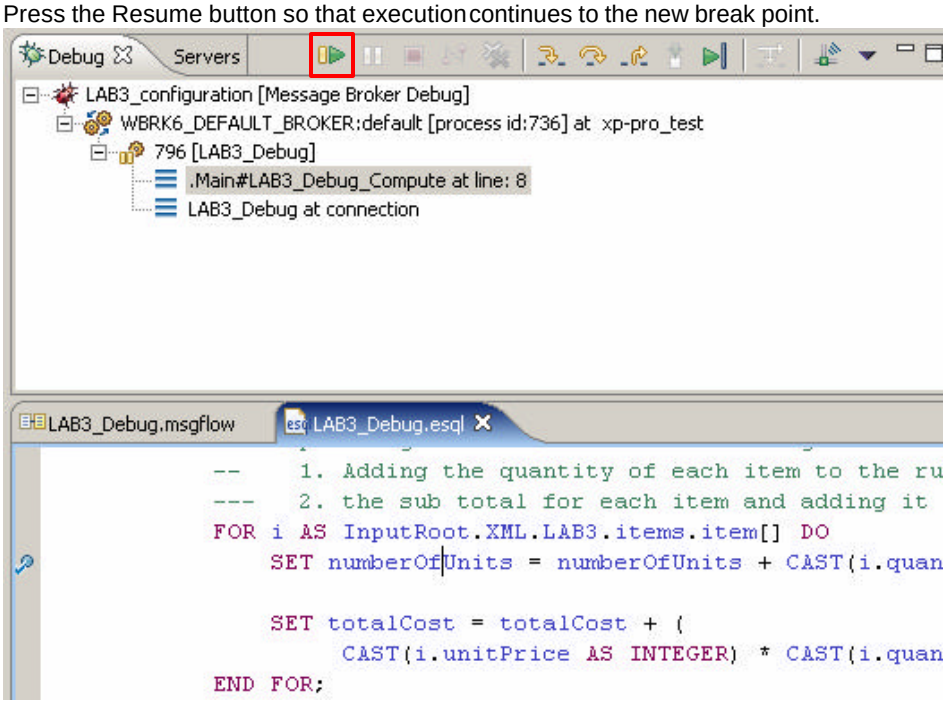
47. The ESQL break point is then met.



48. At this stage scroll down the ESQL file into the WHEN statement for when testType = 2 and add a new break point for the first line within the FOR loop the line which begins “SET numberOf…”. Right click to the left of the line (on the grey margin) and select Add Breakpoint.



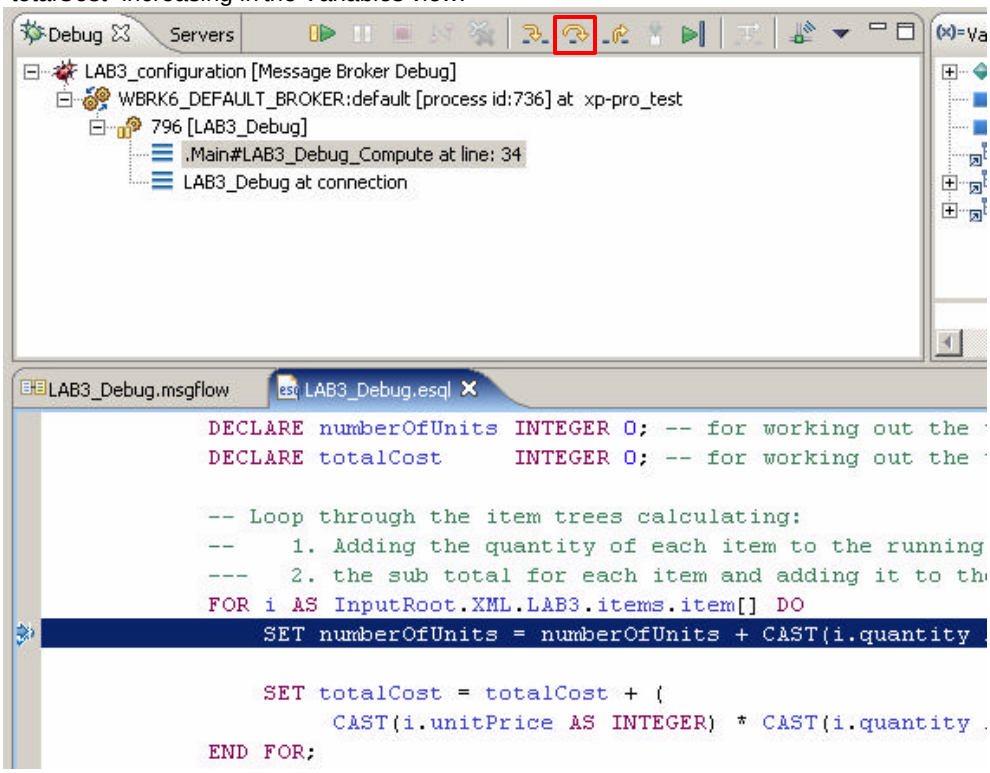
49. Press the Resume button so that execution continues to the new break point.



