

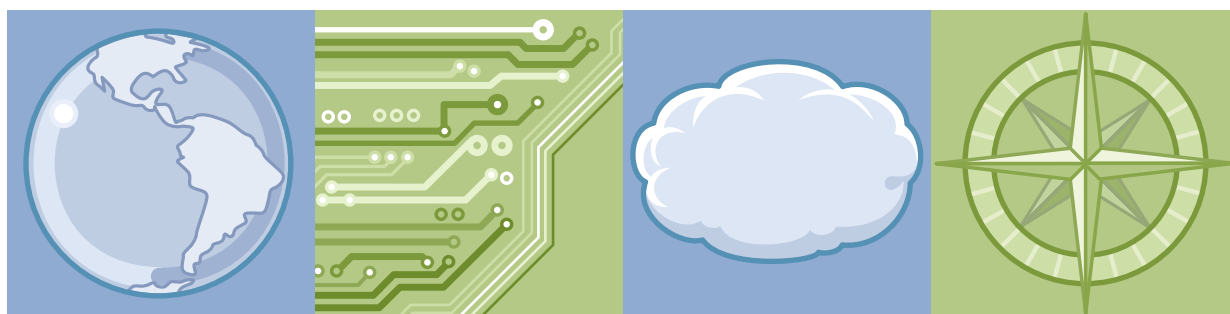


# IBM Training

## Student Exercises

### **IBM Integration Bus V10 System Administration**

Course code WM646 ERC 1.0



IBM Systems  
Middleware

## Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide.

The following are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide:

AIX®	CICS®	DataPower®
DB2®	developerWorks®	Express®
Informix®	MQSeries®	Notes®
ObjectGrid®	PartnerWorld®	PowerHA®
RACF®	Redbooks®	Tivoli®
WebSphere®	z/OS®	

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

VMware and the VMware "boxes" logo and design, Virtual SMP and VMotion are registered trademarks or trademarks (the "Marks") of VMware, Inc. in the United States and/or other jurisdictions.

Other product and service names might be trademarks of IBM or other companies.

### December 2015 edition

The information contained in this document has not been submitted to any formal IBM test and is distributed on an "as is" basis without any warranty either express or implied. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will result elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

© Copyright International Business Machines Corporation 2015.

**This document may not be reproduced in whole or in part without the prior written permission of IBM.**

US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

# Contents

<b>Trademarks</b> .....	<b>v</b>
<b>Exercises description</b> .....	<b>vii</b>
<b>Exercise 1. Integration node setup and customization</b> .....	<b>1-1</b>
Part 1: Get build and version information .....	1-2
Part 2: Verify user authorizations .....	1-7
Part 3: Create an integration node and integration servers .....	1-8
<b>Exercise 2. Connecting to IBM MQ</b> .....	<b>2-1</b>
Part 1: Use IBM MQ Explorer to create the queue manager .....	2-4
Part 2: Create the integration nodes and integration servers .....	2-5
Part 3: Create the SYSTEM.BROKER queues and topics on the integration node queue manager .....	2-8
Part 4: Create an MQEndpoint policy .....	2-10
<b>Exercise 3. Using the IBM Integration Toolkit</b> .....	<b>3-1</b>
Part 1: Work with a simple message flow .....	3-5
Part 2: Create and deploy a BAR file .....	3-8
Part 3: Test the message flow by using the Flow Exerciser .....	3-13
<b>Exercise 4. Administering the IBM Integration Bus runtime components</b> .....	<b>4-1</b>
Part 1: Use the IBM Integration command interface .....	4-3
Part 2: Use the IBM Integration web interface .....	4-6
Part 3: Use the IBM Integration API Exerciser .....	4-12
Part 4: Back up and restore the integration node .....	4-16
<b>Exercise 5. Using file-based security to control administration access</b> .....	<b>5-1</b>
Part 1: Activate file-based security on the integration node .....	5-4
Part 2: Define administration roles and set file-based permissions. ....	5-6
Part 3: Test the IBM Integration web interface with security .....	5-8
<b>Exercise 6. Using queue-based security to control administration access</b> .....	<b>6-1</b>
Part 1: Set IBM MQ permissions for users .....	6-4
Part 2: Test the default unsecured environment .....	6-6
Part 3: Activate administration security and test the secured environment .....	6-10
<b>Exercise 7. Implementing web services and web services security</b> .....	<b>7-1</b>
Part 1: Configure the integration node HTTP listener .....	7-4
Part 2: Review the message flows .....	7-5
Part 3: Create and deploy a BAR file .....	7-7
Part 4: Test the web service application .....	7-12
Part 5: Implement SSL for the web service application .....	7-14
<b>Exercise 8. Using problem diagnosis tools</b> .....	<b>8-1</b>
Part 1: Examine and deploy the BAR file .....	8-3
Part 2: Test the message flow .....	8-5

Part 3: Manage traces and Trace nodes .....	8-6
Part 4: Administer the message flow debugger .....	8-11
<b>Exercise 9. Identifying runtime problems .....</b>	<b>9-1</b>
Part 1: Analyze message flow runtime failures .....	9-2
Part 2: Isolate and identify problems .....	9-6
<b>Exercise 10. Viewing runtime statistics .....</b>	<b>10-1</b>
Part 1: Enable message flow and resource statistics .....	10-3
Part 2: View message flow and resource statistics .....	10-5
Part 3: Set up subscriptions for resource statistics .....	10-10
<b>Exercise 11. Administering workload management policies .....</b>	<b>11-1</b>
Part 1: Create a workload management policy .....	11-3
Part 2: Attach a workload management policy to a message flow .....	11-4
<b>Exercise 12. Recording and replaying message flow data .....</b>	<b>12-1</b>
Part 1: Set up the environment for record and replay .....	12-3
Part 2: View messages in the IBM Integration web interface .....	12-6
Part 3: Replay messages .....	12-11
Part 4: Handle failed messages .....	12-13
<b>Appendix A. Exercise solutions .....</b>	<b>A-1</b>

# Trademarks

The reader should recognize that the following terms, which appear in the content of this training document, are official trademarks of IBM or other companies:

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide.

The following are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide:

AIX®	CICS®	DataPower®
DB2®	developerWorks®	Express®
Informix®	MQSeries®	Notes®
ObjectGrid®	PartnerWorld®	PowerHA®
RACF®	Redbooks®	Tivoli®
WebSphere®	z/OS®	

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

VMware and the VMware "boxes" logo and design, Virtual SMP and VMotion are registered trademarks or trademarks (the "Marks") of VMware, Inc. in the United States and/or other jurisdictions.

Other product and service names might be trademarks of IBM or other companies.



# Exercises description

This course includes the following exercises:

- Integration node setup and customization
- Connecting to IBM MQ
- Using the IBM Integration Toolkit
- Administering the IBM Integration Bus runtime components
- Using file-based security to control administration access
- Using queue-based security to control administration access
- Implementing web services and web services security
- Using problem diagnosis tools
- Identifying runtime problems
- Viewing runtime statistics
- Administering workload management policies
- Recording and replaying message flow data

In the exercise instructions, you can check off the line before each step as you complete it to track your progress.

Most exercises include required sections, which should always be completed. It might be necessary to complete these sections before you can start later exercises. Some exercises might also include optional sections that you might want to complete if you have sufficient time and want an extra challenge.

Use the following user name and password to log in to course image:

User name: `iibadmin`

Password: `websphere`





# Exercise 1. Integration node setup and customization

## What this exercise is about

In this exercise, you complete the configuration tasks that prepare the IBM Integration Bus environment for use. You also start the IBM Integration web user interface to view the integration node and integration server properties.

## What you should be able to do

After completing this exercise, you should be able to:

- Get build and version information for IBM Integration Bus components
- Verify user authorizations
- Create and verify an integration node and integration server
- Start the IBM Integration web user interface

## Introduction

In the first part of this exercise, you verify the maintenance (fix pack) levels of the IBM Integration Bus software.

In this second part of the exercise, you verify that the user ID that is used in the lab exercises has administrative privilege to create and manage IBM Integration Bus and IBM MQ resources.

In the third part of this exercise, you use the IBM Integration command interface to create an integration node and report integration node properties. You then use the IBM Integration command interface and the IBM Integration web interface to create integration servers on the new integration node.

## Requirements

- IBM Integration Bus V10
- IBM MQ V8
- User ID of “iibadmin” that is a member of the “mqbrkrs” group

## Exercise instructions

The required licensed programs are already installed on your exercise image. These licensed programs include:

- IBM Integration Bus V10
- IBM MQ V8
- IBM DB2 Express 10.1

Your user ID (`iibadmin`) is a member of the Windows “Administrators” group.

**For all exercises in this course, use the information that is listed in this table.**

<b>Windows user ID for lab exercises</b>	<code>iibadmin</code>
<b>Windows password for lab exercises</b>	<code>web1sphere</code>
<b>IBM Integration Bus installation directory</b>	<code>c:\Program Files\IBM\IIB\10.0.0.0\</code>
<b>IBM MQ installation directory</b>	<code>c:\Program Files\IBM\WebSphere MQ</code>
<b>IBM Integration Bus integration node for development</b>	<code>TESTNODE_iibadmin</code>

The lab image also includes the `C:\labfiles` directory that contains, test files, scripts, and other components that you use during the course exercises.

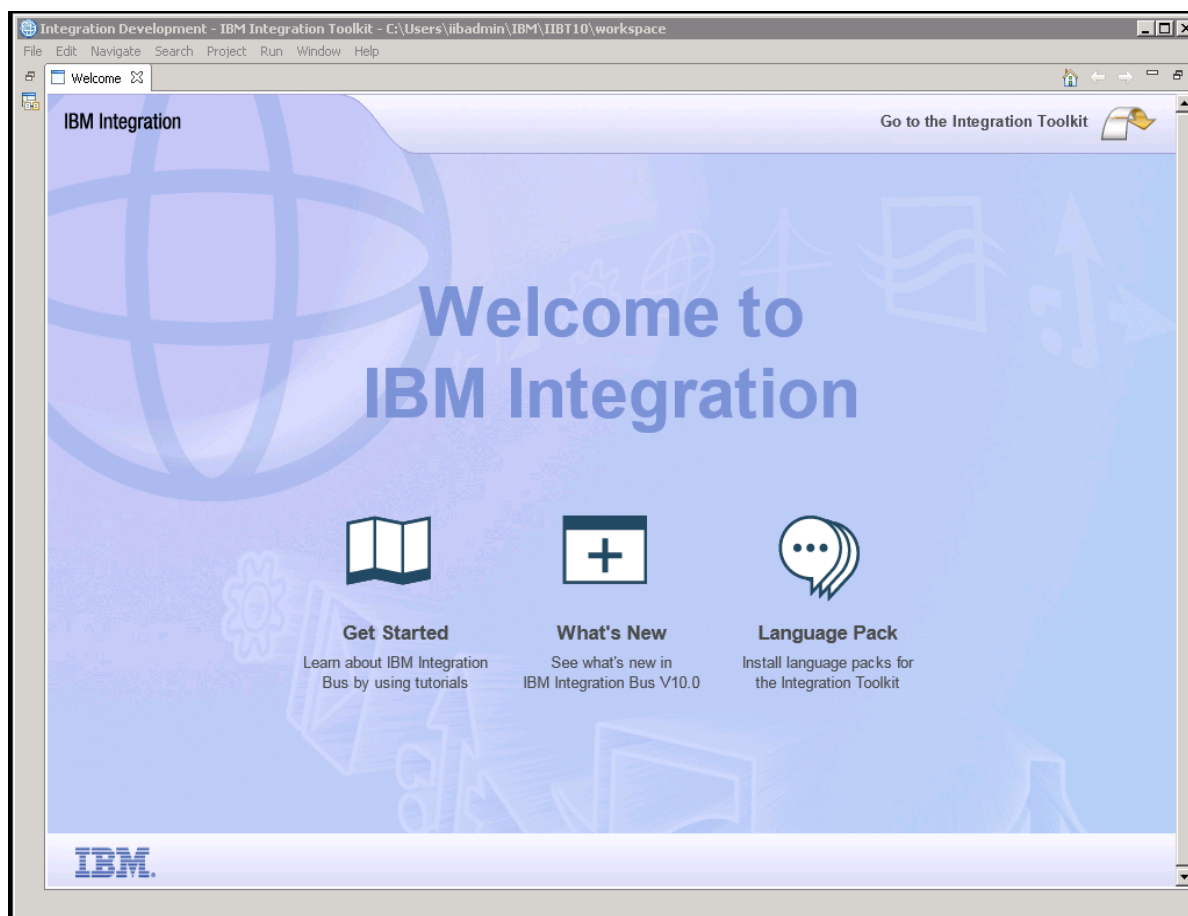
### ***Part 1: Get build and version information***

It is important to know the current software version and build level for all your software. In this part of the exercise, you verify the maintenance (fix pack) levels of the IBM Integration Bus software.

- \_\_\_ 1. Get the software version information for the IBM Integration Toolkit.
  - \_\_\_ a. From the Windows **Start** menu, click **All Programs > IBM Integration Bus 10.0.0.0 > IBM Integration Toolkit 10.0.0.0**. Optionally, you can double-click the **IBM Integration Toolkit** shortcut icon on the Windows desktop.

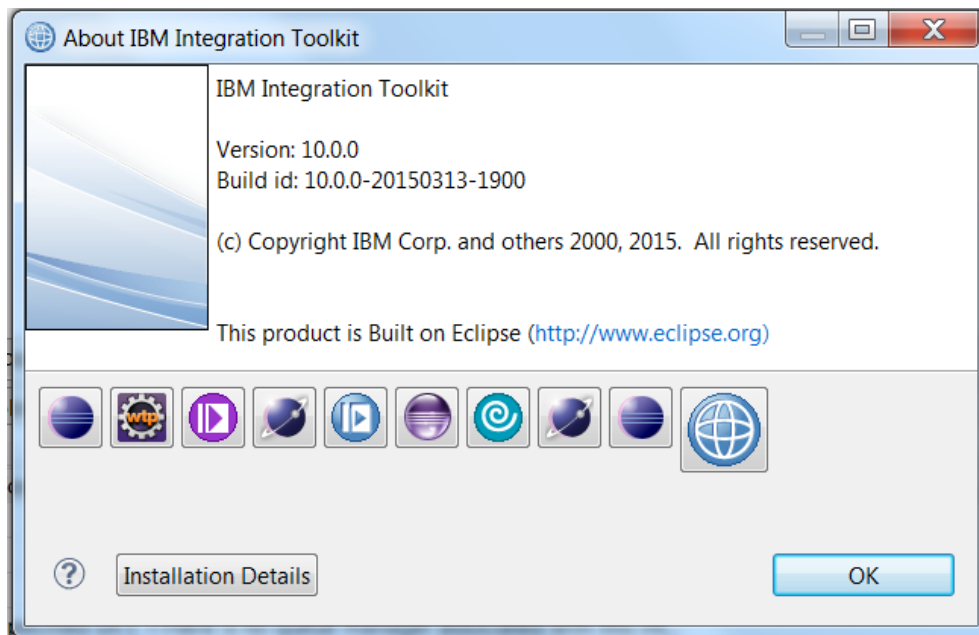
After a few moments, the Integration Toolkit opens and the Welcome page is presented.

On the Welcome page, you can learn more about the features of Integration Bus, see what is new in IBM Integration Bus V10, and install language packs for the Integration Toolkit.



- \_\_\_ b. Close the Welcome page by clicking **Go to the Integration Toolkit**.
- \_\_\_ c. To display configuration information about the Integration Toolkit and the Integration Bus environment, click **Help > About IBM Integration Toolkit**.

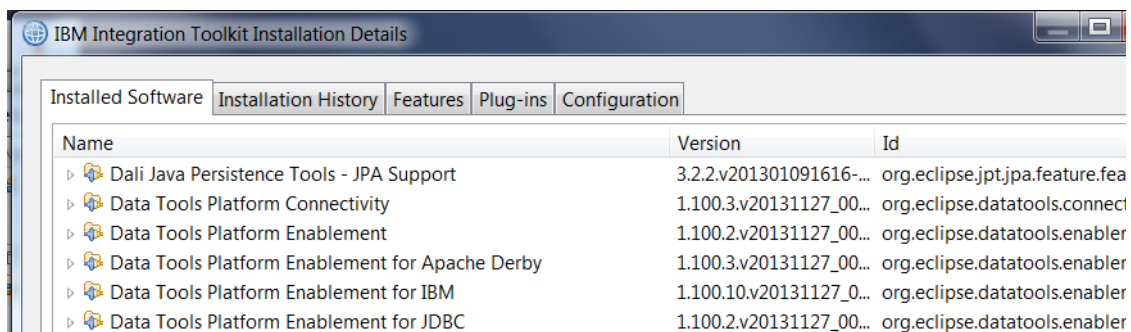
The About IBM Integration Toolkit window shows the Integration Toolkit version information.



The **Build ID** identifies the most recent fix pack that is applied. In this example, the build ID is 10.0.0, dated March 13 2015 (20150313).

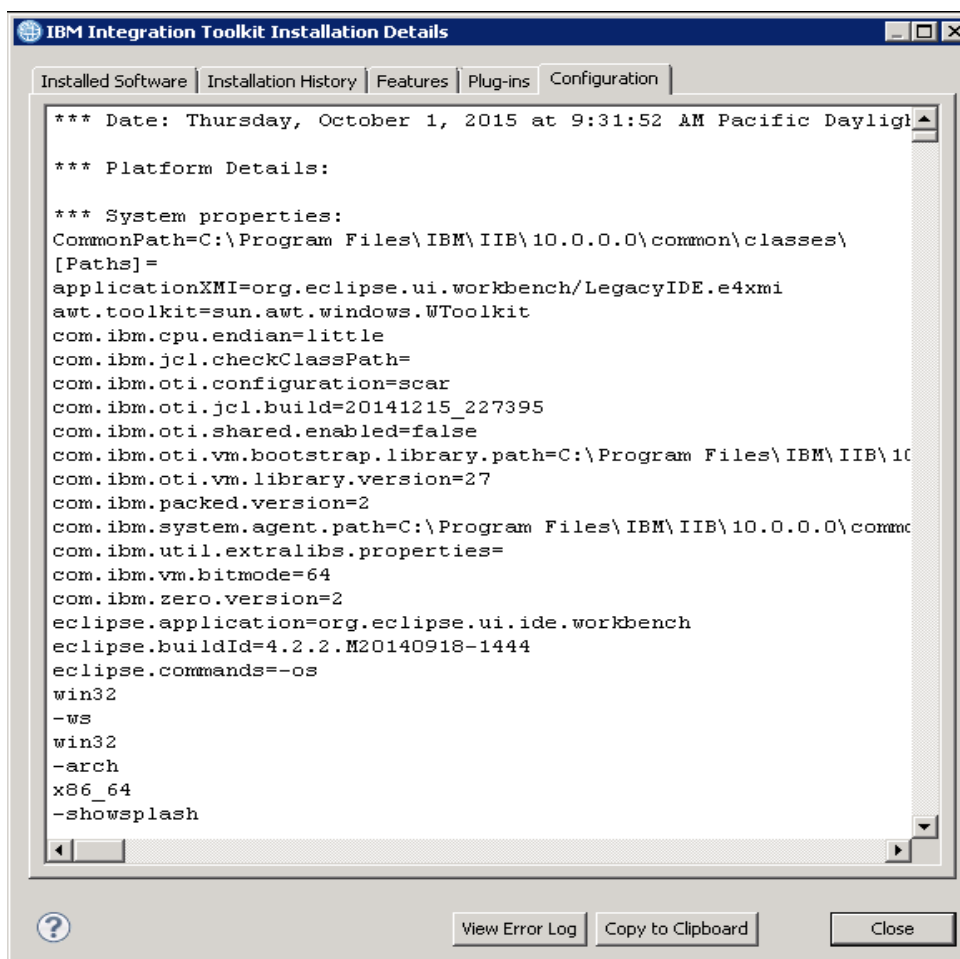
- \_\_\_ d. The icons across the About IBM Integration Toolkit window contain information about the features that are installed in the Integration Toolkit. Click one or more of these icons and explore the installed elements.
- \_\_\_ e. Click **Installation Details** in the About IBM Integration Toolkit window to show the installed Integration Toolkit components and their versions.

