Lab MQ: Build and test a simple message flow to use MQ nodes

Contents

Lab MQ: Build and test a simple message flow to use MQ nodes

In this lab, you build and test a simple message flow that uses MQ nodes. A message flow is like a program but is developed using a visual paradigm.

This getting started lab also outlines some concepts and features of IBM Integration Bus. The lab is based on use of IBM Integration Bus for Developers version 10 ("Developer Edition") on a Windows computer.

Introduction

The message flow that you develop makes basic use of IBM WebSphere MQ; it accepts an input message on a local WebSphere MQ queue, processes the message (in this case to log trace information to a file), and then outputs the message to another local WebSphere MQ queue.

The message flow that you develop is deployed to an integration server (in an integration node) where it runs. An *integration server* is an operating system process where user flows run. When running, the flow is available to process messages. There is no need to restart the integration node or the integration server to deploy a new or changed message flow.

The unit of deployment is a Broker Archive file (BAR). Broker archive files have a file extension of “bar”. A BAR file is essentially a ZIP file with a deployment descriptor. The deployment descriptor allows certain properties of the message flow to be overridden

The BAR file holds the artifacts to be deployed to a specific integration server in a specific integration node. It may contain applications and libraries in addition to message flows, message sets, XSL Transformations (XSLT) Style Sheets, Java Archive (JAR) files, XML schema files or WSDL files. Also, it may contain related source files. When you add a message flow to the BAR file, additional validation of the message flow is performed. The BAR file is then deployed to the integration node. The final validation of the artifacts is done by the integration node. If errors are found by the integration node, they are reported in the event log.

Getting to know the toolkit

1. To start the IBM Integration Toolkit ("the toolkit"), use the Windows Start menu; for example: click **Start** > **All Programs** > **IBM Integration Bus 10.0.0.0 Developer Edition** > **IBM Integration Toolkit 10.0.0.0**

2. The toolkit stores your projects in a folder called a workspace. When the toolkit has started, either use an existing workspace or create a workspace for the Lab. In this example, the toolkit option to always prompt for a workspace has been turned off, and the menu options are used to switch to a new workspace: `C:\IIBT10\workspace`
   a. Click **File** > **Switch Workspace** > **Other...**
   b. Change the Workspace field to `C:\IIBT10\workspace`
   c. Click **OK**

   The toolkit restarts to use the chosen workspace, and for a new workspace shows the welcome screen:

3. Click **Go to the Integration Toolkit**

Initially, the integration toolkit for a new workspace is displayed as shown in the following screen capture. Note the "TESTNODE_*userID*" integration node, and its “default” integration server.

4. Take a look around at the toolkit. For information about the toolkit, see the information in the toolkit help: **IBM Integration Bus 10.0.0** > **Product overview** > **IBM Integration Bus technical overview** > **IBM Integration Toolkit** (or the same IBM Integration Toolkit page in the online IBM Knowledge Center.

Toolkit key ideas

The Integration Toolkit is based on Eclipse and provides one Perspective specifically for IBM Integration Bus in addition to other Perspectives from Rational Application Developer and Eclipse.

The previous screenshot is of the Integration Development perspective. It is divided into multiple views (or panes). Each view is identified by a tab at the top of the view. On the lower left is the Outline view.

On the upper left is the Navigator view, which has tabs for projects (Application Development) and patterns (Patterns Explorer). The Navigator view contains the projects that are available within your workspace.

The area below the navigator view is the summary area. The Integration Nodes tab will show all defined local nodes as well as connections to remote nodes that have been created.

The large area on the right is used by the resource editors. When an editor has opened a resource, it will also be represented by a tab. Below the editor view is a pair of views for Properties and Problems.

On the top right are tabs for the perspectives that have been opened. To change an open perspective, you can simply click on its tab.