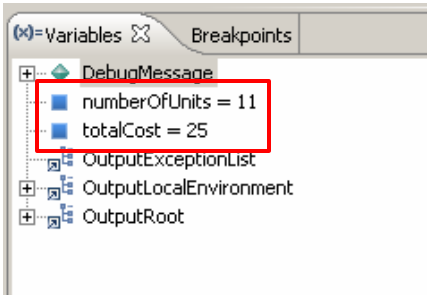
Note: Whilst connected to the debugger active breakpoints have a tick against the blue dot.

50. As the message is of testType 2 this time the previous resume will stop the flow on the break point that has just been added.

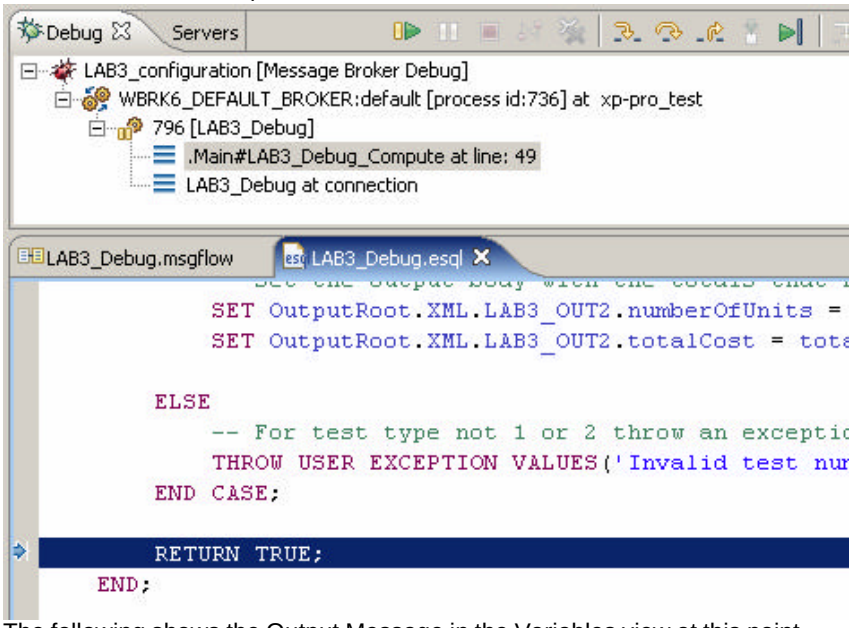
    Step Over the lines of this FOR loop and observe the variables “numberOfUnits” and “totalCost” increasing in the Variables view.



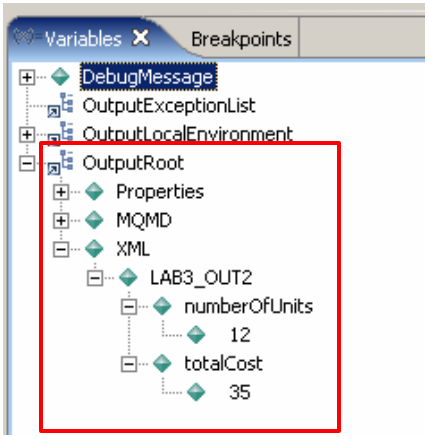
    The following shows the result of four presses of the Step Over button.