Within the Installation Details page, you can select tabs to view the installed components, the IBM Integration Toolkit installation history, the installed features, and plug-ins.



- \_\_\_ f. in the IBM Integration Toolkit Installation Details window, click the **Configuration** tab.

The **Configuration** tab contains the details about the installed configuration of the Integration Toolkit environment. This page is a summary of the other tabs on the Installation Details page.



Before you update any components in the Integration Toolkit, you should use the **Copy to Clipboard** action, and then paste the clipboard contents into a file. Save the file for reference in case you need to determine which components are installed at a specific point in time.

- \_\_\_ g. Click **Close** to close the configuration details.
- \_\_\_ h. Click **OK** to close the About IBM Integration Toolkit window.
- \_\_\_ i. Leave the IBM Integration Toolkit open (minimize it if you want); you use it later in this exercise.
- \_\_\_ 2. Get the software version information for the IBM Integration Bus.
  - \_\_\_ a. Start an IBM Integration Bus Console session by clicking **All Programs > IBM Integration Bus 10.0.0.0 > IBM Integration Console 10.0.0.0** from the Windows **Start** menu. As an option, you can double-click the **IBM Integration Console** shortcut icon on the desktop.

**Important**

When IBM Integration Bus is installed on Windows, a special command console is also installed from which you run Integration Bus commands. It is a Windows command shell, but it sets some environment variables when it is started.

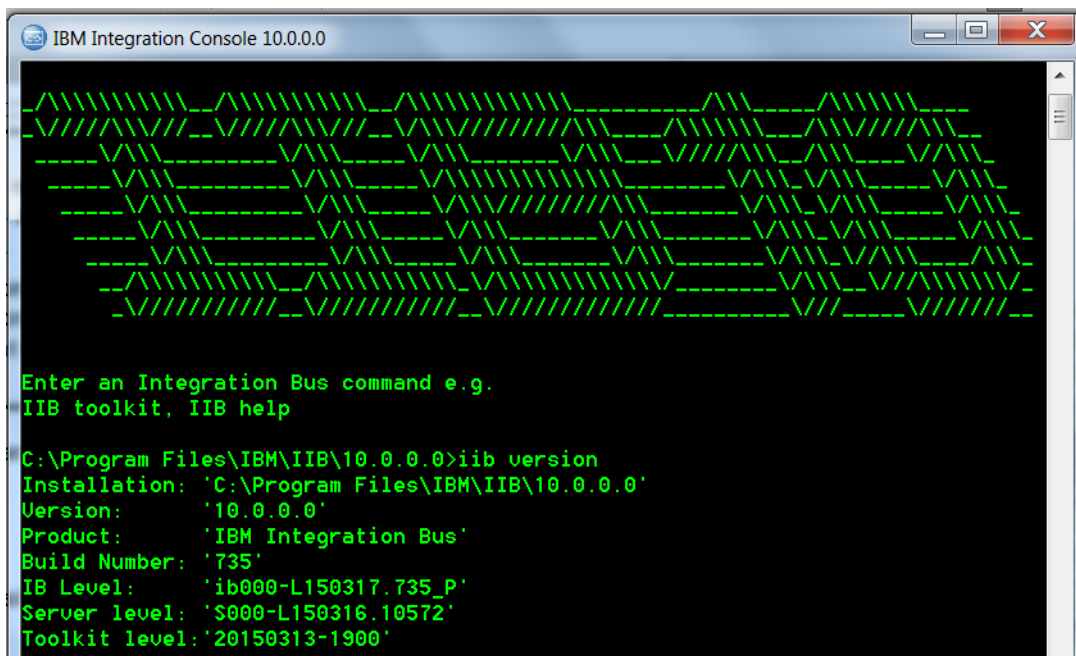
Be sure to use an Integration Bus command console session when you are directed to do so in the exercise instructions. If you attempt to use a Windows command prompt session, the commands you enter might fail because the appropriate environment variables are not set.

- \_\_\_ b. Use the `iib` command to report the details of an IBM Integration Bus component.

In the IBM Integration Console, enter the command:

```
iib version
```

The response message contains the version, product, build levels and build number.



```
IBM Integration Console 10.0.0.0

Enter an Integration Bus command e.g.
IIB toolkit, IIB help

C:\Program Files\IBM\IIB\10.0.0.0>iib version
Installation: 'C:\Program Files\IBM\IIB\10.0.0.0'
Version:      '10.0.0.0'
Product:      'IBM Integration Bus'
Build Number: '735'
IB Level:     'ib000-L150317.735_P'
Server level: 'S000-L150316.10572'
Toolkit level: '20150313-1900'
```

The **Version** and **Level** fields identify the fix pack that is applied to IBM Integration Bus. This information is useful when working with IBM Support.

In the example, the version is 10.0.0.0, which indicates that no fix pack is applied. For example, FixPack 1 would show as version 10.0.0.1.

- \_\_\_ c. Minimize the IBM Integration Console window; you use it later in this exercise.

## Part 2: Verify user authorizations

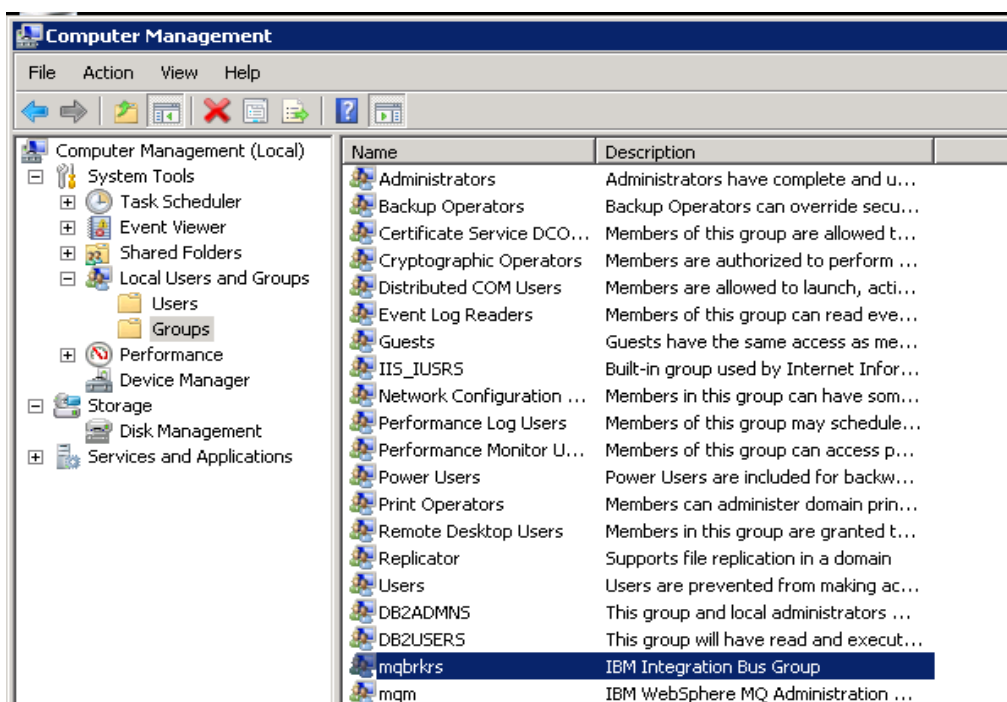
In this part of the exercise, you verify that the user ID that is used in the lab exercises has administrative privileges.

- \_\_\_ 1. To administer the integration node components and resources, the integration node service ID must be a member of the `mqbrkrs` security group.

In this step, you verify that the `mqbrkrs` security group exists on your system and that your account (`iibadmin`) is a member. If the `mqbrkrs` group cannot be found, you create it and add your account as a member of that group.

- \_\_\_ a. From the Windows **Start** menu, click **All Programs > Administrative Tools > Computer Management**.
- \_\_\_ b. Expand **System Tools > Local Users and Groups**.
- \_\_\_ c. Click **Groups** to display the users and groups.

If you are using the VMware image for this course, the `mqbrkrs` group is already created. You do not need to create it.



- \_\_\_ d. Double-click the `mqbrkrs` group to display the members of the group.
- \_\_\_ e. Verify that `iibadmin` (your user ID for this course) is listed as a member of the group.



### Note

If you are not using the pre-built image for this course, and the `mqbrkrs` group does not exist, you can create it and then add your user ID to the **Members** list.

1. Right-click **Users** and click **New > Group** from the menu.
2. Type `mqbrkrs` in the **Group name** field.

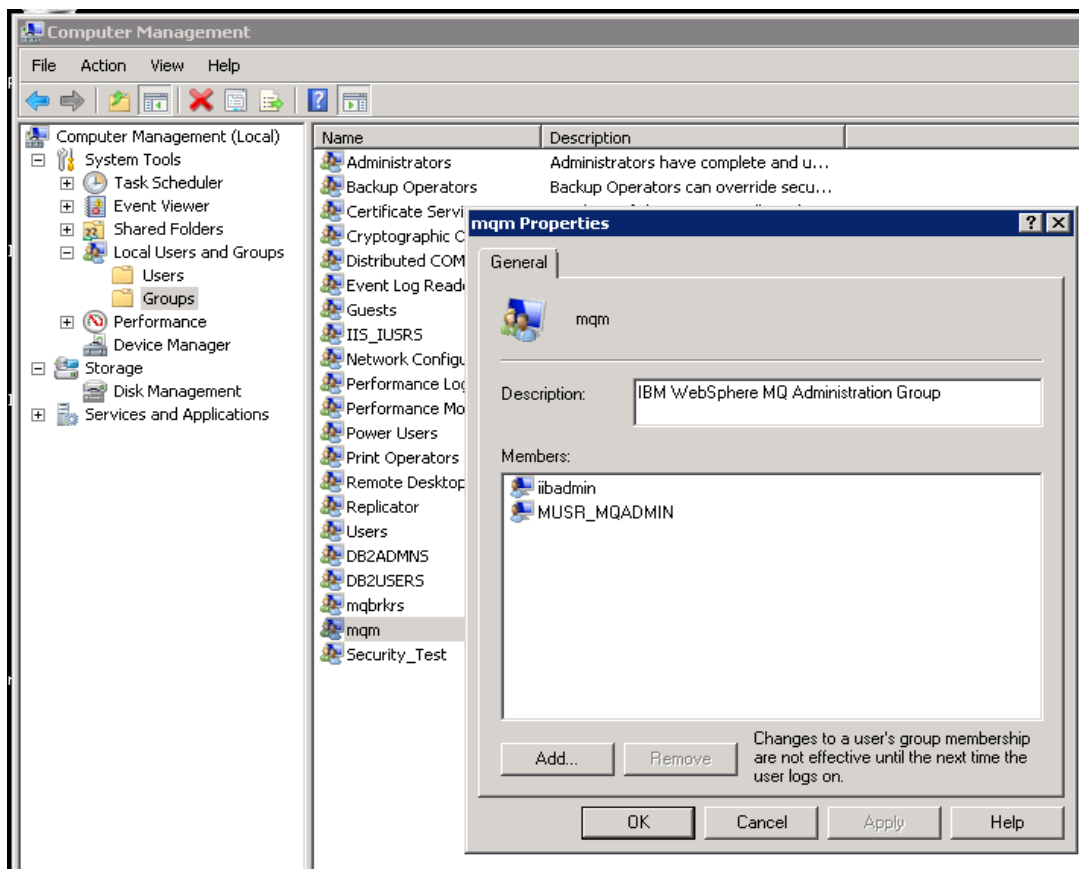
3. Click **OK**.
4. Open the new group and then add your account ID as a member of the **mqbrkrs** group.

\_\_\_ f. Click **Cancel** to close the Properties window for the **mqbrkrs** group.

- \_\_\_ 2. Some Integration Bus applications require IBM MQ. In these applications, the administrator must also be a member of the IBM MQ administration group **mqm** (this group is created automatically when IBM MQ is installed).

Verify that your user ID **iibadmin** is a member of **mqm** group.

If necessary, add **iibadmin** to the **mqm** group.



- \_\_\_ 3. Close the Computer Management window.

### **Part 3: Create an integration node and integration servers**

In this part of the exercise, you use the IBM Integration Bus command interface to create an integration node. This integration node does not have an IBM MQ queue manager that is specified on the integration node.



**Note**

For the following steps, you use an IBM Integration Console. Remember that command arguments are case-sensitive.

- \_\_\_ 1. Start an IBM Integration Console session with Windows “Administrator” privileges.

Click **Start > All Programs > IBM Integration Bus 10.0.0.0** and then right-click **IBM Integration Console 10.0.0.0**, and then click **Run as administrator**.

Optionally, you can right-click the **IBM Integration Console** icon on the desktop and then click **Run as administrator**.

Click **Yes** to run the IBM Integration Console with “Administrator” privileges.

- \_\_\_ 2. Type the following command to create an integration node that is named IIBNODE\_NOQM:

```
mqsicreatebroker IIBNODE_NOQM
```

- \_\_\_ 3. Start the new integration node.

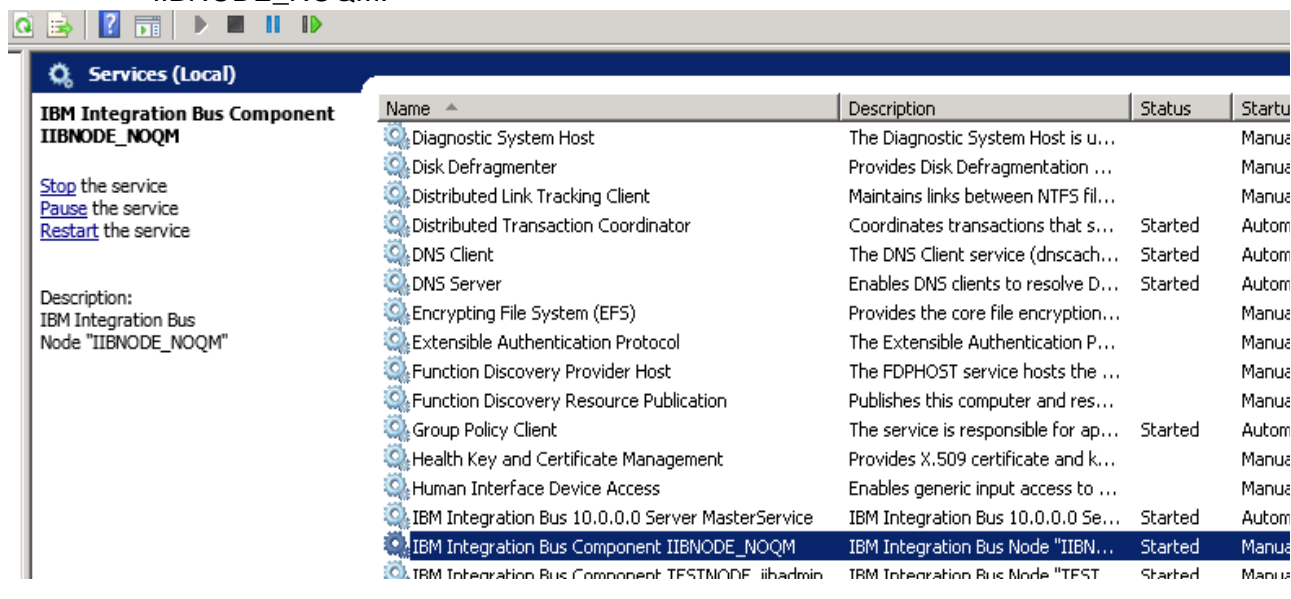
In the IBM Integration Console, type:

```
mqsistart IIBNODE_NOQM
```

You should receive a response similar to the following example.

```
BIP8071III: Successful command initiation, check the system log to ensure that
the component started without problems and that it continues to run without
problems.
```

- \_\_\_ 4. On Windows, a Windows service is created automatically when you create an integration node. Verify that a Windows service is created for the new integration node.
  - \_\_\_ a. In Windows, click **Start > All Programs > Administrative Tools > Services**.
  - \_\_\_ b. Scroll down and verify that you see a service for IBM Integration Bus Component IIBNODE\_NOQM.



**Note**

You might also see a service for the IBM Integration Bus Component TESTNODE\_iibadmin. This is the service for the development integration node that is created automatically the first time that the IBM Integration Toolkit starts.

- \_\_ c. Close the Services window.
- \_\_ 5. Check that the integration node environment is set up correctly.

In the IBM Integration Console, type:

```
mqsicvp IIBNODE_NOQM
```

You should receive a response similar to the following example.

```
BIP8873I: Starting the component verification for component 'IIBNODE_NOQM'.
BIP8876I: Starting the environment verification for component
'IIBNODE_NOQM'.
BIP8894I: Verification passed for 'Registry'.
BIP8894I: Verification passed for 'MQSI_REGISTRY'.
BIP8894I: Verification passed for 'Java Version - 1.7.0 IBM Windows AMD 64
build pwa6470_27sr2fp10-20141218_02(SR2 FP10)
BIP8894I: Verification passed for 'MQSI_FILEPATH'.
BIP8878I: The environment verification for component 'IIBNODE_NOQM' has
finished successfully.
BIP8882I: Starting the WebSphere MQ verification for component
'IIBNODE_NOQM'.
BIP8874I: The component verification for 'IIBNODE_NOQM' has finished
successfully.
```

- \_\_ 6. Verify that new integration is created in “advanced” mode.

In the IBM Integration Console, type:

```
mqsimode IIBNODE_NOQM
```

You should receive a response similar to the following example.

```
BIP1802I: Integration node 'IIBNODE_NOQM' is in 'advanced' mode.
BIP1831I: Integration node 'IIBNODE_NOQM' has mode extensions ''.
BIP8071I: Successful command completion.
```

---

\_\_ 7. Display the node properties for the new integration node.

In the IBM Integration Console, type:

**`mqsireportbroker IIBNODE_NOQM`**

You should receive a response similar to the following example.

```
BIP8927I: Integration node name 'IIBNODE_NOQM'
Last mqsisstart path = 'C:\Program Files\IBM\IIB\10.0.0.0\server'
mqsiprofile install path = 'C:\Program Files\IBM\IIB\10.0.0.0\server'
Work path = 'C:\ProgramData\IBM\MQSI'
Integration node UUID = ''
Process id = '3004'
Queue Manager = ''
User lil path = ''
User exit path = ''
Active user exits = ''
LDAP principal = ''
LDAP credentials = ''
ICU converter path = ''
Trusted (fastpath) Queue Manager application = 'false'
Configuration change timeout = '300' seconds
Internal configuration timeout = '60' seconds
Statistics major interval = '60' minutes
Operation mode = 'advanced'
Fixpack capability level = '' (effective level 'unrestricted')
Integration node registry format = 'v10.0'
Administration security = 'inactive'
Multi-instance integration node = 'false'
Shared Work Path = 'none'
Start as WebSphere MQ Service = 'undefined'
HTTP listener port = '7080'
Cache manager policy = 'disabled'
Cache manager port range = '2820-2039'
Integration registry hostname = ''
Default integration node CCSID = '5348'
```

\_\_ 8. Report integration node service details.

In the IBM Integration Console, type:

**mqsiservice IIBNODE\_NOQM**

You should receive a response similar to the following example.

```
BIPmsgs en_US
  Console OEM CP=437, ICU CCSID=5348
  Default codepage=ibm-5348_P100-1997, in ascii=ibm-5348_P100-1997
  JAVA console codepage name=cp437
```

```
Install Path = C:\Program Files\IBM\IIB\10.0.0.0\server
Shared Work Path = NOT_HA_ENABLED_BROKER
Local Work Path = C:\ProgramData\IBM\MQSI
process id = 888
service userId = LocalSystem
general default userId = LocalSystem
pubsub migration = no
fastpath Queue Manager = no
configuration timeout = 300
configuration delay timeout = 60
statistics major interval = 60
ComponentType = Broker
Fixpack capability level = (effective level unrestricted)
```

\_\_\_ 9. Get web interface port number for the integration node.