Eclipse is project oriented – artifacts are organized into projects. A project has a specific type. Project types that are specific to IBM Integration Bus are Applications and Libraries. Also, legacy projects of type Message Flow Project and Message Set Project can be created or imported into the toolkit. Since they predate the concept of Applications and Libraries, they will be visible in the hierarchy under a Folder called “Independent Resources”.

Building a simple message flow from scratch

1. In the Application Development tab, click **New...** > **Application**

   Note: This action is also available as an icon on the toolbar.
2. In the Application name field, type the name for the new application: `IntroLab`
3. Click **Finish**

   Now create a new message flow...
4. Select **IntroLab**
5. Click **New...** > **Message Flow**

   Tip: Select the application first before creating a new artifact for that application. Otherwise, if you start to create a new artifact you would need to then select the application as the “Container” for the message flow.
6. In the Message flow name field, type the name for the new message flow: `IntroMessageFlow`
7. Click **Finish**

   This displays the Message Flow Editor where you can compose the message flow.

8. Click on the Message Flow Editor. Information about the message flow appears in the Properties pane.
9. In the Properties page, select the Properties tab
10. Select the Description tab

11. Enter the following information:
    - **Version** `1.0`
    - **Short Description** `Introduction to IBM Integration Bus with MQ nodes`
    - **Long description** Your choice; for example: `Trying MQ features in IIB`
12. Save your changes.

    Tip: It is a good practice to save your changes as you go. If there is a small asterisk next to the message flow name, this indicates that changes to the message flow have not been saved.

Add an input node to the message flow

A message flow must begin with an Input node. An input node establishes the environment for the flow. There is an Input node for each of the various protocols that IBM Integration Bus supports. This flow is to process a WebSphere MQ message, so we need an MQInput node.

1. In the node palette, open the WebSphere MQ drawer. (Click on the palette to open or close it.)
2. Add an MQInput node to the canvas; for example drag the **MQInput** node from the palette and then drop it onto the canvas.

   You must specify a name for the node. When a node is initially added, its name can be changed immediately by over-typing the default name (or by entering a new value in the Node name field in the Description tab of the Properties pane).

   A good practice is to provide a new name for each node that is descriptive of the function that it provides. For most of the labs, you will be renaming the nodes. If you use the names as suggested it will make it easier to follow the lab guide. Another "naming convention" for MQInput and MQOutput nodes is to use the queue name that the node is accessing.
3. Change the name of the node to `XML_Input` (Press Enter to complete the rename.)
4. On the Basic tab, set the Queue name field to `LAB.IN`

   Note: Queue names are case-sensitive. All queue names in the lab are in uppercase.

5. Select the Description tab. This tab is used for general documentation. It can also be used to change the node name, as shown in the message flow editor. (Changing this name does not affect the operation of the message flow.)
6. In the Short description field, type `Q:LAB.IN`
7. In the Long description field, type your choice of text; for example: `Getting started with an MQInput node`

Move the mouse pointer onto the node name on the flow editor canvas. The information in the Short description field is displayed, so might be useful as hover help to clarify use of the node. When there are multiple nodes on the canvas, if you move from node to node with the mouse, the same tab in the Properties will be displayed.

Add a trace node to the message flow

To trace operations through the message flow, add a Trace node:

1. In the node palette, open the Construction drawer
2. Drag the Trace node and then drop it to the right of the XML_Input node on the canvas. You do not need to rename the Trace node.
3. Specify a file into which trace output is to be written.
   a. In the Basic tab, use the pull down list on the Destination field to select `File`.
   b. In the File Path field, enter `C:\XML_Input_Trace.txt`
4. Specify what information is to be produced in the trace output.

   The information in the Pattern box tells the node what information to produce in the trace output. If you type a line of raw text, it is echoed to the output.
   a. In the Pattern box, enter a line of text of your choice. (No quotes are needed.)
   b. In the Pattern box, enter the string `${Root}` exactly as indicated . This tells the Trace node to dump out the entire contents of the message that enters the node.