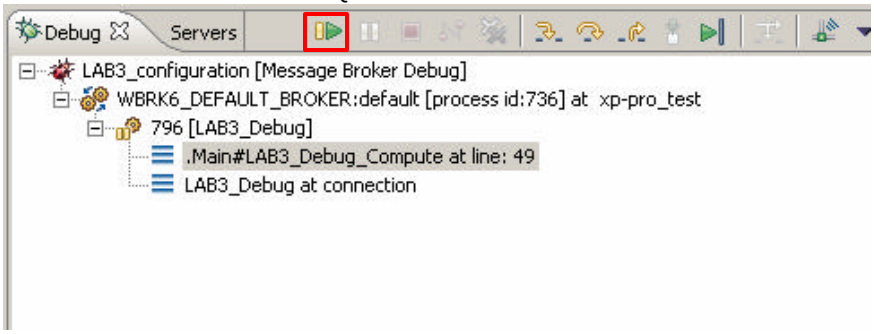
51. By Stepping Over to the “RETURN TRUE;” line (8 times in total) observe the message that has been set in the OutputRoot.



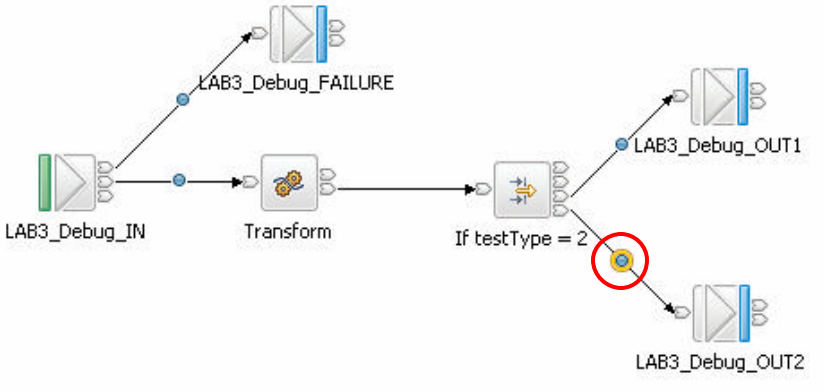
The following shows the Output Message in the Variables view at this point.



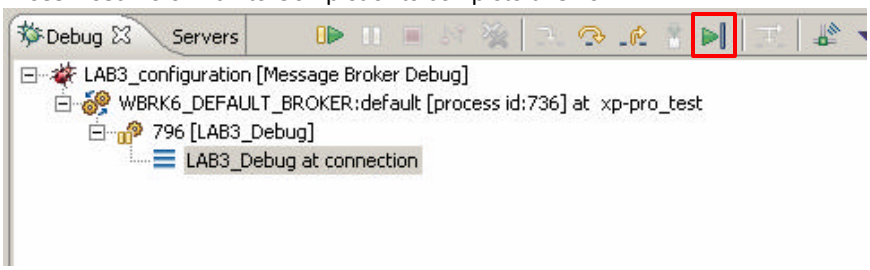
52. Press Resume to leave the ESQL function.



53. This time the execution has been paused by the connection break point set proir to the "LAB3_Debug_OUT2" MQ Output node the filter node has routed to its True terminal.