\_\_\_ a. In the IBM Integration Console, type:

```
mqsireportproperties IIBNODE_NOQM -b webadmin -o HTTPConnector -a
```

You should receive a response that is similar to the following example.

```
HTTPConnector
  uuid='HTTPConnector'
  address=''
  port='4415'
  maxPostSize=''
  acceptCount=''
  compressableMimeTypes='text/html,text/css,application/javascript,image/gif,
  image/png,application/json'
  compression='on'
  connectionLinger=''
  connectionTimeout=''
  maxHttpHeaderSize=''
  maxKeepAliveRequests=''
  maxThreads=''
  minSpareThreads=''
  noCompressionUserAgents=''
  restrictedUserAgents=''
  socketBuffer=''
  tcpNoDelay=''
  enableLookups='false'
  serverName=''
```

\_\_\_ b. Record the value for HTTPConnector **port** property. You use this value in the next step to start the IBM Integration web interface for this integration node.



### Information

IBM Integration Bus assigns a web user administration port number dynamically. When the first integration node is started, it is assigned the default port of 4414. For subsequent integration nodes, the port number increments and the node is assigned the next port number in sequence.

\_\_\_ 10. Create an integration server by using a command.

In the IBM Integration Console, type:

```
mqsicreateexecutiongroup IIBNODE_NOQM -e server1
```

You should receive a response that is similar to the following example.

```
BIP1124I: Creating integration server 'server1' on integration node
'IIBNODE_NOQM'
BIP1117I: Integration server was created successfully.
The integration node has initialized the integration server.
```

- \_\_\_ 11. Confirm the components on the integration node by using the **mqsilist** command. Type:

```
mqsilist IIBNODE_NOQM
```

You should receive a response that is similar to the following example.

```
BIP8071I: Integration server 'server1' on integration node 'IIBNODE_NOQM' is running.
```

- \_\_\_ 12. Start the IBM Integration web interface for the new integration node.

- \_\_\_ a. Open a web browser. Your course image includes both Internet Explorer and Mozilla Firefox.

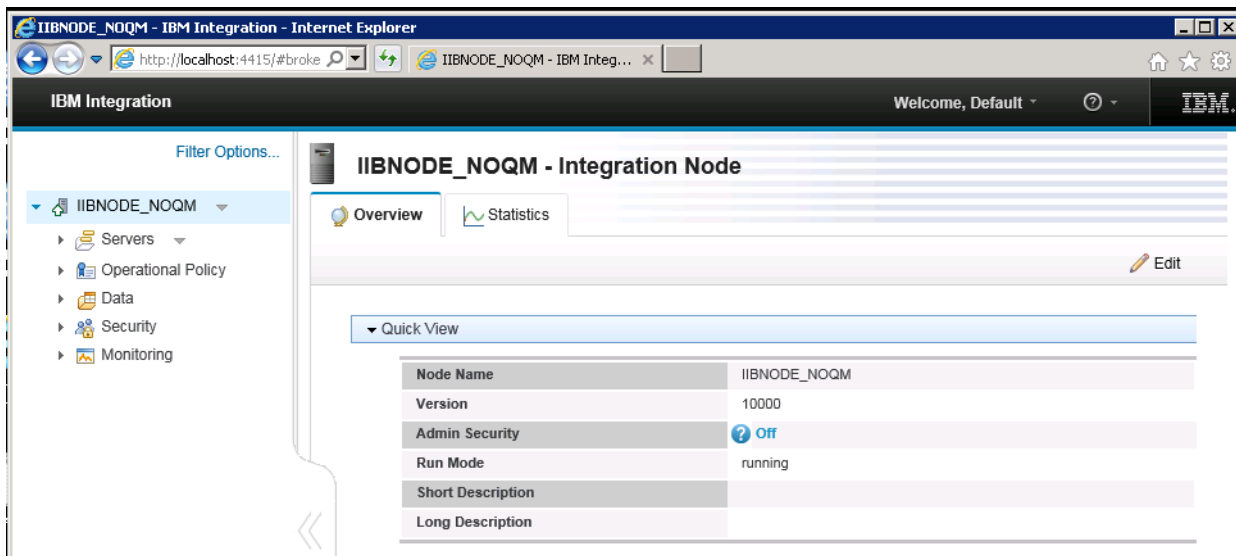
- \_\_\_ b. For the URL, type:

```
http://localhost:port
```

Replace **port** in this command with the HTTPConnector **port** value that the **mqsireportporties** command returns.

For example, if the port number that the **mqsireportproperties** command returns in step 9 is 4417, type **http://localhost:4417**.

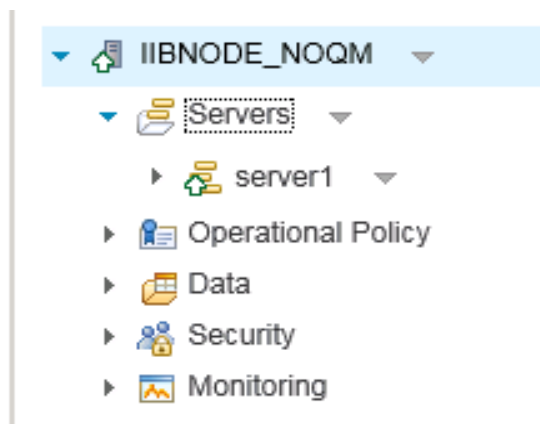
- \_\_\_ c. After a brief pause, the IBM Integration user interface for the IIBNODE\_NOQM integration node starts.



- \_\_\_ 13. Display the current integration servers that are defined on the integration node.

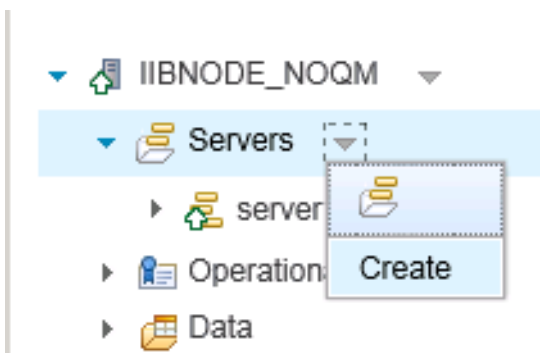
Click the down arrow that is to the left of the **Servers** folder.

You should see one integration server that is named **server1**. You should also see that icon next to the integration server name contains an up pointing green arrow, which indicates that the integration server is running.

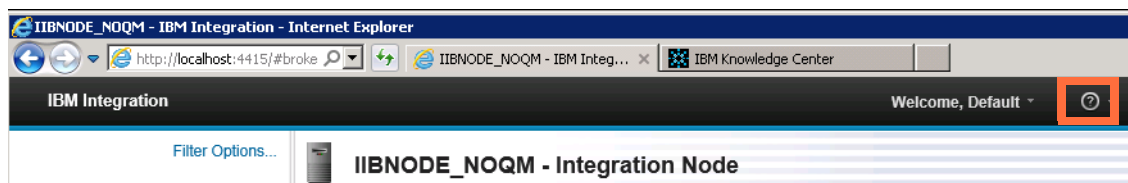


- \_\_\_ 14. Create an integration server that is named **server2** on the integration node by using the IBM Integration web interface.

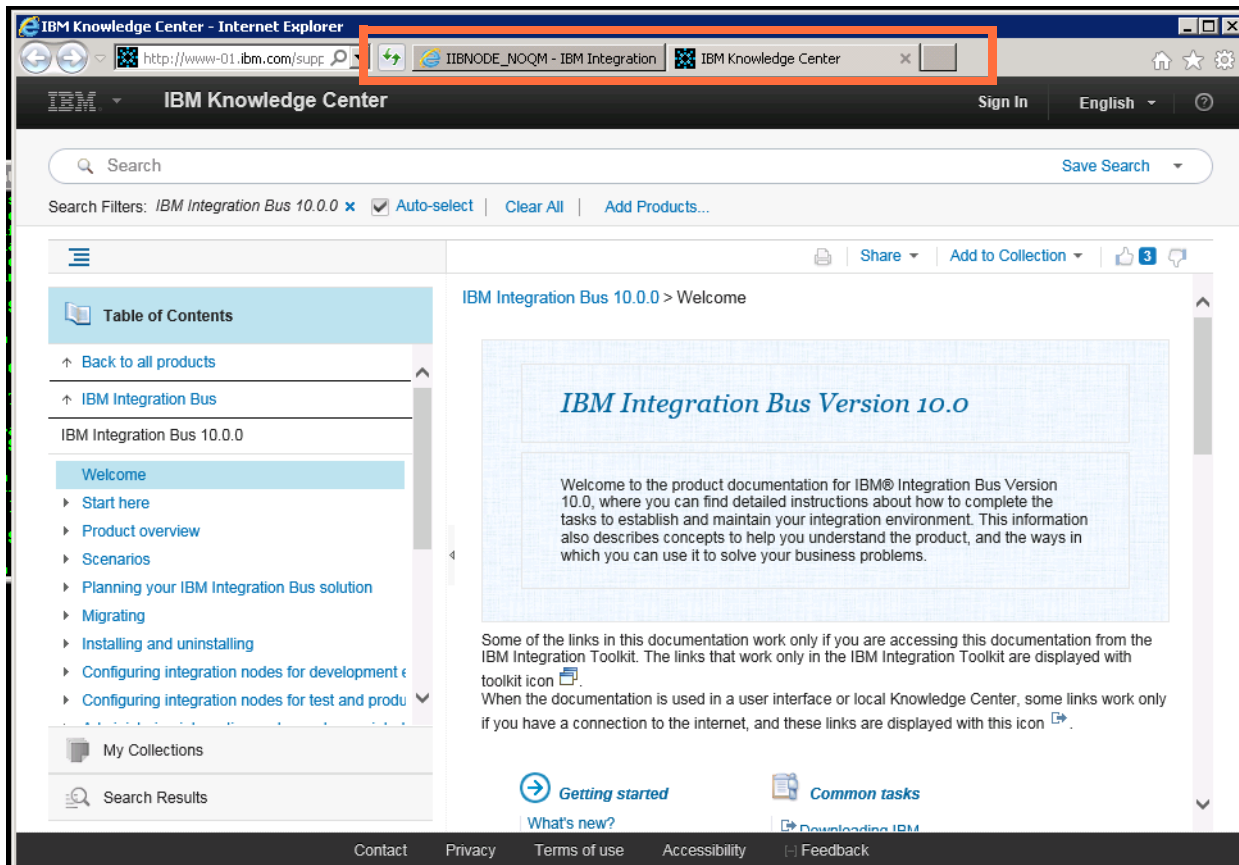
- \_\_\_ a. In the IBM Integration web interface, click the down arrow to the right of the **Servers** folder and then click **Create**.



- \_\_\_ b. In the New Integration Server window, type **server2** and then click **OK**.
- \_\_\_ c. After a brief pause, you should see a message that the integration server was created. You should also see that the **Servers** folder now contains two integration servers: **server1** and **server2**.
- \_\_\_ 15. Start the IBM Knowledge Center for IBM Integration Bus V10 from the IBM Integration web user interface.
- \_\_\_ a. Click the **Help** icon in the IBM Integration web interface.



- \_\_\_ b. The IBM Knowledge Center web page opens in a new tab in the web browser.



- \_\_\_ c. Take some time to investigate the sections in the IBM Knowledge Center for IBM Integration Bus.
- For example, you can type `mqsicreatebroker` in the **Search** area to find information about the `mqsicreatebroker` command.

## Exercise clean-up

- \_\_\_ 1. Close the web browser window.
- \_\_\_ 2. Close the IBM Integration Console window.

## End of exercise



## Exercise review and wrap-up

In the first part of this exercise, you verified the maintenance (fix pack) levels of the IBM Integration Bus software.

In this second part of the exercise, you verified that the user ID that is used in the lab exercises has administrative privileges to create and administer IBM Integration Bus and IBM MQ resources.

In the third part of this exercise, you used the IBM Integration command interface to create an integration node and report integration node properties. You then used the IBM Integration command interface and the IBM Integration web interface to create integration servers on the new integration node. You also used the IBM Integration web interface to access the IBM Knowledge Center for IBM Integration Bus V10.

Having completed this exercise, you should be able to:

- Get build and version information for IBM Integration Bus components
- Verify user authorizations
- Create and verify an integration node and integration server
- Start the IBM Integration web user interface



## Exercise 2. Connecting to IBM MQ

### What this exercise is about

In this exercise, you create an IBM MQ queue manager and an IBM Integration Bus integration node that uses the queue manager as a default queue manager. You create a second integration node with the same default queue manager as the first integration node. Finally, you create the IBM Integration Bus system queues on the queue manager and an MQEndpoint policy.

### What you should be able to do

After completing this exercise, you should be able to:

- Create an integration node that uses a default IBM MQ queue manager
- Share a default IBM MQ queue manager with multiple integration nodes
- Create the IBM Integration Bus system queues on a queue manager
- Create an MQEndpoint policy

### Introduction

With the release of IBM Integration Bus V10, an IBM MQ queue manager is no longer a prerequisite for integration nodes.

You can develop and deploy many message flow applications with Integration Bus independently of IBM MQ; however, some Integration Bus capabilities still require access to an IBM MQ queue manager. For example, the MQInput and MQOutput nodes in a message flow require access to an IBM MQ queue manager. Other Integration Bus capabilities require that a queue manager that has special SYSTEM.BROKER queues is specified on the integration node. For example, event driven message processing nodes such as the Collector node and the Aggregation node require that a queue manager is specified on the integration node.



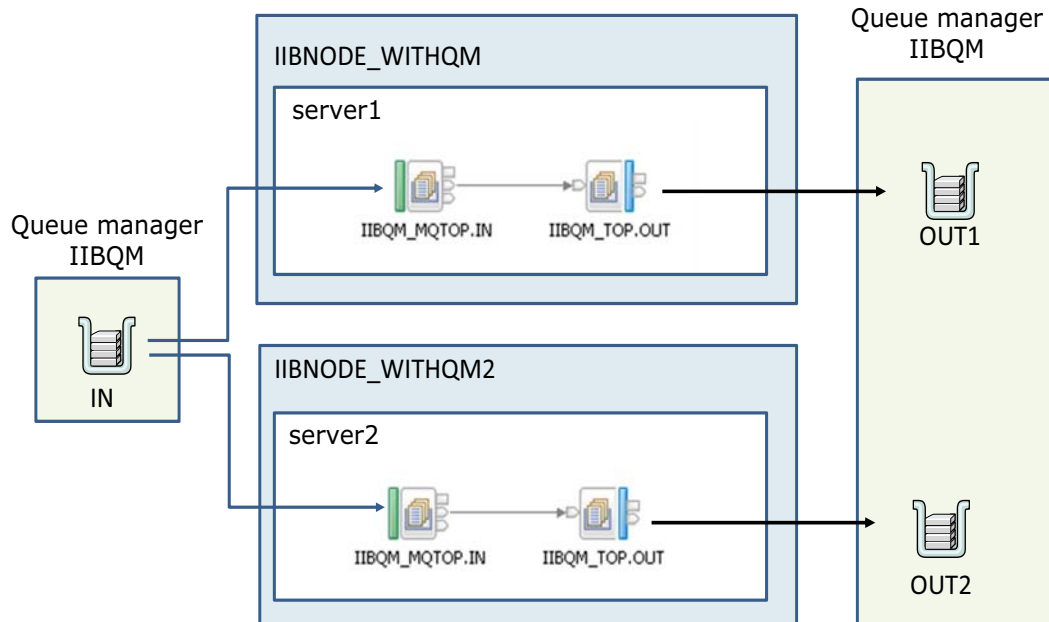
#### Information

The IBM MQ message processing nodes do not require a queue manager to be specified on the integration node unless you want to use a default queue manager for the local IBM MQ connection.

In IBM Integration Bus V10, multiple integration nodes can share a queue manager. Flows that are deployed to integration servers in different

integration nodes that share a queue manager can put and get from different queues or the same queues.

Using a shared queue manager can provide high availability for an application. If one of the flows, integration server, or integration node fails, the other flow on the other integration node can assume the workload. In the example in the figure, the integration nodes IIBNODE\_WITHQM and IIBNODE\_WITHQM2 share the IBM MQ queue manager IIBQM.



In the first part of this exercise, you create a queue manager that is named IIBQM.

In the second part of this exercise, you create two integration nodes (IIBNODE\_WITHQM and IIBNODE\_WITHQM2). Each integration node specifies the same queue manager (IIBQM) as the default queue manager. You also create an integration server on each integration node.

In the third part of this exercise, you add the BROKER.SYSTEM queues to the queue manager.

Integration Bus administrators can use an MQEndpoint policy to control the values of IBM MQ node connection properties at run time. The MQEndpoint policy overrides any equivalent properties that are specified for the **MQ Connection** properties in the message flow definition.

In the fourth part of the exercise, you create an MQEndpoint policy on the development integration node TESTNODE\_iibadmin by using the IBM Integration web interface. The TESTNODE\_iibadmin integration node is the development integration node that is created automatically for developers the first time that the IBM Integration Toolkit starts.

In this part of the exercise, you also export an MQEndpoint policy and then import it into another integration node (IIBNODE\_NOQM).

## Requirements

- A lab environment with the IBM Integration Bus V10 Bus and IBM MQ V8
- A user **iibadmin** who is a member of the **mqbrkrs** and **mqm** groups
- The IIBNODE\_NOQM integration node and integration servers that are created in Exercise 1, Part 3. It also requires that the integration node IIBNODE\_NOQM is running.

## Exercise instructions

### Part 1: Use IBM MQ Explorer to create the queue manager

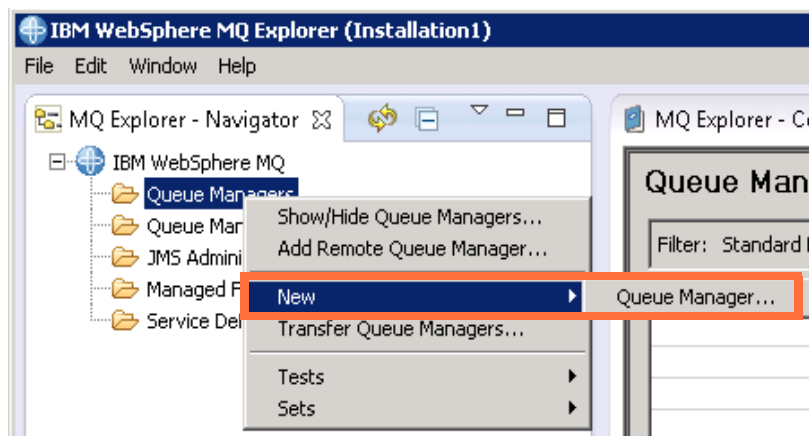
In this part of the exercise, you use IBM MQ Explorer to create a queue manager that is named IIBQM and a dead-letter queue that is named DLQ.

- \_\_\_ 1. Start IBM MQ Explorer.

From the Windows **Start** menu, click **All Programs > IBM WebSphere MQ > WebSphere MQ (Installation 1)** or double-click the **WebSphere MQ Explorer (Installation1)** icon on the desktop.