      Important: The pattern uses curly braces, not parenthesis. The expression between the curly braces will be evaluated at run time.

Add an MQOutput node to the message flow

The MQOutput node delivers an output message from a message flow to a WebSphere MQ queue.

1. In the node palette, open the WebSphere MQ drawer
2. Drag the MQOutput node and then drop it to the right of the Trace node on the canvas.
3. Change the node name to `Send_As_XML` (For example, overtype the name while it is highlighted, and then press Enter.)
4. On the Basic tab, set the Queue name field to `LAB.SEND.AS.XML`

Input and output terminals

As you work with the nodes, you work with their Input and Output terminals. Input terminals are typically named `In`. Most nodes have an Output terminal named `Out`. They may have several other terminals.

Some terminals have common names such as Failure or Catch and others are unique to that particular node. Some nodes allow you to define the terminals. The terminals are given a name when they are defined.

If you hover the mouse pointer over a terminal, a small popup dialog appears that identifies the name of the terminal.

The lab instructions identify the terminals to be used when connecting nodes together.

Creating a logical path through the message flow

To create a logical path for a message to follow through the message flow, you wire the nodes together.

1. You wire the Out terminal of the XML_Input node to the In terminal of the Trace node. Choose one of the two techniques:
   a. Position the mouse pointer over the Out terminal (in the middle), click and drag to the target and then click again.

b. Right click on a node and select **Create Connection**. This is an example of a Terminal Selection presented as a result of the Create Connection.

The following steps show use of the first (drag and drop) technique:

2. Position the mouse pointer over the middle terminal of the XML_Input node (Hover text shows it is the Out terminal)
3. Drag the terminal to the left (input) terminal of the Trace node, and then click that terminal.
4. Place your mouse pointer on the connection. A pop up a summary of "from-to" information will appear.
5. Connect the Trace node output terminal to the Send_As_XML node input terminal, by using the same technique (or the alternative technique)

Your message flow should now show the three nodes connected.

Creating the WebSphere MQ queues

The message flow uses WebSphere MQ as its Input and Output transport, so we must create the MQ Input and Output queues in WebSphere MQ, on the queue manager used by IBM Integration Bus. The following steps use the WebSphere MQ Explorer, and assume that the queue manager name is QM1.

1. Start the WebSphere MQ Explorer. For example, right-click the WebSphere MQ icon in the system tray and then in the context menu click **WebSphere MQ Explorer**.
2. Take a moment to familiarize yourself with the Explorer. This tool is used quite often when working with WebSphere MQ.

   If you want to learn more about the WebSphere MQ Explorer, see the help provided in the Explorer; for example, in the Help system: **WebSphere MQ Explorer** > **Reference** > **Views in MQ Explorer**
3. Create the Input queue, LAB.IN (as specified earlier on the MQInput node):
   a. In the Navigator pane, display the Queues folder for the queue manager: Expand **IBM WebSphere MQ** > **Queue Managers** > **QM1**
   b. Right-click the Queues folder
   c. Select **New** > **Local Queue...**
   d. In the Name field of the dialog, type `LAB.IN`
   e. Click **Finish**
   f. A Confirmation dialog id displayed to confirm that the queue was created successfully. Close the dialog.
4. Create the Output queue, LAB.SEND.AS.XML (as specified earlier on the MQOutput node):
   a. In the Navigator pane, display the Queues folder for the queue manager: Expand **IBM WebSphere MQ** > **Queue Managers** > **QM1**
   b. Right-click the Queues folder
   c. Select **New** > **Local Queue...**
   d. In the Name field of the dialog, type `LAB.SEND.AS.XML`
   e. Click **Finish**
   f. A Confirmation dialog id displayed to confirm that the queue was created successfully. Close the dialog.

The message flow is now complete, and ready to be tested in the runtime environment.

Testing the message flow

To test the message flow from the Integration Toolkit, the following steps use the Flow Exerciser. You can use the Flow Exerciser to deploy a message flow and then create and send messages to the flow. After the messages are processed, the paths that the messages took are highlighted. You can then view the structure and content of the logical message tree on any highlighted connection in the message flow.