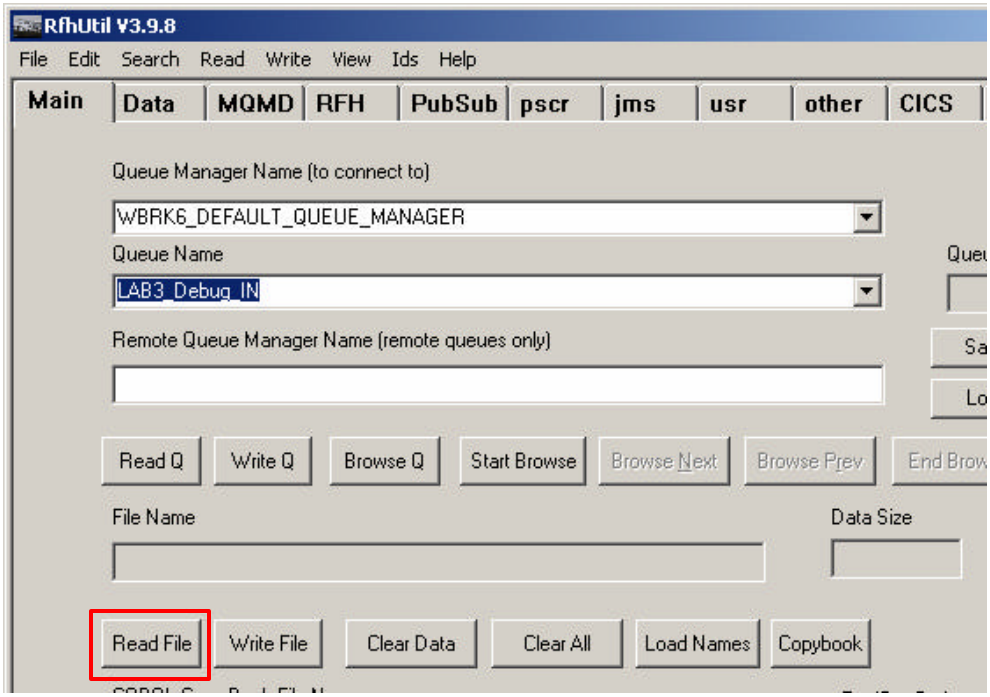
54. Press Resume or Run to Completion to complete this flow.



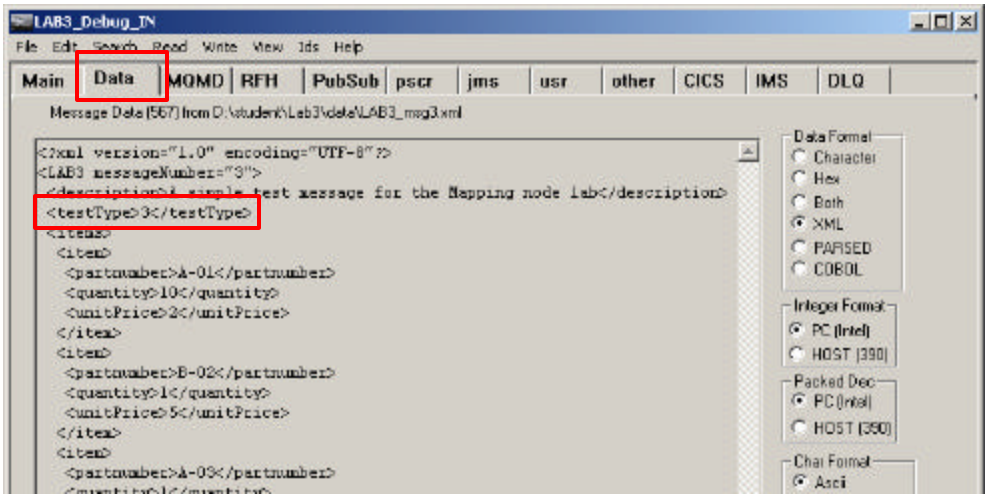
You may wish to confirm that the message is in fact on the output queue using MQ Explorer or RFHUTIL for LAB3_Debug_OUT2.