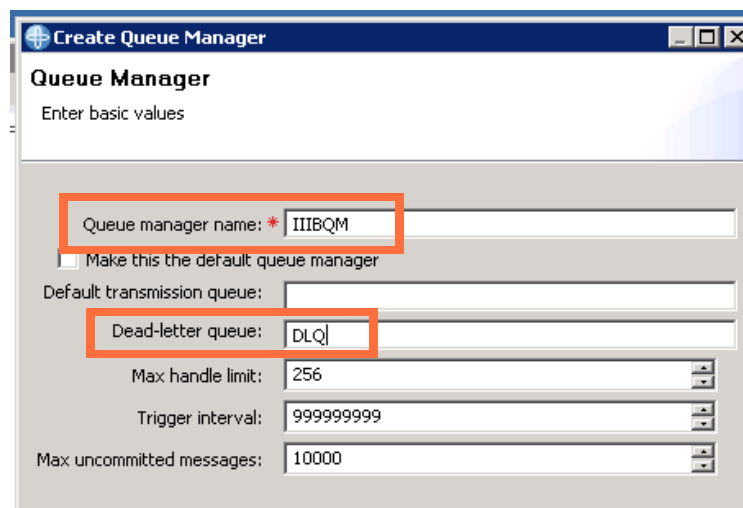
- \_\_\_ 2. Create a queue manager that is named IIBQM with a dead-letter queue that is named DLQ.

- \_\_\_ a. In the **MQ Explorer Navigator** view, right-click **Queue Managers** and then click **New > Queue Manager**.



- \_\_\_ b. For the **Queue Manager** name, type: IIBQM

- \_\_\_ c. For the **Dead-letter queue**, type: DLQ

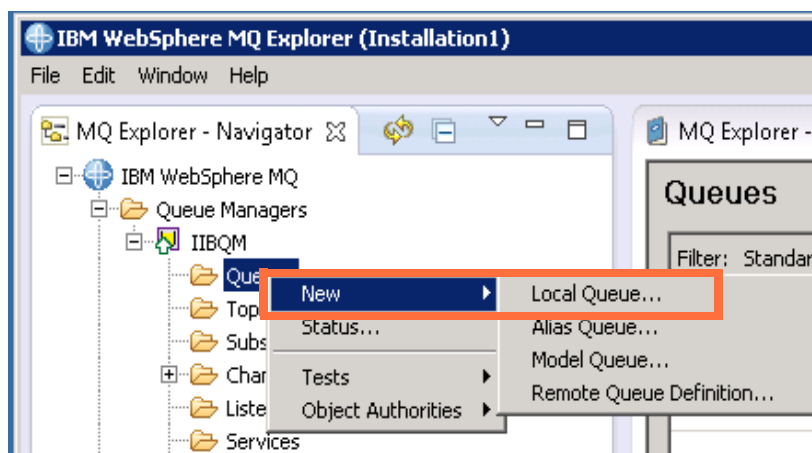


- \_\_\_ d. Click **Finish**.

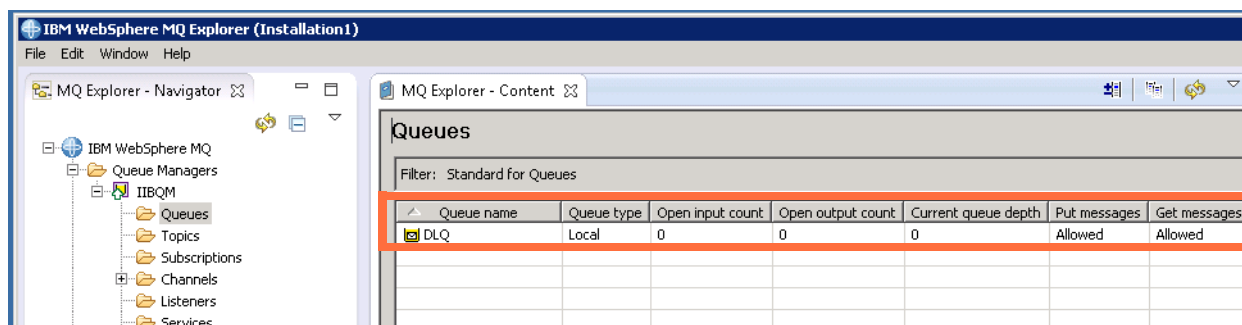
After a brief pause, the queue manager is created and started. A listener is also started on the default TCP port of 1414.

The queue manager is shown under the **Queue Managers** folder in the **MQ Explorer - Navigator** view.

- \_\_\_ 3. Create a queue that is named `DLQ` for the dead-letter queue.
  - \_\_\_ a. In the **MQ Explorer - Navigator** view, expand the **IIBQM** queue manager folder.
  - \_\_\_ b. Right-click **Queues** and then click **New > Local Queue**.



- \_\_\_ c. For the queue **Name**, type: `DLQ`
- \_\_\_ d. Click **Finish**.
- \_\_\_ e. Click **OK** in the Confirmation window.
- \_\_\_ f. Verify that the queue `DLQ` is listed in the **Queues** content view.



- \_\_\_ 4. Minimize IBM MQ Explorer.

## Part 2: Create the integration nodes and integration servers

In this part of the exercise, you use the IBM Integration Bus command interface to create two integration nodes that use the same queue manager as the default queue manager. You also create an integration server on each integration node.

- \_\_\_ 1. Create an integration node that is named `IIBNODE_WITHQM`.
  - \_\_\_ a. Start an IBM Integration Console session with Windows "Administrator" privileges.

Click **Start > All Programs > IBM Integration Bus 10.0.0.0** and then right-click **IBM Integration Console 10.0.0.0**, and then click **Run as administrator**.

Optionally, you can right-click the **IBM Integration Console** icon on the desktop and then click **Run as administrator**.

Click **Yes** to run the IBM Integration Console with “Administrator” privileges.

- \_\_ b. Enter the following command to create an integration node that is named IIBNODE\_WITHQM that specifies the queue manager IIBQM:

```
mqsicreatebroker IIBNODE_WITHQM -q IIBQM
```

- \_\_ c. Start the new integration node.

In the IBM Integration Console, type:

```
mqsistart IIBNODE_WITHQM
```

You should receive a response similar to the following example.

```
BIP8071II: Successful command initiation, check the system log to ensure
that the component started without problems and that it continues to run
without problems.
```

- \_\_ 2. Check that the integration node environment is set up correctly for IIBNODE\_WITHQM. Verify that the integration node properties for the integration node indicate that queue manager IIBQM is specified on the integration node.

In the IBM Integration Console, type:

```
mqsicvp IIBNODE_WITHQM
```

You should receive a response similar to the following example.

```
BIP8873I: Starting the component verification for component
'IIBNODE_WITHQM'.
BIP8876I: Starting the environment verification for component
'IIBNODE_WITHQM'.
BIP8894I: Verification passed for 'Registry'.
BIP8894I: Verification passed for 'MQSI_REGISTRY'.
BIP8894I: Verification passed for 'Java Version - 1.7.0 IBM Windows AMD 64
build pwa6470_27sr2fp10-20141218_02(SR2 FP10)
BIP8894I: Verification passed for 'MQSI_FILEPATH'.
BIP8878I: The environment verification for component 'IIBNODE_WITHQM' has
finished successfully.
BIP8882I: Starting the WebSphere MQ verification for component
'IIBNODE_NOQM'.
BIP8841I: The WebSphere MQ verification for component 'IIBNODE_WITHQM' had
finished successfully.
BIP8874I: The component verification for 'IIBNODE_WITHQM' has finished
successfully.
```

- \_\_ 3. Create an integration server on IIBNODE\_WITHQM that is named server1.



In the IBM Integration Console, type:

```
mqsicreateexecutiongroup IIBNODE_WITHQM -e server1
```

You should receive a response that is similar to the following example.

```
BIP1124I: Creating integration server 'server1' on integration node  
'IIBNODE_WITHQM'
```

```
BIP1117I: Integration server was created successfully.
```

```
The integration node has initialized the integration server.
```

- \_\_\_ 4. Get the IBM Integration web interface port number for the IIBNODE\_WITHQM integration node.
  - \_\_\_ a. In the IBM Integration Console, type the following command to get the HTTP Connection port number:

```
mqsireportproperties IIBNODE_WITHQM -b webadmin -o HTTPConnector -a
```
  - \_\_\_ b. Record the value for HTTPConnector **port**. You use this value in the next step to start the IBM Integration web interface for this integration node.



### Information

IBM Integration Bus assigns a web user administration port number dynamically. When the first integration node is started, it is assigned the default port of 4414. For subsequent integration nodes, the port number increments and the node is assigned the next port number in sequence.

- \_\_\_ 5. Start the IBM Integration web interface for the new integration node.
  - \_\_\_ a. Open a web browser.
  - \_\_\_ b. For the URL, type:

```
http://localhost:port
```

Replace **port** in this command with the HTTPConnector **port** value that the `mqsireportproperties` command returns.

For example, if the port number that the `mqsireportproperties` command returns in Step 4 is 4416, type `http://localhost:4416`.
  - \_\_\_ c. After a brief pause, the IBM Integration user interface for the IIBNODE\_NOQM integration node starts.
- \_\_\_ 6. Following the same procedure that you used for creating the integration node IIBNODE\_WITHQM and the integration server **server1**, create another integration node that is named IIBNODE\_WITHQM2 and an integration server that is named **server2**.

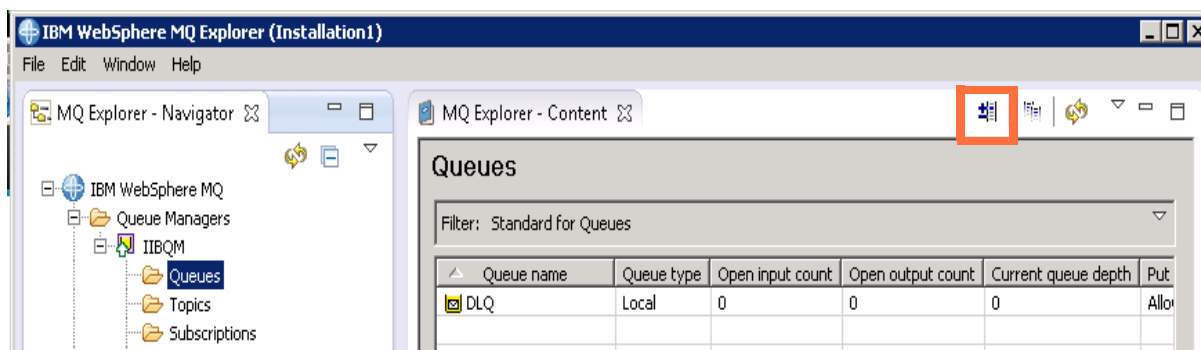
Be sure to specify the same queue manager, IIBQM, as the queue manager on the integration node.

### Part 3: Create the **SYSTEM.BROKER** queues and topics on the integration node queue manager

Some Integration Bus capabilities and message processing nodes require access to an IBM MQ queue manager and **SYSTEM.BROKER** queues and topics on the integration node queue manager.

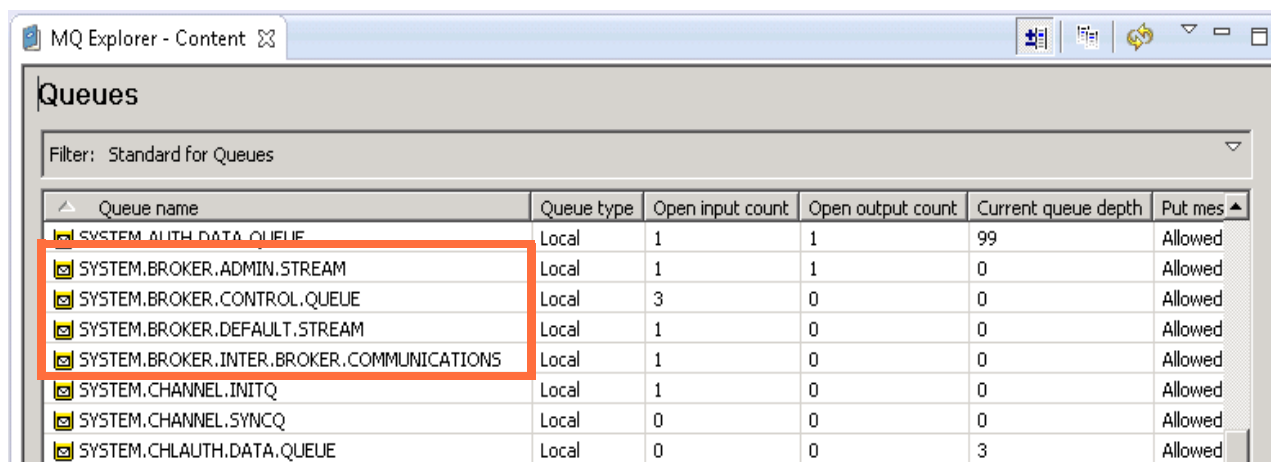
In this part of the exercise, you use an Integration Bus sample command file to add the **SYSTEM.BROKER** queues and topics to the IIBQM queue manager that the IINODE\_WITHQM and IIBNODE\_WITHQM2 integration nodes use.

- \_\_\_ 1. In IBM MQ Explorer, check the **SYSTEM** queues on the queue manager IIBQM.
  - \_\_\_ a. In the IBM MQ Explorer **Queues** content view, click the **Show System Objects** icon to display the **SYSTEM** queues and topics.



- \_\_\_ b. Locate the current set of **SYSTEM.BROKER** queues.

IBM MQ created these queues. They are generic queues that are automatically set up for all brokers and queue managers on a network. These queues are not specific to Integration Bus.



- \_\_\_ 2. In the IBM Integration Console, type the following command to go to the Integration Bus sample directory for IBM MQ. This directory contains the batch file that creates the **SYSTEM.BROKER** queues.

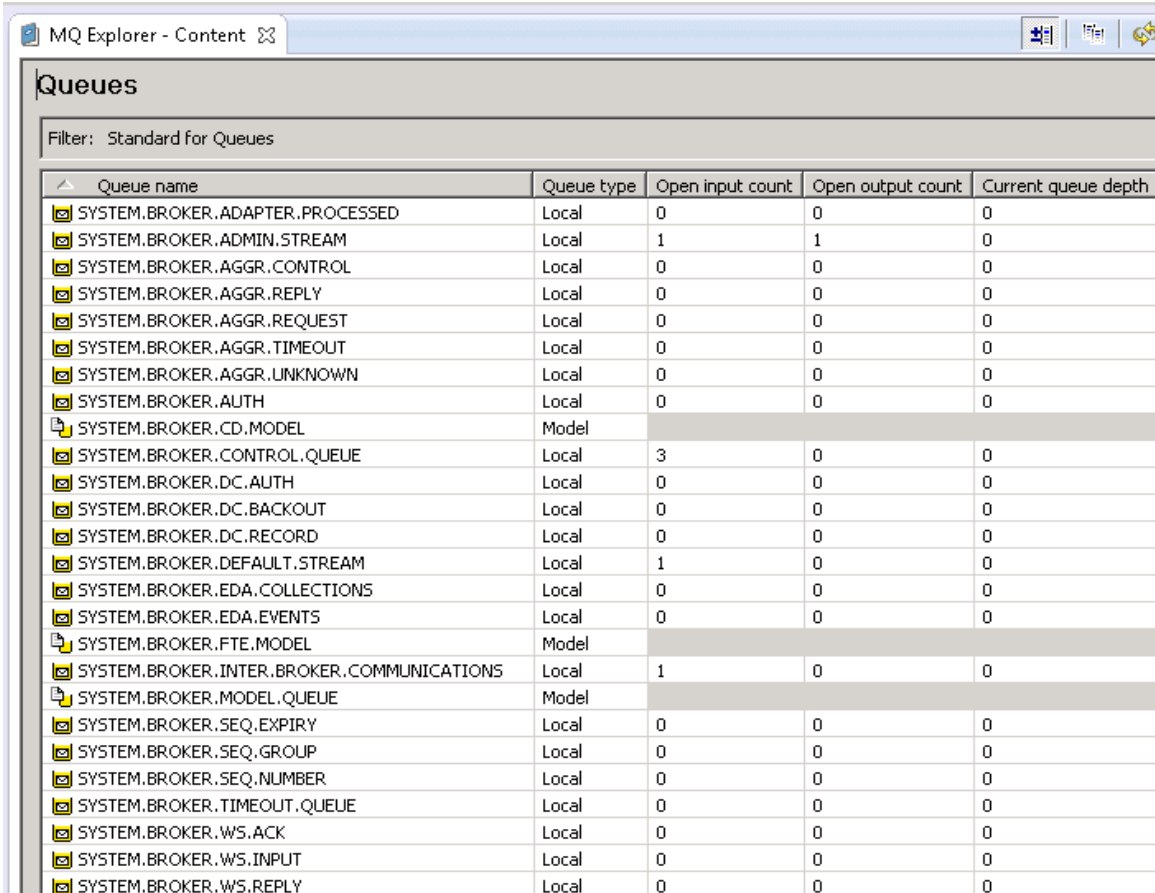
```
cd server\sample\wmq
```

- \_\_\_ 3. In the IBM Integration Console, type the following command to run the command file on the IIBQM queue manager.

```
iib_createqueues.cmd IIBQM
```

You should see messages that indicate that the SYSTEM.BROKER queues and topics are created. You should also see messages that indicate that security is set on the queues by using the IBM MQ setmqaut command.

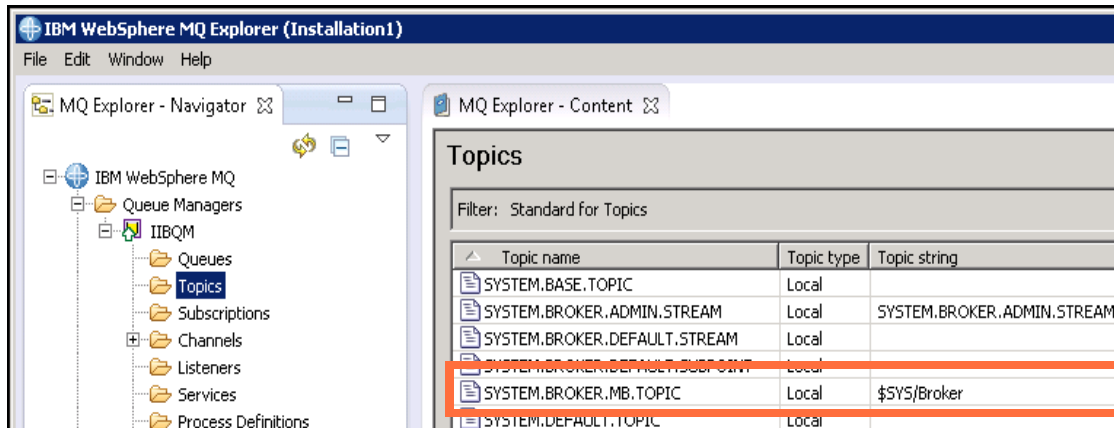
- \_\_\_ 4. In IBM MQ Explorer, check the SYSTEM queues on the queue manager IIBQM and verify that the queue manager now contains the Integration Bus SYSTEM.BROKER queues.



Queue name	Queue type	Open input count	Open output count	Current queue depth
SYSTEM.BROKER.ADAPTER.PROCESSED	Local	0	0	0
SYSTEM.BROKER.ADMIN.STREAM	Local	1	1	0
SYSTEM.BROKER.AGGR.CONTROL	Local	0	0	0
SYSTEM.BROKER.AGGR.REPLY	Local	0	0	0
SYSTEM.BROKER.AGGR.REQUEST	Local	0	0	0
SYSTEM.BROKER.AGGR.TIMEOUT	Local	0	0	0
SYSTEM.BROKER.AGGR.UNKNOWN	Local	0	0	0
SYSTEM.BROKER.AUTH	Local	0	0	0
SYSTEM.BROKER.CD.MODEL	Model			
SYSTEM.BROKER.CONTROL.QUEUE	Local	3	0	0
SYSTEM.BROKER.DC.AUTH	Local	0	0	0
SYSTEM.BROKER.DC.BACKOUT	Local	0	0	0
SYSTEM.BROKER.DC.RECORD	Local	0	0	0
SYSTEM.BROKER.DEFAULT.STREAM	Local	1	0	0
SYSTEM.BROKER.EDA.COLLECTIONS	Local	0	0	0
SYSTEM.BROKER.EDA.EVENTS	Local	0	0	0
SYSTEM.BROKER.FTE.MODEL	Model			
SYSTEM.BROKER.INTER.BROKER.COMMUNICATIONS	Local	1	0	0
SYSTEM.BROKER.MODEL.QUEUE	Model			
SYSTEM.BROKER.SEQ.EXPIRY	Local	0	0	0
SYSTEM.BROKER.SEQ.GROUP	Local	0	0	0
SYSTEM.BROKER.SEQ.NUMBER	Local	0	0	0
SYSTEM.BROKER.TIMEOUT.QUEUE	Local	0	0	0
SYSTEM.BROKER.WS.ACK	Local	0	0	0
SYSTEM.BROKER.WS.INPUT	Local	0	0	0
SYSTEM.BROKER.WS.REPLY	Local	0	0	0

- \_\_\_ 5. The command script also creates the topic SYSTEM.BROKER.MB.TOPIC, which maps to the topic string \$SYS/Broker. This topic is required for the Integration Bus record and replay, and statistics components.