1. In the flow editor, click the Start Flow exerciser icon ( ) in the Flow exerciser toolbar.

   The message flow is deployed to the integration server, and the integration server is set to recording mode. The message flow in the message flow editor is now in read-only mode

   Note: Note that a BAR file is automatically created and used to deploy the message flow. The BAR file is shown in the Navigation pane.

2. Send a message to the message flow
   a. Open the Send Message dialog, to choose or create a message to send. Click the **Send message to the flow** icon ( ) in the Flow exerciser toolbar.
      Continue to select the existing input message for this Lab.
   b. In the Send Message dialog, click the New icon ( )
   c. Select the checkbox **Import from file**, then click the **File system...** button
   d. Navigate to the file `IN_Request.xml`, select that file, and then click **Open**
      The Send message dialog displays the contents of the message.

      You can browse the message and optionally change some of the content.
3. Click **Send** to send the message to the message flow.
   A dialog window is displayed to show events in progress information.

   In the Progress Information, you can see that the message is sent to the WebSphere MQ queue "LAB.IN". Also, recall that the output target from the message flow is a WebSphere MQ queue, LAB.SEND.AS.XML. The event `MQ Queue Monitor "LAB.SEND.AS.XML"` indicates that the output message was delivered to the output queue. If you look at the queues in the WebSphere MQ Explorer when the test message is sent, you see that the message is added to the MQ queues:

4. Examine the connections through which the message passed.
   The connections through which the message passed are highlighted in the flow editor.

   Click a highlighted connection to view the data that passed through the connection. A Recorded Message dialog is displayed that shows the logical message tree for each occasion that a message passed through the connection.
5. To return the message flow that is in the message flow editor to edit mode, click the **Return flow to edit mode** icon ( ). The recorded messages are cleared from the message flow and you are able to edit the flow again. The message flow is still deployed on the integration server, and the integration server is still in recording mode.
6. If you are not recording messages on any other message flow that is deployed on the integration server, turn off recording mode for the integration server:
   a. In the Integration Nodes view, right-click the integration server.

b. Select **Stop Recording**

All recorded messages are cleared from the integration server, and the integration server is removed from recording mode.

Examine the output of the Trace Node

When a message is sent through the message flow, the Trace node records output to the file `C:\XML_Input_Trace.txt`. When we configured the node, we indicated that the output should be:

```
Trace for my MQ flow
${Root}
```

where, `${Root}` tells the Trace node to dump out the entire contents of the message that enters the node.

1. Open the file `C:\XML_Input_Trace.txt` (for example, in Windows Notepad)
2. Note that the output starts with:

```
Trace for my MQ flow  ( ['MQROOT' : 0x18d609f0]
  (0x01000000:Name):Properties = ( ['MQPROPERTYPARSER' : 0x211b1ee0]
...
```

3. To see the actual application data scroll down to the bottom (Ctrl+End).

```
...
    (0x03000000:NameValue):OriginalLength   = -1 (INTEGER)
  )
  (0x01000000:Name):BLOB       = ( ['none' : 0x211b28d0]
    (0x03000000:NameValue):UnknownParserName = '' (CHARACTER)
    (0x03000000:NameValue):BLOB              = X'3c3f786d6c2076657273696f6e3d22312e302220656e636f64696e673d227574662d
  )
)
```

The message is a BLOB (a Binary Large Object); just a string of bytes shown in hexadecimal that happens to be XML. (You can associate an XML parser with the input message.)

Each time that you test the message flow, new data is appended to the end of the trace file. Each time, to see the latest information you need to scroll down to the end of the file.

Summary

In this lab, you have built and tested a simple message flow that uses MQ nodes. You have sent a message through the flow, and seen the message put onto input and output queues in WebSphere MQ. You have also seen trace output written to a file.