8. Debugging the third route through the Message Flow

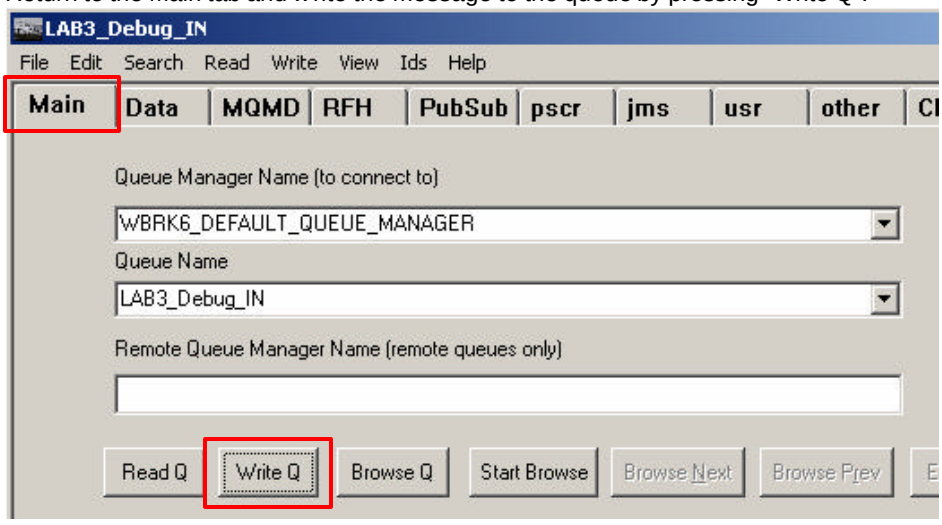
55. Return to RFHUTIL to read the final test message C:\student\Lab3\data\LAB3_msg3.xml



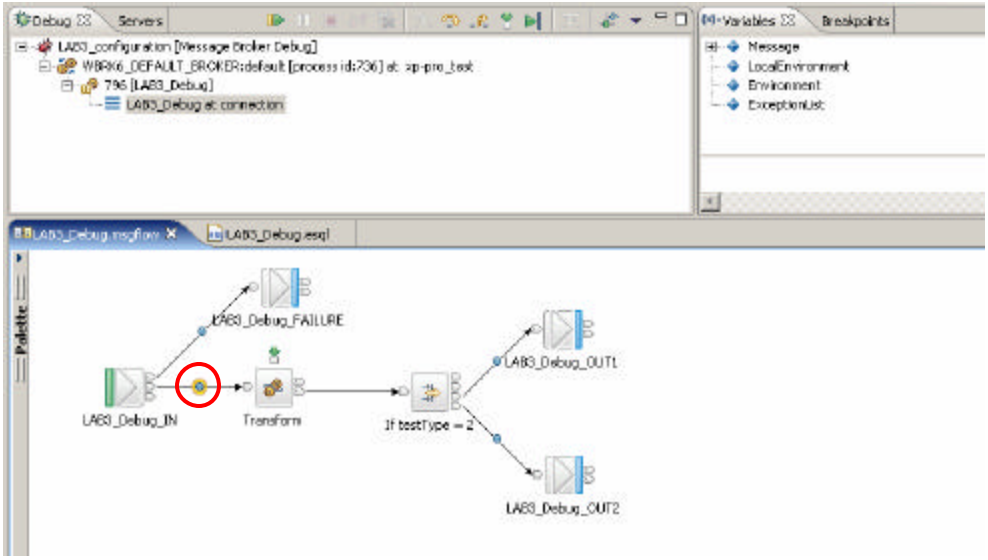
56. The Data tab shows this message is of testType 3.



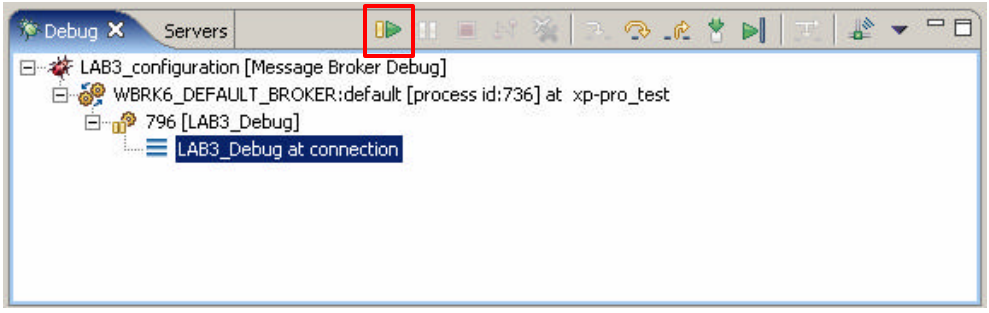
57. Return to the Main tab and write the message to the queue by pressing “Write Q”.