Click **Topics** under the IIBQM queue manager in the **MQ Explorer - Navigator**. Verify that the **SYSTEM.BROKER.MB.TOPIC** is created on the queue manager.



\_\_\_ 6. Minimize IBM MQ Explorer.

## Part 4: Create an MQEndpoint policy

Integration Bus administrators can use an MQEndpoint policy to control the values of IBM MQ node connection properties at run time. The MQEndpoint policy overrides any equivalent properties that are specified for the **MQ Connection** properties in the message flow definition.

In this part of the exercise, you create an MQEndpoint policy on the development integration node **TESTNODE\_iibadmin** by using the IBM integration web interface. The **TESTNODE\_iibadmin** integration node is the development integration node that is created automatically for developers the first time that the IBM Integration Toolkit starts.

In this part of the exercise, you also export an MQEndpoint policy and then import it into another integration node (**IIBNODE\_NOQM**).

\_\_\_ 1. Open a web browser and start the IBM Integration web interface for the **TESTNODE\_iibadmin** integration node.

\_\_\_ a. Ensure that the integration node **TESTNODE\_iibadmin** is running. Type:

```
mqsilist
```

If the results of this command show that **TESTNODE\_iibadmin** is not running, start it by using the **mqsisstart** command.

\_\_\_ b. Get the HTTP Connector port value for the web interface by typing the following command in the IBM Integration Console.

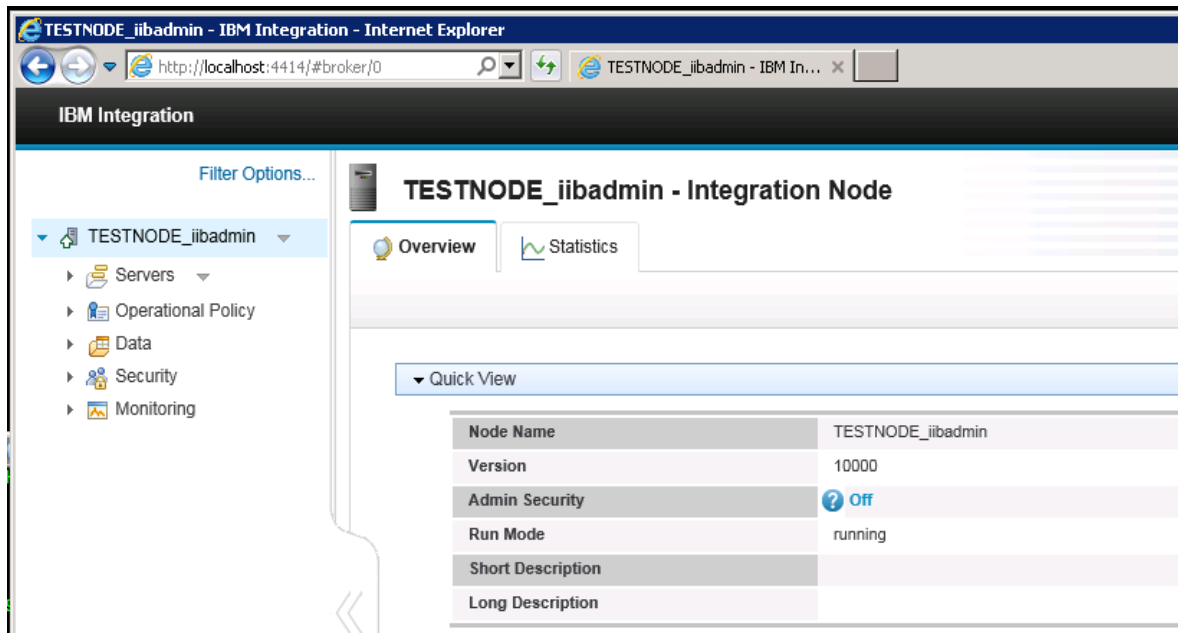
```
mqsireportproperties TESTNODE_iibadmin -b webadmin -o HTTPConnector -a
```

\_\_\_ c. Record the value for port property.

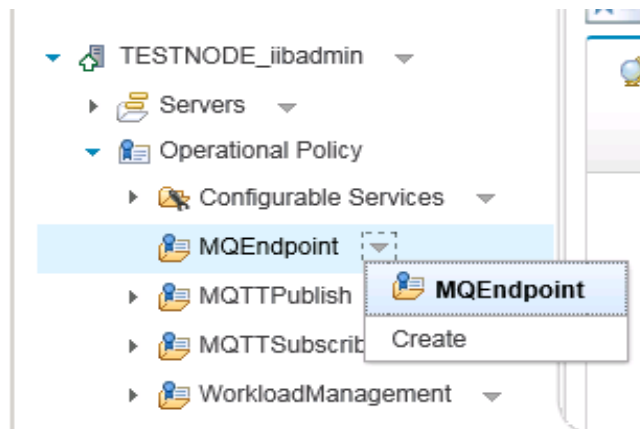
\_\_\_ d. In the web browser, type the following URL to start the IBM Integration web interface for **TESTNODE\_iibadmin**

```
http://localhost:port
```

Replace **port** in this command with the HTTPConnector **port** value that the **mqsireportporties** command returns.



- \_\_\_ 2. In the IBM Integration web interface, expand **Operational Policy**.
- \_\_\_ 3. Click the down arrow on the right side of **MQEndpoint** to display the menu and then click **Create**.



- \_\_\_ 4. Define the MQEndpoint policy for a local connection to the queue manager IIBQM.
  - \_\_\_ a. For the **Policy name**, type: **LocalConnectToIIBQM**
  - \_\_\_ b. For the **Connection**, select **Local queue manager**.
  - \_\_\_ c. For the **Queue manager name**, type: **IIBQM**
  - \_\_\_ d. For the **Queue manager host name**, type: **localhost**
  - \_\_\_ e. For the **Listener port number**, type: **1414**
  - \_\_\_ f. For the **Channel name**, type: **SYSTEM.DEF.SVRCONN**

**Operational Policy - MQEndpoint**

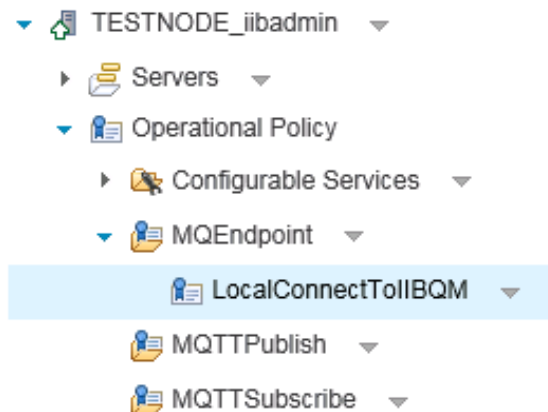
Overview

Save Save As Revert

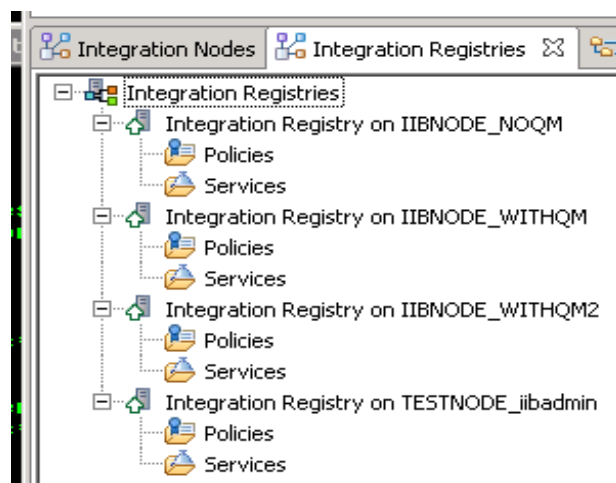
Use a policy to control the operational behavior of a message flow node at run time... Oct 8, 2015, 6:55:29 AM x

Policy name*	LocalConnectToIIBQM
Connection	Local queue manager
Queue manager name	IIBQM
Queue manager host name	localhost
Listener port number	1414
Channel name	SYSTEM.DEF.SVRCONN
Security identity	
Use SSL	<input type="checkbox"/>
SSL peer name	
SSL cipher specification	

- \_\_\_ g. Click **Save**.
- \_\_\_ h. After a brief pause, you should see the new MQ Endpoint policy under the **MQEndpoint** policy folder in the IBM Integration web interface.



- \_\_\_ 5. Verify that the new policy is available to the developer in the Integration Toolkit.
- \_\_\_ a. Start the IBM Integration Toolkit if it is not already running.
- \_\_\_ b. In the Integration Toolkit, click the **Integration Registries** tab (in the same view as the **Integration Nodes** tab).

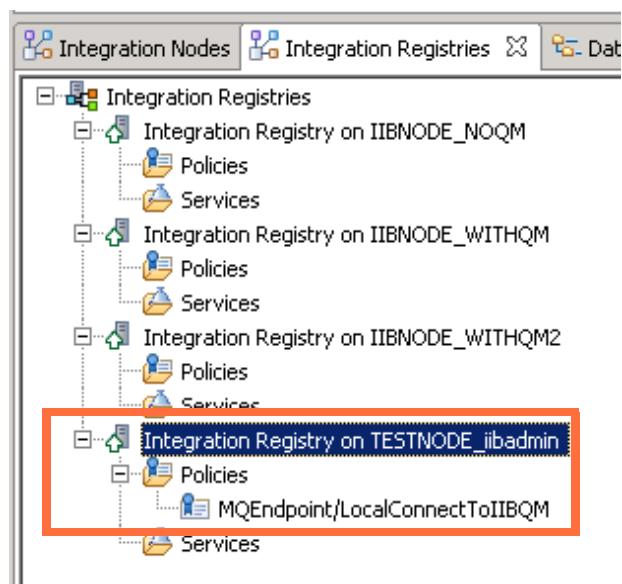


The **Integration Registries** tab contains an entry for each integration node to which the Integration Toolkit can connect. By default, the Integration Toolkit can connect to any local integration node.

- \_\_\_ c. You created the MQEndpoint policy on TESTNODE\_iibadmin.

Right-click **Integration Registry on TESTNODE\_iibadmin** and then click **Refresh** to update the Integration Registry view in the IBM Integration Toolkit.

- \_\_\_ d. Verify that the MQEndpoint policy that is named **LocalConnectToIIBQM** is listed under the **Integration Registry on TESTNODE\_iibadmin** folder.



### Information

The application developer can now reference the MQEndpoint policy in the message flow by specifying the policy URL in the **Policy URL** field in the **Policy** properties on the IBM MQ message processing nodes.

The format of a policy URL is `/apiv1/policy/policyType/policyName`.

For example, the policy URL for the MQEndpoint policy that you created in this exercise is:  
`/apiv1/policy/MQEndpoint/LocalConnectToIIBQM`.

- \_\_\_ 6. Export the MQEndpoint policy to the `C:\labfiles\Lab02-MQ` directory.
- \_\_\_ a. In the IBM Integration Console, go to the `C:\labfiles\Lab02-MQ` directory. Type:
- ```
cd C:\labfiles\Lab02-MQ
```
- \_\_\_ b. Type the following command to create a policy file from the **LocalConnectToIIBQM** MQEndpoint policy that was created on TESTNODE\_iibadmin:

```
mqsiexportpolicy TESTNODE_iibadmin -t MQEndpoint -l LocalConnectToIIBQM  
-f LocalConnectToIIBQM.xml
```

The command should create the file and display the current values of the MQEndpoint policy.

\_\_\_ c. Using Windows Explorer, verify that the **LocalConnectToIIBQM.xml** file was created in the `C:\labfiles\Lab02-MQ` directory.

\_\_\_ 7. View the contents of the **LocalConnectToIIBQM.xml** file.

You can double-click the file to view it in a web browser (shown in the screen capture) or open Windows Notepad to view it as a text file.

```
<?xml version="1.0" encoding="UTF-8" standalone="true"?>  
- <policy type="MQEndpoint">  
  - <policyProperties>  
    - <mqConnectionDetailsPolicy>  
      <connection>SERVER</connection>  
      <destinationQueueManagerName>IIBQM</destinationQueueManagerName>  
      <queueManagerHostname>localhost</queueManagerHostname>  
      <listenerPortNumber>1414</listenerPortNumber>  
      <channelName>SYSTEM.DEF.SVRCONN</channelName>  
      <useSSL>>false</useSSL>  
    </mqConnectionDetailsPolicy>  
  </policyProperties>  
</policy>
```

- \_\_\_ 8. Import the exported MQEndpoint policy definition to create an MQEndpoint Policy on IIBNODE\_NOQM.

In the IBM Integration Console, type:

```
mqsicreatepolicy IIBNODE_NOQM -t MQEndpoint -l LocalConnectToIIBQM -f  
LocalConnectToIIBQM.xml
```

After a brief pause, you should receive a response that the command completed successfully.

- \_\_\_ 9. Verify that the MQEndpoint policy was imported successfully by using the **mqsiexportpolicy** command.



In the IBM Integration Console, type:

```
mqsireportpolicy IIBNODE_NOQM -t MQEndpoint -r
```

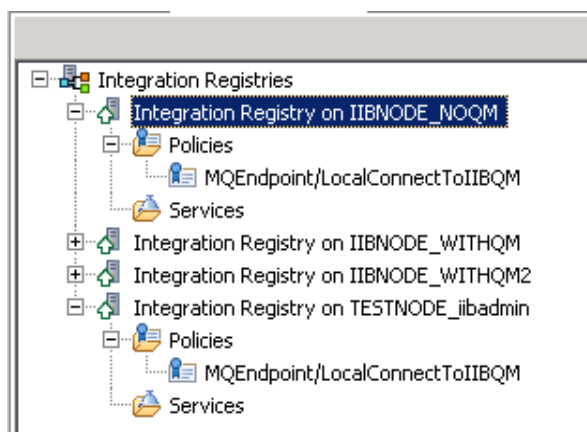
You should receive the following response.

```
BIP1895I: Policy type 'MQEndpoint' Policy name 'LocalConnectToIIBQM' Policy
URI '/apiv1/policy/MQEndpoint/LocalConnectToIIBQM' Policy content '<?xml
version="1.0" encoding="UTF-8" standalone="yes"?><policy type="MQEndpoint">
<policyProperties>          <mqConnectionDetailsPolicy>
<connection>SERVER</connection>
<destinationQueueManagerName>IIBQM</destinationQueueManagerName>
<queueManagerHostname>localhost</queueManagerHostname>
<listenerPortNumber>1414</listenerPortNumber>
<channelName>SYSTEM.DEF.SVRCONN</channelName>
<useSSL>>false</useSSL>          </mqConnectionDetailsPolicy>
</policyProperties></policy>'
BIP8071I: Successful command completion.
```



### Note

You can also verify that the MQEndpoint policy was imported into the IIBNODE\_NOQM integration node by using the IBM Integration web interface for IIBNODE\_NOQM or by viewing the Integration Registry in the IBM Integration Toolkit.



## Exercise clean up

1. Stop the IIBNODE\_WITHQM2 integration node. In the Integration Console, type:

```
mqsistop IIBNODE_WITHQM2
```

## End of exercise

## Exercise review and wrap-up

In the first part of this exercise, you created a queue manager that is named IIBQMGR.

In the second part of this exercise, you created two integration nodes (IIBNODE\_WITHQM and IIBNODE\_WITHQM2). Each integration node specifies the same queue manager (IIBQMGR) as the default queue manager. You also created an integration server on each integration node.

In the third part of this exercise, you added the BROKER.SYSTEM queues to the queue manager.

In the fourth part of the exercise, you created an MQEndpoint policy on the development integration node TESTNODE\_iibadmin by using the IBM Integration web interface. You also exported an MQEndpoint policy and then imported it into another integration node.

Having completed this exercise, you should be able to:

- Create an integration node that uses a default IBM MQ queue manager
- Share a default IBM MQ queue manager with multiple integration nodes
- Create the IBM Integration Bus system queues on a queue manager
- Create an MQEndpoint policy

## Exercise 3. Using the IBM Integration Toolkit

### What this exercise is about

In this exercise, you use the IBM Integration Toolkit to complete basic configuration tasks. You also import, deploy, and test a message flow to learn the administrative tasks available in the development environment.

### What you should be able to do

After completing this exercise, you should be able to:

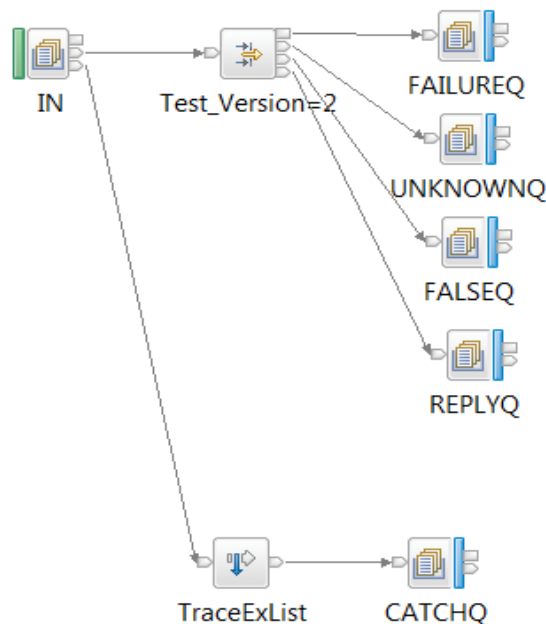
- Import, deploy, and test a message flow
- Use the BAR file editor to examine the BAR file contents

### Introduction

You typically administer IBM Integration Bus by using the IBM Integration web user interface, and the IBM Integration Console command interface. Some basic administration functions are also available in the IBM Integration Toolkit.

In this exercise, you use import, deploy, and test a message flow so that you can gain an understanding of the administrative tasks that a developer often does in a development environment.

In the first part of this exercise, you import an Integration Bus project interchange file into the Integration Toolkit. The project interchange file includes a message flow application. The message flow is shown in the following figure.



This message flow accepts a simple XML message from an IBM MQ queue of the form:

```
<InMsg><Version>2</Version></InMsg>
```

The message flow validates the value of `<Version>` in a Filter node as follows:

- If `<Version>` equals 2, send the message through the **True** terminal and put the message on the REPLYQ node, which puts the message on the REPLY queue.
- If `<Version>` equals 1 or 3, send the message through the **False** terminal to the FALSEQ node, which puts the message on the FALSE queue.
- If `<Version>` does not equal 1, 2, or 3, send the message through the **Unknown** terminal to the UNKNOWNQ node, which puts the message on the UNKNOWN queue.
- If an error occurs during message processing on the Filter node, the message is sent through the terminal to the FAILUREQ node, which puts the message on the FAILURE queue. If the put to the FAILURE queue fails, the message is routed back to the IN node and sent to the TraceExtList node.



#### Note

You learn more about the handling of message flow processing errors later in this course.

In the second part of this exercise, you create a BAR file and examine the configurable properties by using the Integration Toolkit BAR file editor. You then deploy the BAR file to the development integration server.

In the third part of this exercise, you test the message flow by using the Integration Toolkit Flow exerciser.

## Required materials

- IBM MQ and IBM Integration Bus software components
- IIBQM queue manager that you created in Exercise 2
- Lab exercise files in the `C:\labfiles\Lab03-Tlkt` folder.
- RFHUtil in the `C:\Software` folder

## Exercise instructions

### Exercise setup

- \_\_\_ 1. This exercise assumes that you successfully completed Part 1 of Exercise 2 in which you created the queue manager IIBQM.

If you did not do Part 1 of Exercise 2, complete that part of the exercise before proceeding with this exercise.

- \_\_\_ 2. Use the IBM MQ **runmqsc** command to define the queues that are required for this exercise. You use a command script for this purpose. The command file is in the **C:\labfiles\Lab03-Tlkt\resources** directory.

- \_\_\_ a. Start an IBM Integration Console session if one is not already running.

- \_\_\_ b. Change to the **C:\labfiles\Lab03-Tlkt\resources** directory.

Enter the command:

```
cd C:\labfiles\Lab03-Tlkt\resources
```

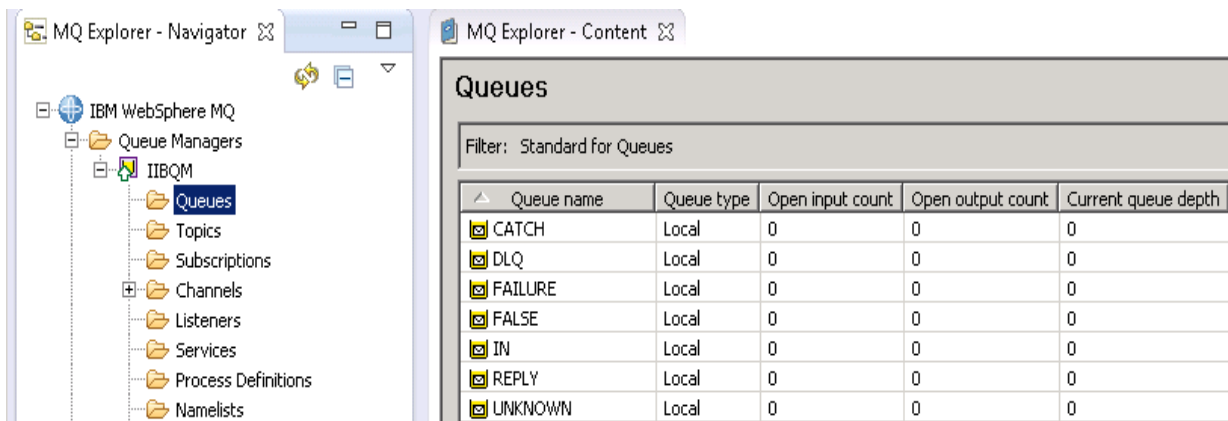
- \_\_\_ c. Enter the command:

```
runmqsc IIBQM < C:\labfiles\Lab03-Tlkt\resources\Q_Defs.mqsc
```

You should receive a response that indicates that the queues are created, similar to the following example.

```
1 : def ql(IN) replace
AMQ8006: WebSphere MQ queue created.
: *
2 : def ql(REPLY) replace
AMQ8006: WebSphere MQ queue created.
: *
3 : def ql(FAILURE) replace
AMQ8006: WebSphere MQ queue created.
: *
4 : def ql(UNKNOWN) replace
AMQ8006: WebSphere MQ queue created.
: *
5 : def ql(FALSE) replace
AMQ8006: WebSphere MQ queue created.
: *
6 : def ql(CATCH) replace
AMQ8006: WebSphere MQ queue created.
:
7 : end
6 MQSC commands read.
No commands have a syntax error.
All valid MQSC commands were processed.
```

- \_\_\_ d. Use the IBM MQ Explorer to verify that the local queues IN, REPLY, FAILURE, UNKNOWN, FALSE, and CATCH are created on IIBQM queue manager.

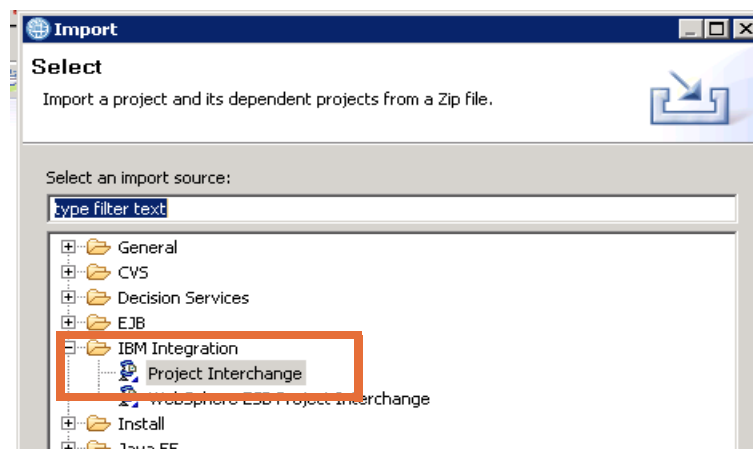


- \_\_\_ e. Minimize the IBM Integration Console and IBM MQ Explorer.

## Part 1: Work with a simple message flow

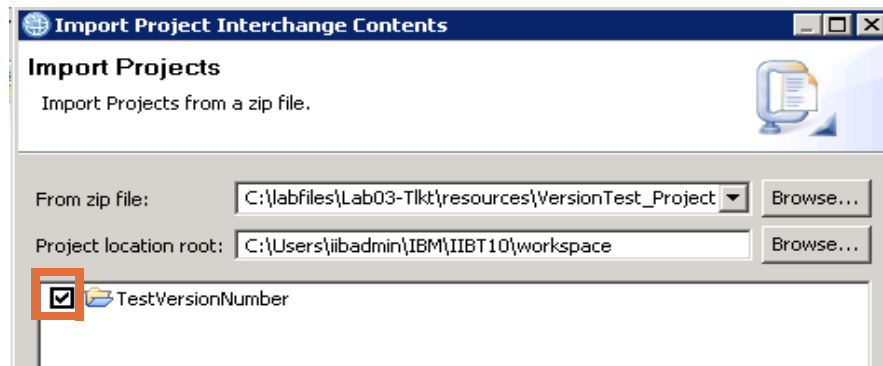
In this part of the exercise, you import an Integration Bus application into the Integration Toolkit workspace and examine the message flow.

- \_\_\_ 1. Start the IBM Integration Toolkit if it is not already running.
- \_\_\_ a. Double-click the **IBM Integration Toolkit 10.0.0.0** icon on the desktop.
- You can also start the toolkit by clicking **All Programs > IBM Integration Bus 10.0.0.0 > IBM Integration Toolkit 10.0.0.0**.
- \_\_\_ b. Close the Welcome window to go to the **Integration Development** perspective.
- \_\_\_ 2. Import an application from a project interchange file into the Integration Toolkit workspace.
- \_\_\_ a. From the Integration Toolkit menu bar, click **File > Import**.
- The Import window is displayed.
- \_\_\_ b. Verify that **IBM Integration > Project Interchange** is selected, and then click **Next**.



- \_\_\_ c. To the right of **From zip file**, click **Browse**.
- \_\_\_ d. Browse to the **C:\labfiles\Lab03-Tlkt\resources** directory.

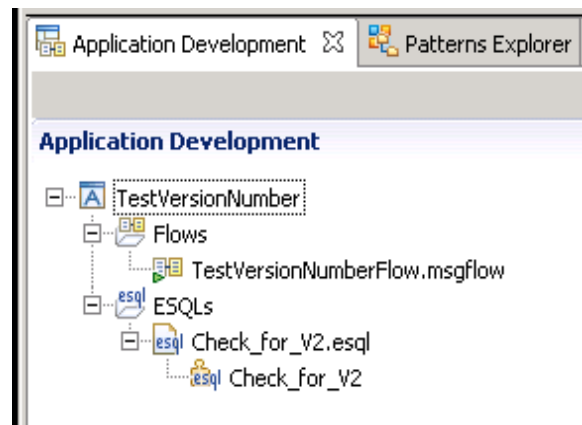
- \_\_\_ e. Click **VersionTest\_ProjectInterchange.zip** and then click **Open**.
- \_\_\_ f. Select the **TestVersionNumber** check box (if it is not already selected) and then click **Finish**.



The Integration Toolkit imports the application and constructs the workspace.

- \_\_\_ 3. The **TestVersionNumber** application is now displayed in the **Application Development** view.

Fully expand the **TestVersionNumber** application by clicking the plus (+) signs and examine its contents.



You see that the project contains folders for **Flows** and **ESQLs**.

- \_\_\_ 4. Double-clicking an object in the **Application Development** view causes the corresponding editor to open the object in the editor pane, on the right side of the Integration Toolkit.

Open the message flow in the Message Flow editor by double-clicking the file **TestVersionNumberFlow.msgflow** in the **Application Development** view.

- \_\_\_ 5. Examine the message flow. Verify that it agrees with the figure that is displayed at the beginning of the exercise.
- \_\_\_ 6. The properties of each node in the message flow are preset to transform and route the message as described at the beginning of the exercise.

Verify that the node properties match the following table.

You can view (or change) a property by right-clicking a node and then clicking **Properties**.



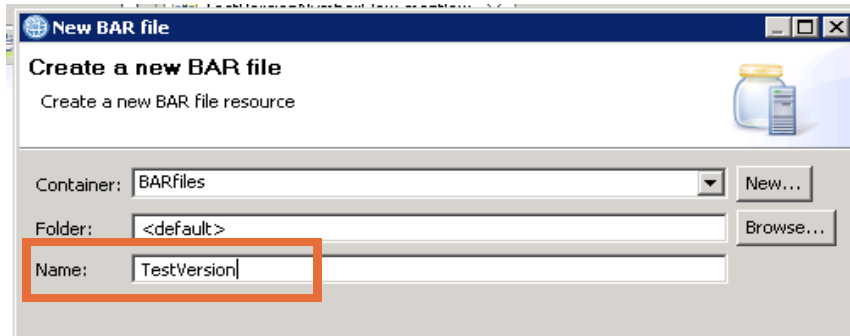
Node properties are grouped into pages. The Basic page is initially visible for most nodes.

Node name	Node type	Properties
IN	MQInput	<b>Basic:</b> Queue name = IN  <b>MQ Connection:</b> Connection = Local queue manager Destination queue manager name = IIBQM  <b>Input Message Parsing:</b> Message Domain = XMLNSC  <b>Advanced:</b> Transaction mode = Yes
TraceExList	Trace	<b>Basic:</b> Destination = File File path = C:\TraceExList.txt Pattern = In TraceExList! Root ==> \${Root} ExceptionList ==> \${ExceptionList}
REPLYQ	MQOutput	<b>Basic:</b> Queue name = REPLYQ  <b>MQ Connection:</b> Connection = Local queue manager Destination queue manager name = IIBQM
FAILUREQ	MQOutput	<b>Basic:</b> Queue name = FAILURE  <b>MQ Connection:</b> Connection = Local queue manager Destination queue manager name = IIBQM
Test_Version=2	Filter	<b>Basic:</b> Filter expression = {default}Check_for_V2
CATCHQ	MQOutput	<b>Basic:</b> Queue name = CATCH  <b>MQ Connection:</b> Connection = Local queue manager Destination queue manager name = IIBQM
UNKNOWNQ	MQOutput	<b>Basic:</b> Queue name = UNKNOWN  <b>MQ Connection:</b> Connection = Local queue manager Destination queue manager name = IIBQM
FALSEQ	MQOutput	<b>Basic:</b> Queue name = FALSE  <b>MQ Connection:</b> Connection = Local queue manager Destination queue manager name = IIBQM

## Part 2: Create and deploy a BAR file

In this part of the exercise, you create and deploy a BAR file in the Integration Toolkit. You also examine the configurable properties in the BAR file editor.

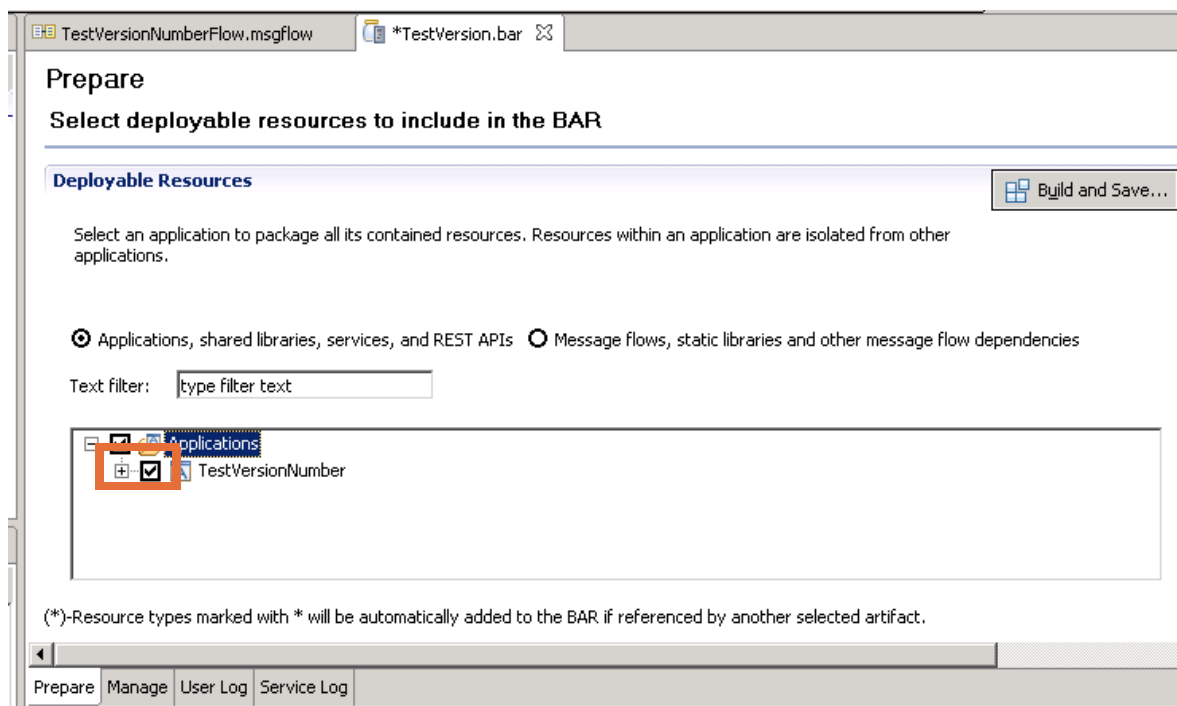
- \_\_\_ 1. In the Integration Toolkit, click **File > New > BAR file**.
- \_\_\_ 2. For the BAR file **Name**, type **TestVersion** and then click **Finish**.



The BAR File editor opens with the **Prepare** tab.

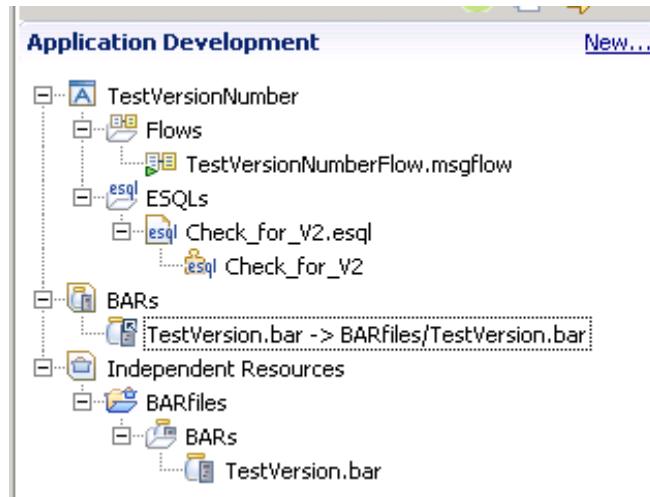
- \_\_\_ 3. On the **Prepare** tab, you identify the resources to include in the BAR file.

Click the **TestVersionNumber** application from the list of **Deployable Resources** to select it as a component of the BAR file.



- \_\_\_ 4. Click **Build and Save** to build the BAR file and save the workspace. Click **OK** in the confirmation window.

- \_\_\_ 5. The BAR file now appears in the **Application Development** view under the **BARs** folder and under the **Independent Resources > BARfiles > BARs** folder.

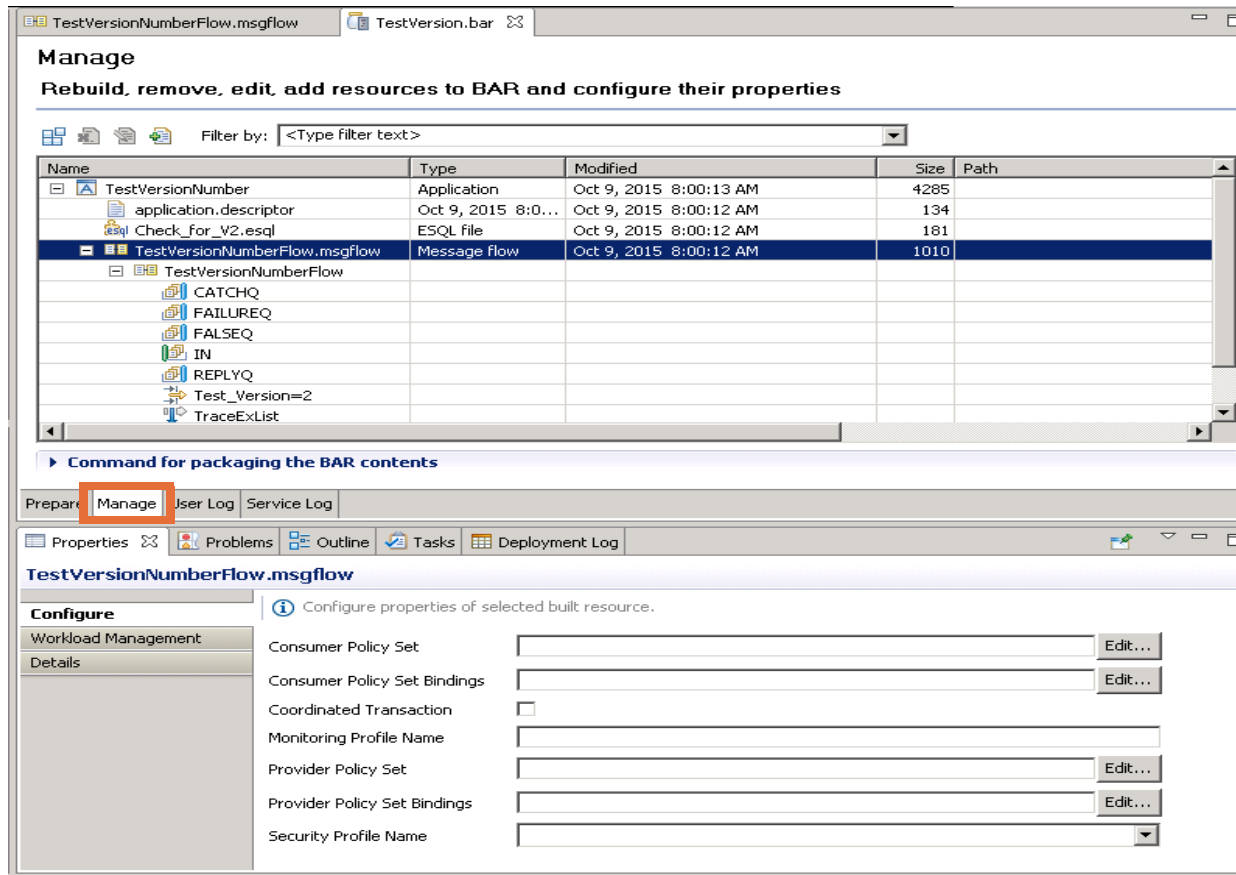


- \_\_\_ 6. In the BAR file editor, click the **Manage** tab.

On the **Manage** tab, you can review and manage the resources that are contained within the BAR file. You can add, remove, or modify resources, and override configurable properties. You can then rebuild the BAR file, in preparation for deployment.