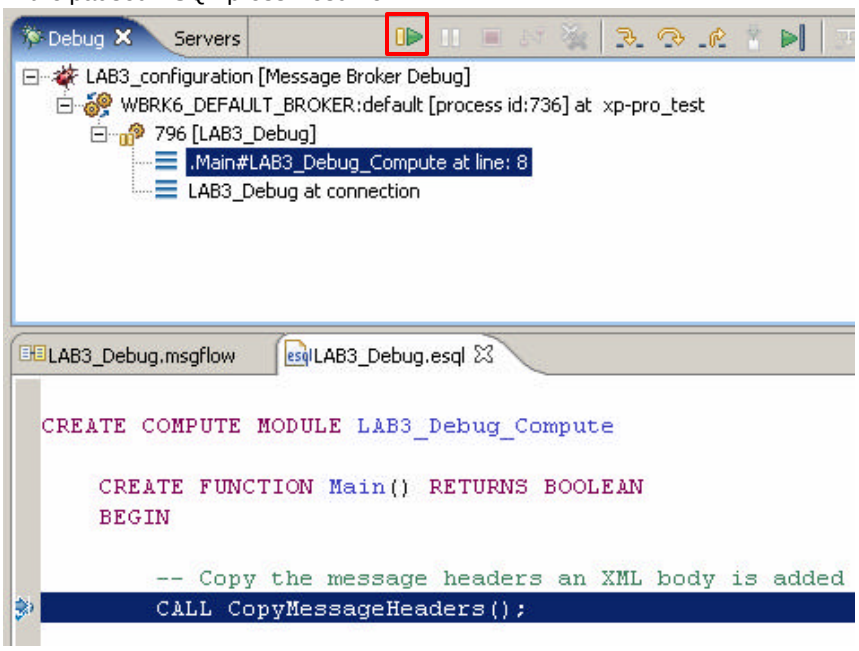
58. Return to the Message Broker Toolkit the message is paused by the first flow break point.



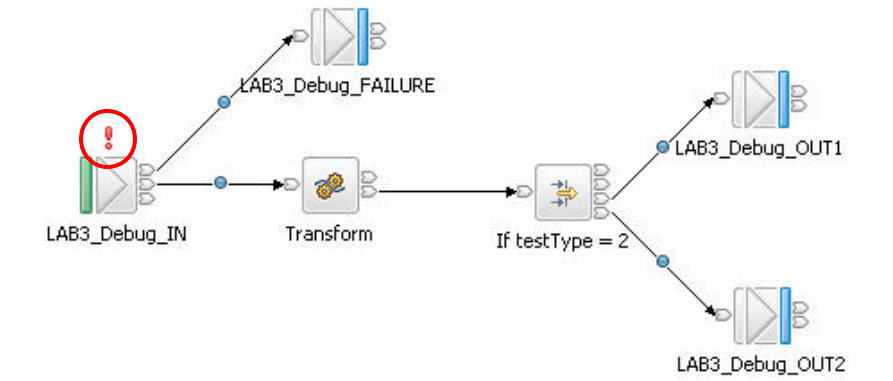
59. Press Resume.



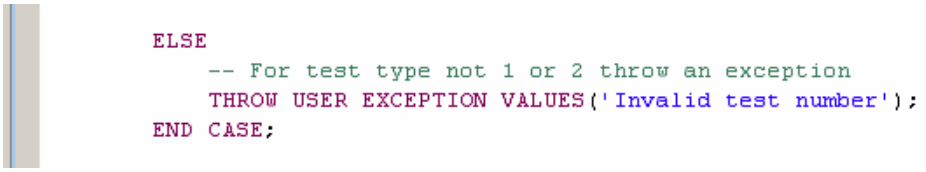
60. In the paused ESQL press Resume.