- \_\_\_ 7. Fully expand the contents of the BAR file on the **Manage** tab.

When you expand the contents of the message flow, you see an entry for each node in the message flow. The nodes are listed in alphabetical order.



- \_\_\_ 8. Examine the contents of the **Properties** view below the BAR File editor view as you click the message flow and each node that is contained within the BAR file.

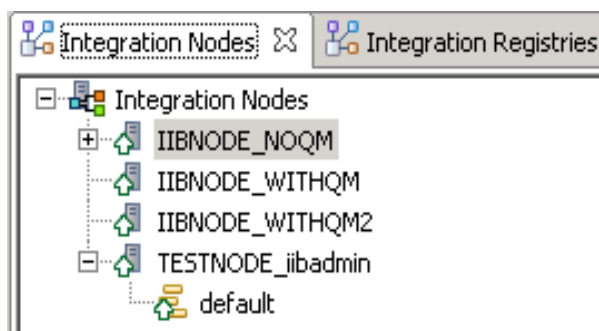
These properties are known as *configurable properties*. These properties can be overridden in the BAR file. In most cases, these properties are a subset of the node properties that you see in the Message Flow editor.

As an administrator, you might want to use the BAR file editor in the Integration Toolkit to modify the configurable properties and update the BAR file. Take some time to examine the configurable properties for the message flow and for each node.

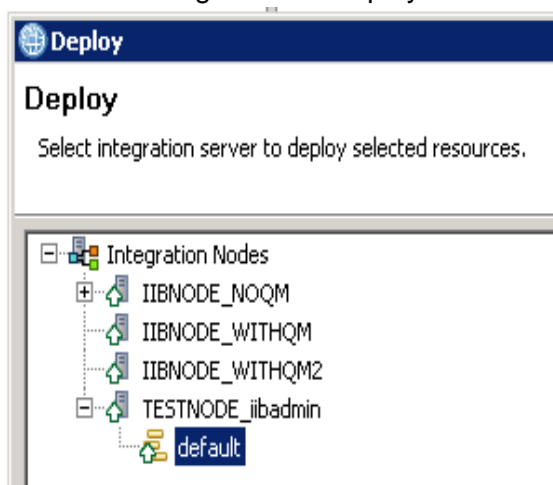
- \_\_\_ 9. Deploy the BAR file to the integration server that is named **default** on the TESTNODE\_iibadmin integration node.

- \_\_\_ a. Verify that the integration node and the integration server are running.

In the Integration Toolkit **Integration Nodes** view, verify that TESTNODE\_iibadmin has a green arrow and that the default integration server also has a green up arrow.



- \_\_\_ b. In the **Application Development** view, right-click the **TestVersion.bar** and then click **Deploy** from the menu. The **Deploy** dialog box is displayed.
- \_\_\_ c. Select the integration server that is named **default** under the TESTNODE\_iibadmin integration node as the target for the deployment.



- \_\_\_ d. Click **Finish**. A **Deploying** dialog box is displayed while the BAR file is deployed to the integration server.

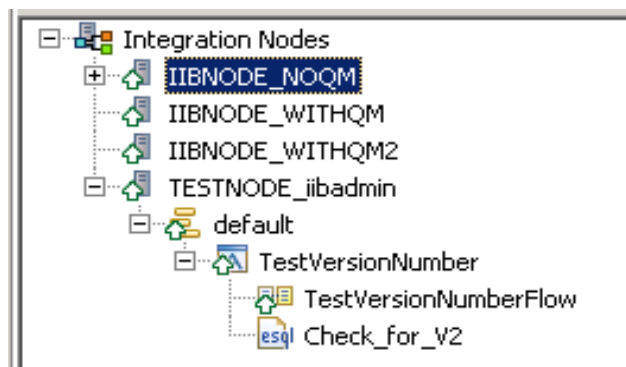


### Note

As you learn later in the course, there are several ways to deploy a message flow to an integration server. Another commonly used method in the Integration Toolkit is to click the BAR file in the **Application Development** navigator and drag it to the integration server in the **Integration Nodes** view.

- \_\_\_ 10. Verify that the operation was successful by using the Integration Toolkit **Integration Nodes** view and the **Deployment log**.
  - \_\_\_ a. Select the **Integration Nodes** view in the lower-left view of the Integration Toolkit (if it is not already selected).

- \_\_\_ b. Expand the **Integration Nodes > TESTNODE\_iibadmin > default** hierarchy until you see the message flow, **TestVersionNumberFlow**.



If the icon next to a message flow is a green upward arrow, the flow is successfully deployed and is running.

If the icon is a yellow triangle, examine the **Problems** view (in the lower-right pane of the IBM Integration Toolkit) for more status information. It might be that the message flow is deployed but not yet running.

If the message flow start is in progress, the icon changes to the green upward arrow when the message flow is started. It is also possible that the flow might require a manual start, in which case you right-click the message flow, and then click **Start**.

You also see another item in the hierarchy (**Chec\_for\_V2**). This component is the ESQL code that the message flow uses in the Filter node. It is included in the BAR file automatically.

- \_\_\_ c. Click the **Deployment Log** tab in the lower-right pane of the Integration Toolkit.
- The Deployment Log shows whether the BAR file was deployed successfully.
- \_\_\_ d. Expand the message in the **Deployment Log** by clicking the plus sign (+).

Details		Timestamp
[TestVersion.bar] has been deployed to integration server default on integration node TESTNODE_iibadmin		Fri Oct 09 08:29...
BIP2871I: The request made by user 'WS2008R2X64\iibadmin' to 'deploy' the resource 'C:/Users/iibadmin/IBM/IIBT10/w		Fri Oct 09 08:29...

You should see a message that indicates that the BAR file was successfully deployed to the integration server. **BIPnnnnns** messages that are preceded with a blue “i” icon (for “information”) indicate success; a yellow triangle icon indicates a warning message, and a red exclamation point icon indicates a deployment failure.

### Part 3: Test the message flow by using the Flow Exerciser

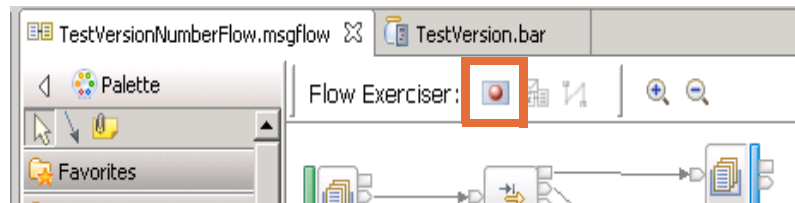
In this part of the exercise, you test the message flow by using the Integration Toolkit Flow exerciser.



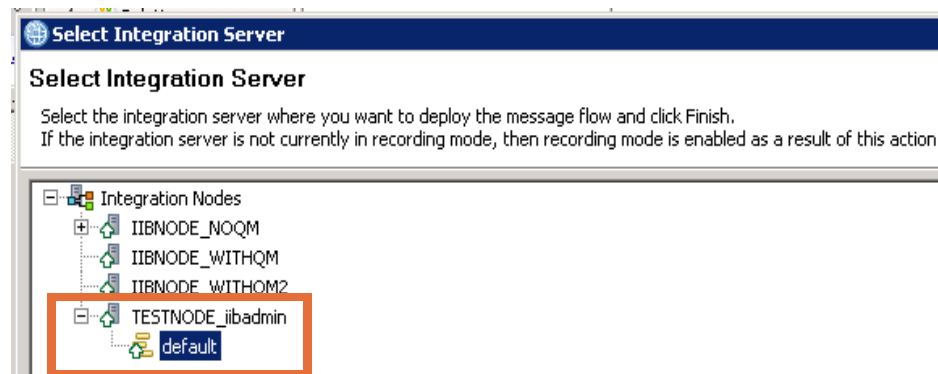
#### Note

The Flow exerciser automatically builds and deploys a BAR file to the selected integration server. When you use the Flow exerciser, you can skip the step to manually build and deploy the BAR file. In this exercise, you manually built and deployed a BAR file in Part 2 so that, as an administrator, you are familiar with the options for building, modifying, and deploying BAR files in the Integration Toolkit.

- \_\_\_ 1. In the Message Flow editor, click the **Start Flow exerciser** icon to deploy the message flow and start the Flow exerciser.



- \_\_\_ 2. Select the **default** integration server on the TESTNODE\_iibadmin integration node and then click **Finish**.



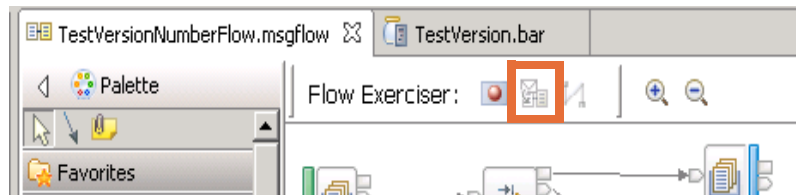
- \_\_\_ 3. The application is deployed to the **default** integration server on TESTNODE\_iibadmin. You should see a window that indicates that the message flow is being deployed.

Click **Close** in the Record Message window.

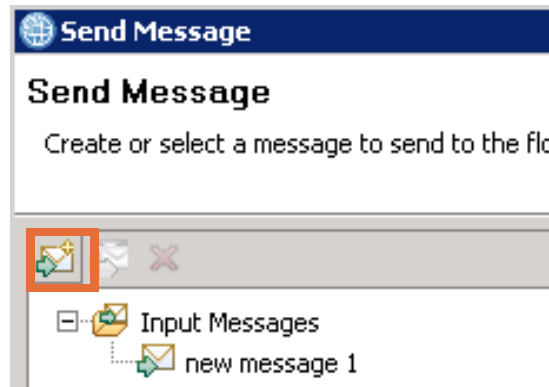
- \_\_\_ 4. The deployed message flow is running. It is waiting for a message to enqueue on the IN queue. When the Flow Exerciser puts a message to the IN queue, the message flow reads (dequeues) the message and sends it through the message flow.

Test the message flow with the Test\_V2\_Msg.xml file by importing the file from the file system and sending the message with the Flow exerciser.

- \_\_\_ a. Click the **Send a message to the flow** icon in the Flow Exerciser toolbar.

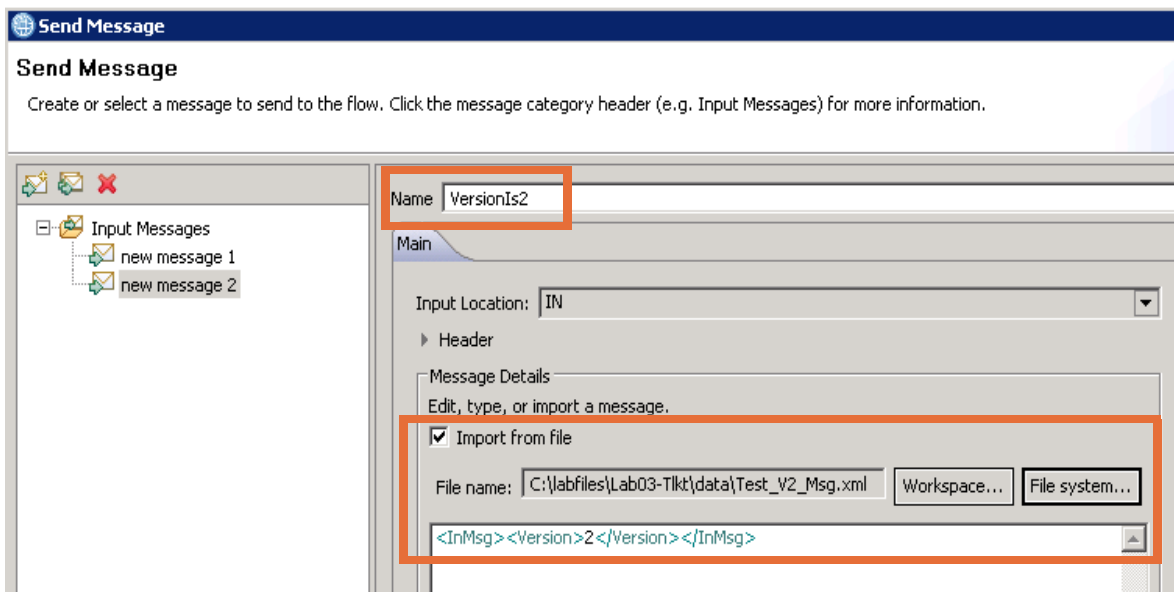


- \_\_\_ b. In the Send Message window, click the **New Message** icon.



- \_\_\_ c. For the **Name**, type: **VersionIs2**
- \_\_\_ d. Click **Import from file** and then click **File system**.
- \_\_\_ e. Browse to `C:\labfiles\Lab03-Tlkt\data`, select the file `Test_V2_msg.xml`, and then click **Open**.

The file is imported into the Flow exerciser.



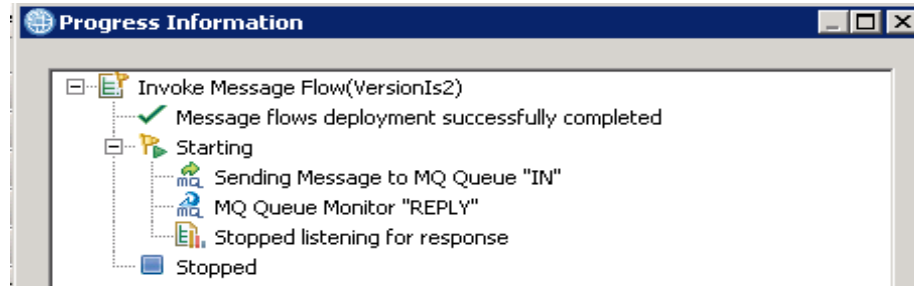
- \_\_\_ f. Examine the data that was read from the file. In this test file, `Version` is set to 2.
- \_\_\_ 5. You are ready to send the test message to the queue for processing by the message flow. Before you do, reexamine the message flow diagram.



Given the test data that you are about to send to the queue, where should the incoming message go? The possible destination queues are REPLY, UNKNOWN, FALSE, CATCH, and FAILURE.

- \_\_\_ 6. Click **Send**.

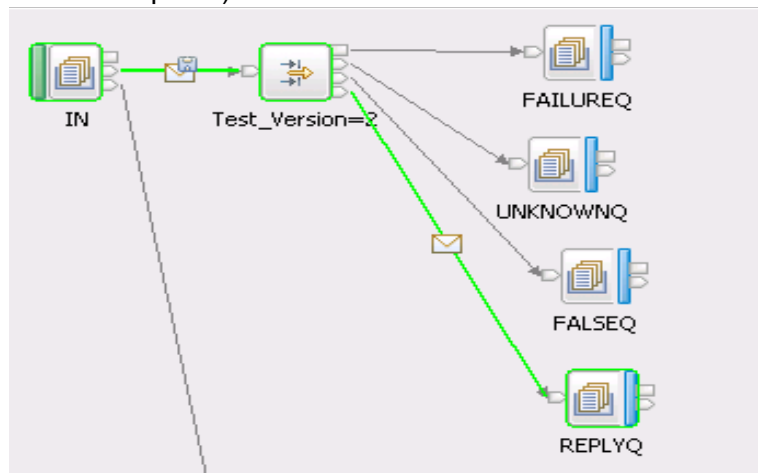
The Progress Information window shows that the message is sent to the input queue IN. It also shows the destination.



- \_\_\_ 7. When the Progress Information window displays **Stopped**, the test is complete. Click **Close**.

- \_\_\_ 8. The message path is highlighted on the message flow. Where did the message go?

The message path should indicate that message is routed to the REPLYQ node (which references the REPLY queue) because the Version = 2.



### Information

If you view the queues by using IBM MQ Explorer, you see that the **Current queue depth** for the output queue is empty. The Integration Toolkit Flow exerciser does not put the message to the output queue; it shows the message path only. If you want to view the messages on the queues, you can use the IBM MQ `amqsput` sample program or the RFHutil SupportPac to test the flow by putting a message to the IN queue.

For example, to use the `amqsput` sample program to verify that the `Test_V2_msg.xml` message is put on the REPLY queue on the IIBQM queue manager, type the following command in an IBM Integration Console window:

```
amqsput IN IIBQM < C:\labfiles\Lab03-Tlkt\data\Test_V2_msg.xml
```

- \_\_\_ 9. Following the same procedure that you followed in Steps 4-8, test the flow by using the other test messages in the `C:\labfiles\Lab03-Tlkt\data\` directory.

You should be able to predict the message destination based on the message content and the filter rules that are described in the exercise **Introduction**.



## Troubleshooting

If the message did not go the expected queue, check the following conditions on the IBM MQ queue manager or the Windows Event Log.

Is the message flow getting messages from the <b>IN</b> queue (is the IBM MQ Open Input Count on the IN queue = 1)?	The message flow might be in a <i>stopped</i> state.  Check the <b>Integration Nodes</b> view in the IBM Integration Toolkit. If necessary, start the message flow.
Is the number of messages on the IBM MQ Backout Requeue Queue = 0?	If Open Input Count for <b>IN</b> = 0, but the deployment was successful, you might have a typographical error in the queue name on the MQInput node. Check the Windows Error Log for one or more error messages.
Is the number of messages on the IBM MQ Backout Requeue Queue > 0 and increasing?	If Open Input Count for <b>IN</b> > 1, there are multiple message flows getting messages from <b>IN</b> . This situation is more likely to happen in later exercises, should you duplicate one flow to serve as a starting point for another.
Was the message sent to the <b>CATCH</b> queue?	Check the Windows Event Log to determine the cause of the runtime error.
Was the message sent to the <b>FALSE</b> or <b>UNKNOWN</b> queue?	Check your test message. Is the <code>&lt;Version&gt;</code> field present? Does <code>&lt;Version&gt; = 2</code> ?

## Exercise Clean up

- \_\_\_ 1. In the Integration Toolkit, complete the following steps:
- \_\_\_ a. Close the message flow in the Message Flow editor.
  - \_\_\_ b. Close the BAR file in the BAR file editor.
  - \_\_\_ c. In the **Integration Nodes** view, right-click the **default** integration server on **TESTNODE\_iibadmin** and then click **Stop recording**.
  - \_\_\_ d. In the **Integration Nodes** view, delete all flows and resources from **default** integration server on **TESTNODE\_iibadmin**.
- Right-click the **default** integration server in the **Integration Nodes** view, and then click **Delete > All Flows and Resources**. Click **OK** in the confirmation window.
- \_\_\_ 2. Close the IBM Integration Toolkit.

## End of Exercise

## Exercise review and wrap-up

In the first part of this exercise, you imported an Integration Bus project interchange file into the Integration Toolkit.

In the second part of this exercise, you created a BAR file and examine the configurable properties by using the Integration Toolkit BAR file editor. You then deployed the BAR file to the development integration server.

In the third part of this exercise, you tested the message flow by using the Integration Toolkit Flow exerciser.

Having completed this exercise, you should be able to:

- Import, deploy, and test a message flow
- Use the BAR file editor to examine the BAR file contents

## Exercise 4. Administering the IBM Integration Bus runtime components

### What this exercise is about

In this exercise, you use the IBM Integration Bus commands and IBM Integration web user interface to manage IBM Integration Bus runtime components and deploy message flow applications. You also back up and restore the integration node configuration data.

### What you should be able to do

After completing this exercise, you should be able to:

- Administer IBM Integration Bus components by using the IBM Integration Bus commands, IBM Integration web user interface, and IBM Integration API Exerciser
- Deploy IBM Integration Bus message flow applications by using the IBM Integration Bus commands, IBM Integration web user interface, and IBM Integration API Exerciser
- Back up and restore an integration node and its configuration

### Introduction

IBM Integration Bus administrators can manage and manipulate the IBM Integration Bus components by the IBM Integration command interfaces and the IBM Integration web user interface. While the IBM Integration web user interface is easier to use and more intuitive, commands give the ability to fully automate the management tasks by using scripts.