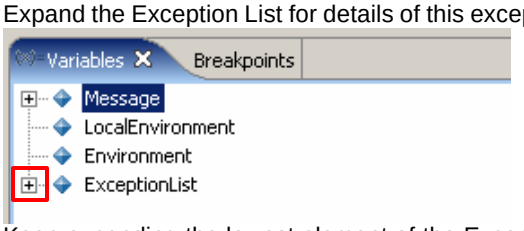
61. This time the message flow view shows that an exception has been caught by the MQ Input node. The exception is identified by an exclamation mark highlighted below.



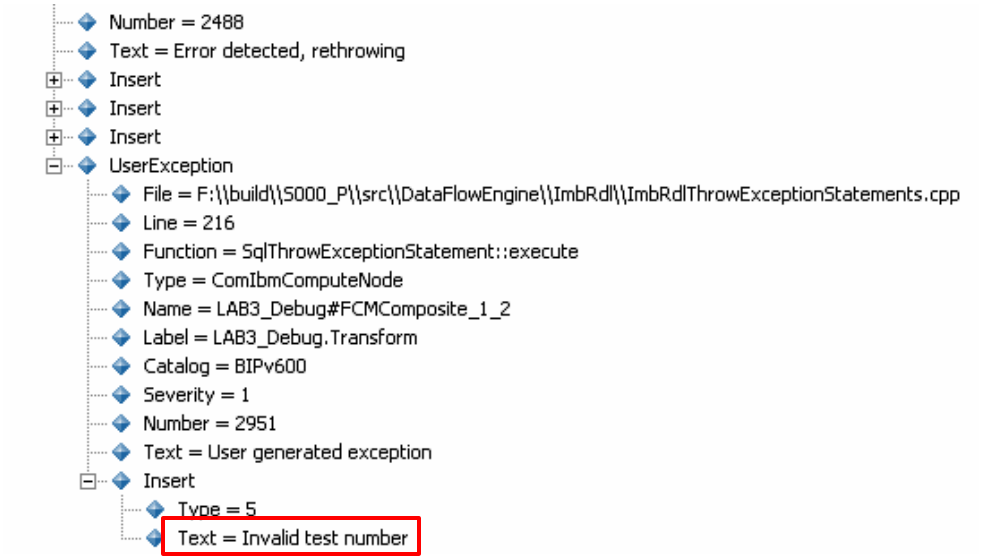
This exception is due to a THROW statement with the ESQL that is executed when testType is any value other than 1 or 2.



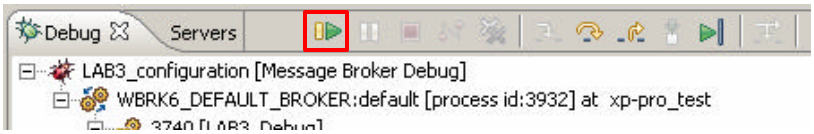
62. Expand the Exception List for details of this exception.



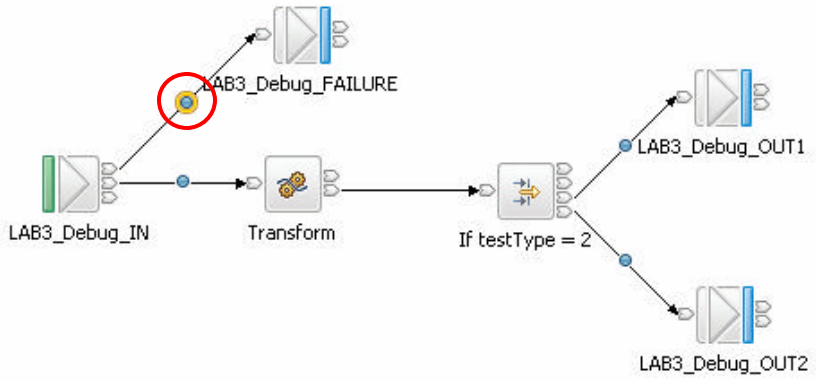
Keep expanding the lowest element of the ExceptionList to eventually reveal the text sent in the user exception.



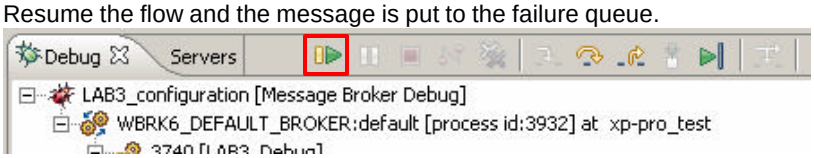
63. Press Resume.



Resuming the flow will return control to the MQ Input node (LAB3_Debug_IN). The exception causes the transaction, under which the MQ message that triggered the flow, to be backed out. As the queue's backout count is set to zero when next got by the MQ Input node it will be sent down the Failure terminal.

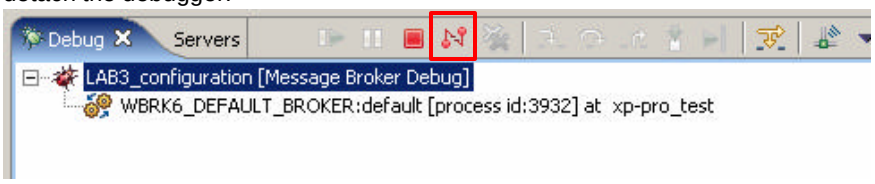


64. Resume the flow and the message is put to the failure queue.

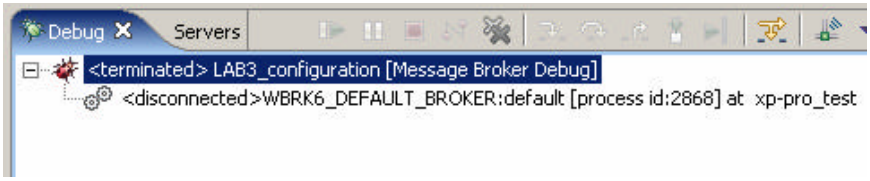


MQ Explorer or RFHUTIL could be used to confirm that the message has been put to LAB3_Debug_FAILURE.

65. Finally select the LAB3_configuration in the Debug view and press the "Disconnect" button to detach the debugger.



The following shows the debug session terminated.



9. Conclusion

This concludes the Debug exercise. Three routes through a simple flow have been explored including the processing of an exception. You will have observed how there is now only one debug view that covers debug for both Message Flows and ESQL.

Appendix 1 – Installation instructions

Creating the MQ queues

1. Run the command "**Lab3mq.bat**" from the install directory:

```
C:\student\Lab3\install>Lab3mq.bat

C:\student\Lab3\install>runmqsc WBRK6_DEFAULT_QUEUE_MANAGER  0<Lab3_Queues.mqsc

5724-B41 (C) Copyright IBM Corp. 1994, 2002.  ALL RIGHTS RESERVED.
Starting MQSC for queue manager WBRK6_DEFAULT_QUEUE_MANAGER.


     :
     : * Run MQSC script for the Debug exercise MQ queues
     :
    1 : define qlocal('LAB3_Debug_IN')       PUT(ENABLED) GET(ENABLED) BOTHRESH
(0) USAGE(NORMAL) REPLACE
AMQ8006: WebSphere MQ queue created.
    2 : define qlocal('LAB3_Debug_OUT1')     PUT(ENABLED) GET(ENABLED) BOTHRESH
(0) USAGE(NORMAL) REPLACE
AMQ8006: WebSphere MQ queue created.
    3 : define qlocal('LAB3_Debug_OUT2')     PUT(ENABLED) GET(ENABLED) BOTHRESH
(0) USAGE(NORMAL) REPLACE
AMQ8006: WebSphere MQ queue created.
    4 : define qlocal('LAB3_Debug_FAILURE')  PUT(ENABLED) GET(ENABLED) BOTHRESH
(0) USAGE(NORMAL) REPLACE
AMQ8006: WebSphere MQ queue created.
4 MQSC commands read.
No commands have a syntax error.
All valid MQSC commands were processed.
```

Initialize the Message Flow and Message Set projects

2. Import the following project into the workspace:
   C:\student\Lab3\install\LAB3_Debug_MFP (Message Flow Project)