In this part of the exercise, you use the IBM Integration command interface to check the status and configuration of the integration node, integration servers, and message flows. You also use the IBM Integration command interface to deploy a message flow.

In the second part of the exercise, you use the IBM Integration web interface to check the status and configuration of the integration node, integration servers, and message flows. You also use the IBM Integration command interface to deploy a message flow.

In the third part of the exercise, you use the IBM Integration API Exerciser sample program to manage the integration node, integration server, and integration node. You also used the Integration API Exerciser sample program to create a script the creates an integration server.

In the fourth part of the exercise, you back up the integration node configuration information.

## Requirements

- IBM Integration Bus V10
- IIBNODE\_NOQM integration node and integration servers that were created in Exercise 1
- Lab exercise files in the C:\labfiles\Lab04-Admin directory

## Exercise instructions



### Note

This exercise uses the IIBNODE\_NOQM integration node and integration servers from Exercise 1, Part 3.

### Part 1: Use the IBM Integration command interface

In this part of the exercise, you use the IBM Integration command interface to check the status and configuration of the integration node, integration servers, and message flows. You also use the command interface to deploy a BAR file that contains an application.



### Important

When you enter commands, do not press Enter where you see a line break in the command. You must enter commands on a single line, although it is permissible for the command to wrap to subsequent lines as you type it. Press Enter only after you type the entire command.

- \_\_\_ 1. IBM Integration Bus supplies a special command console from which you run Integration Bus commands. To view what happens if you try to work outside of the Integration Console, open a Windows command prompt window.

- \_\_\_ a. From Windows, click **Start**
- \_\_\_ b. Enter `cmd` in the **Open** field and then click **OK**.
- \_\_\_ c. Enter the command: `mqsilist IIBNODE_NOQM`

You know that the integration node is deployed and running, so what might cause this problem?

In the lecture material, you learned that you must run `mqsiprofile.cmd` to set up the environment for various components. Do *not* run the `mqsiprofile` command now.

- \_\_\_ d. Close the Windows command prompt window.
- \_\_\_ 2. Review the `mqsiprofile.cmd` file in the `C:\<Integration Bus Install Path>\server\bin` directory.

In the lab exercise image for this course, the *Integration Bus Install Path* is `C:\Program Files\IBM\IIB\10.0.0.0\server`.

- \_\_\_ a. Use Windows Explorer to locate the `mqsiprofile.cmd` file.
- \_\_\_ b. Right-click the file and then click **Edit** to open the file in Notepad.

You can see that the command file sets some environment variables. When you use the IBM Integration Console in the Windows environment, `mqsiprofile.cmd` is automatically called to set the command environment.

- \_\_\_ c. Close the file when you are done looking at it. Do not save any changes if you are prompted to do so.
- \_\_\_ 3. If it is not already running, start the IBM Integration Console as an administrator.  
Right-click the shortcut icon on the desktop, or click **Start > All Programs > IBM Integration Bus 10.0.0.0**, right-click **IBM Integration Console**, and then click **Run as administrator**.
- \_\_\_ 4. Get a list of running integration servers and message flows on IIBNODE\_NOQM by entering the command:

```
mqsilist IIBNODE_NOQM -d2
```



### Note

For most commands, you can enter the command followed by a question mark character (?) to get a summary of the command syntax.

You should see that two integration servers are running on IIBNODE\_NOQM: **server1** and **server2**.

You should also see that no message flows are currently running on the integration servers.

- \_\_\_ 5. Deploy the AdminApp.bar BAR file to the **server2** integration server.  
The AdminApp.bar file contains an application that is named Transformation\_Map. The application contains a message flow that is also named Transformation\_Map.  
When deploying a BAR file, the **mqsideploy** command looks for the file in the current directory (the one from which the command is run) by default. If the BAR file is not in that directory, you must specify the full path.
- \_\_\_ a. In the IBM Integration Console, change directories to the directory that contains the BAR file (C:\labfiles\Lab04-Admin\resources) before you deploy the file. Type:

```
cd C:\labfiles\Lab04-Admin\resources
```

- \_\_\_ b. Type the following command to verify that the file AdminApp.bar is in the C:\labfiles\Lab04-Admin\resources directory.

```
dir
```

- \_\_\_ c. Type the following command to view the contents of the BAR file:

```
mqsireadbar -b AdminApp.bar
```

The command should return the following response.

```
BIP1052I: Reading Bar file using runtime mqsireadbar...
AdminApp.bar: Transformation_Map.appzip (10/12/15 2:51 PM):
```

- \_\_\_ d. Type the command to deploy the AdminApp.bar file:

```
mqsideploy IIBNODE_NOQM -e server2 -a AdminApp.bar
```



The command should return the following response.

```
BIP1039I: Deploying BAR file
'C:\labfiles\Lab04-Admin\resources\AdminApp.bar' to integration node
'IIBNODE_NOQM' <integration server 'server2'>
BIP1092I: The integration node successfully processed the deployment
request.
```

- \_\_\_ 6. Verify that the flow was successfully deployed. Type:

```
mqsilist IIBNODE_NOQM -e server2
```

You should see a response that is similar to the following example:

```
BIP1275I: Application 'Transformation_Map' on integration server 'server2'
is running.
```

- \_\_\_ 7. Remove the deployed message flow application from the **server2** integration server:

```
mqsideploy IIBNODE_NOQM -e server2 -d Transformation_Map
```



### Hint

To reduce the amount of typing you do, you can press the up arrow key on your keyboard in the IBM Integration Console window to retrieve a previous command. You can then edit the portion of command that changed (if necessary), by using the left arrow, right arrow, Delete, and Backspace keys. Press Enter to run the command after you finish editing it.

For example, the command you enter in the next step is identical to the command you entered in step 7. Press the up arrow key until the `mqsideploy` command is displayed, edit the command, and then press Enter to reissue the command.

- \_\_\_ 8. Redeploy the `AdminApp.bar` BAR file to the **server2** integration server on IIBNODE\_QM:

```
mqsideploy IIBNODE_NOQM -e server2 -a AdminApp.bar
```

- \_\_\_ 9. Enter a command that stops the message flow application that you deployed in Step 8:

```
mqsisstopmsgflow IIBNODE_NOQM -e server2 -k Transformation_Map
```

- \_\_\_ 10. Restart the message flow:

```
mqsisstartmsgflow IIBNODE_NOQM -e server2 -k Transformation_Map
```

You should see a confirmation message that the application started.

- \_\_\_ 11. Minimize or close the IBM Integration Console window. You use it later in the exercise.

## Part 2: Use the IBM Integration web interface

In this part of the exercise, you use the IBM Integration web interface to complete basic administration on the integration node, integration server, and message flow applications.

- \_\_\_ 1. Start the IBM Integration web interface for IIBNODE\_NOQM.

If you do not already know the HTTPConnector port for this integration node, type the following the command to find the port.

```
mqsireportproperties IIBNODE_NOQM -b webadmin -o HTTPConnector -a
```

- \_\_\_ 2. Display the IIBNODE\_NOQM **QuickView** and **Advanced Properties**.
  - \_\_\_ a. Select IIBNODE\_NOQM in the IBM Integration web interface navigator in the left pane.
  - \_\_\_ b. Show the integration node **Advanced Properties** by clicking the down arrow on the **Advanced Properties** heading in the right pane.

The screenshot shows the IBM Integration web interface. The left pane contains a navigation tree with the following items: Servers, Operational Policy, Data, Security, and Monitoring. The 'IIBNODE\_NOQM' item is selected and highlighted with a red box. The main pane displays the 'IIBNODE\_NOQM - Integration Node' page. It has tabs for 'Overview' and 'Statistics'. Below the tabs is a 'Quick View' section with a table of properties:

Node Name	IIBNODE_NOQM
Version	10000
Admin Security	Off
Run Mode	running
Short Description	
Long Description	

Below the Quick View section is an 'Advanced Properties' section, which is highlighted with a red box. It contains a table of properties:

Platform Name	Windows Server 2008 R2 Standard
Fixpack Capability	unrestricted
Operation Mode	advanced
Platform Architecture	AMD64
Platform Version	6.1 build 7601 Service Pack 1
Queue Manager	
Build Level	S000-L150316.10572
Admin Agent Process ID	3004

At the bottom of the main pane is a 'Component Properties' section.

- \_\_\_ 3. Display the properties for the **server2** integration server.
  - \_\_\_ a. Expand **Servers** by clicking the down pointing arrow on the left side of **Servers**.
  - \_\_\_ b. Click **server2**.

- \_\_\_ c. Show the integration server **Advanced Properties** by clicking the down arrow on the **Advanced Properties** heading in the right pane.

The screenshot displays the IBM Integration console interface. On the left, a navigation tree shows the hierarchy: IIBNODE\_NOQM > Servers > server1 > **server2** (highlighted with a red box). The main area is titled 'server2 - Integration Server' and has tabs for 'Overview', 'Resource Statistics', and 'Statistics'. The 'Overview' tab is active, showing a 'Quick View' table with the following data:

Integration Server Name	server2
Run Mode	running
UUID	7f6a21da-2ba1-4d52-9200-b2d74b857816
Running	true
Short Description	
Long Description	
Record Mode	Disabled

Below the 'Quick View' table, the 'Advanced Properties' section is expanded (highlighted with a red box), showing a table of configuration settings:

Injection Mode	Disabled
Process ID	2948
Trace Level	none
Soap Nodes use Embedded Listener	true
Thread Local Proxy Name Managers	false
Console Mode	off
HTTP Nodes use Embedded Listener	false
Inactive User Exit List	
Active User Exit List	
Trace Node Level	on
User Trace Level	none

At the bottom of the main pane, there is a link for 'Resource Manager Properties'.

- \_\_\_ 4. Display the properties of a message flow that is deployed to the **server2** integration server.
- \_\_\_ a. Click the arrow on the left side of **server2** to show its contents.
  - \_\_\_ b. Click the arrow on the left side of **Applications** to show its contents. You should see that the application **Transformation\_Map** is deployed to **server2**.
  - \_\_\_ c. Click the arrow on the left side of **Transformation\_Map** to show its contents.
  - \_\_\_ d. Click the arrow on the left side of **Message Flows** to show its contents.
  - \_\_\_ e. Click the **Transformation\_Map** message to display its properties and then expand the **Advanced Properties** and **Deployed Properties** sections.

The screenshot displays the IBM Integration Bus V10 Administration console. On the left, a tree view shows the hierarchy: IIBNODE\_NOQM > Servers > server2 > Applications > Transformation\_Map. The 'Transformation\_Map' message flow is selected and highlighted with a red box. The main panel on the right shows the 'Transformation\_Map - Message Flow' details. It includes tabs for Overview, Statistics, and Operational Policy. The 'Quick View' section shows the following properties:

Message Flow Name	Transformation_Map
Version	
UUID	3a88e189-9b67-480f-b095-4c00ec6ca5bb
Service Trace Level	none
Commit Count	1
Additional Instances	0
Start Mode	Maintained
Coordinated Transaction	no
Commit Interval	0
Running	true
Run Mode	running

The 'Advanced Properties' section shows:

User Trace Level	none
Active User Exit List	
Inactive User Exit List	

The 'Deployed Properties' section shows:

Modified time	2015-10-12 15:18:28.000 -0700
Deployed time	2015-10-12 12:26:51.702 -0700
Bar file name	C:\labfiles\Lab04-Admin\resources\AdminApp.bar

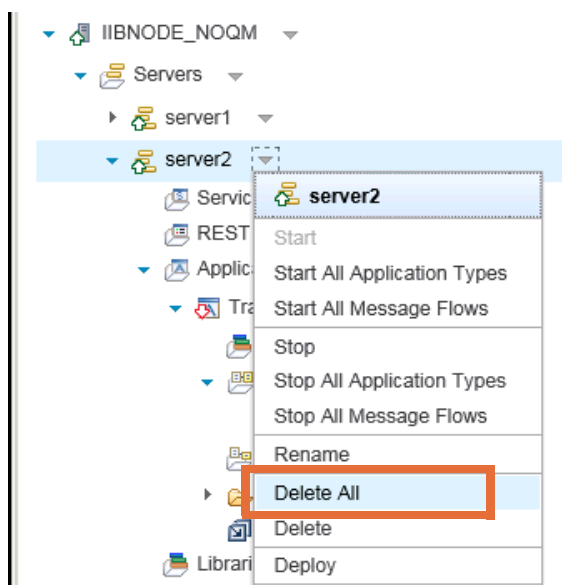
\_\_\_ 5. Stop the message flow that is running on **server2**.

\_\_\_ a. Click the down arrow next to the **Transformation\_Map** message flow and then click **Stop**.

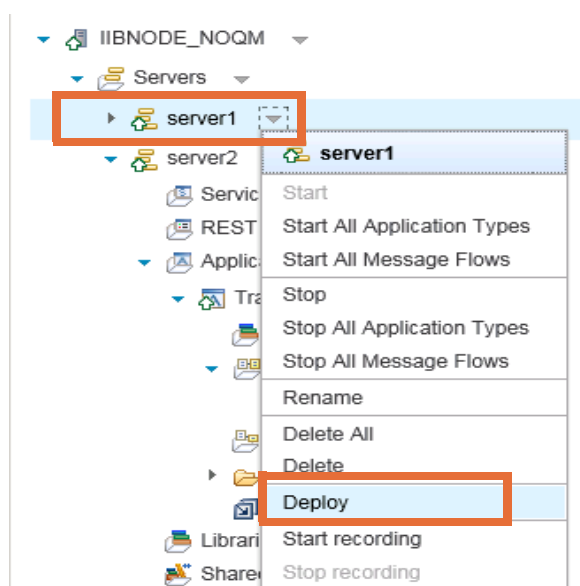
The screenshot shows the 'Transformation\_Map' message flow selected in the tree view. A context menu is open, showing the following options: Start, Stop, and Force stop. The 'Stop' option is highlighted with a red box.

\_\_\_ b. Verify that the **runMode** property for the message flow is now set to *stopped*.

- \_\_\_ 6. Delete all deployed applications and resources on **server2**.
- \_\_\_ a. Click the down arrow on the right side of **server2** to display the menu, and then click **Delete All**.

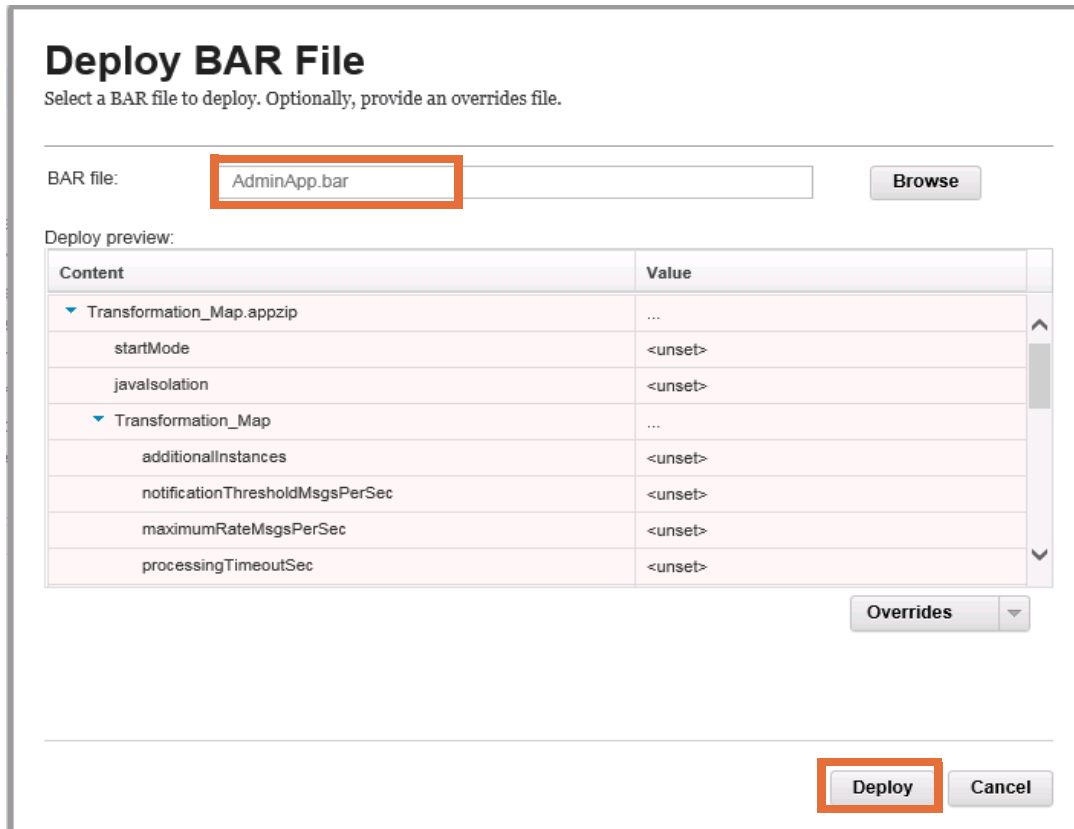


- \_\_\_ b. A confirmation window asks if you are sure that you want to delete all deployed contents. Click **Yes** on the confirmation window.
- \_\_\_ c. Verify that no applications are deployed to **server2**.
- \_\_\_ 7. Deploy the AdminApp.bar file to **server1**.
- \_\_\_ a. Click down-arrow on right of **server1** and then click **Deploy**.



- \_\_\_ b. Click **Browse** and browse to C:\labfiles\Lab04-Admin\resources, click AdminApp.bar, and then click **Open**.

- \_\_\_ c. Expand the contents of the BAR file to view the configurable properties of the message flow and the message flow node.



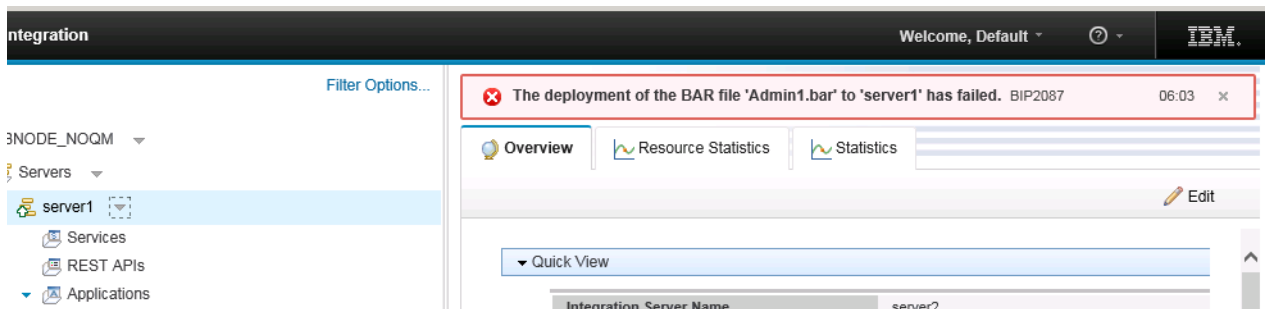
**Deploy BAR File**  
Select a BAR file to deploy. Optionally, provide an overrides file.

BAR file:

Deploy preview:

Content	Value
Transformation_Map.appzip	...
startMode	<unset>
javasolution	<unset>
Transformation_Map	...
additionalInstances	<unset>
notificationThresholdMsgsPerSec	<unset>
maximumRateMsgsPerSec	<unset>
processingTimeoutSec	<unset>

- \_\_\_ d. Click **Deploy**.
- \_\_\_ e. Verify that the **Transformation\_Map** message flow application is deployed to **server1**.
- \_\_\_ 8. The IIBNODE\_NOQM integration node that you are using in this exercise does not have a queue manager that is associated with it. In this step, you see what happens when you try to deploy a message flow that requires a default queue manager.
- \_\_\_ a. In the IBM Integration web interface for IIBNODE\_NOQM, click down-arrow on right of **server1** and then click **Deploy**.
- \_\_\_ b. Click **Browse** and browse to C:\labfiles\Lab04-Admin\resources, click Admin1.bar, and then click **Open**.
- \_\_\_ c. Click **Deploy**. After a brief pause, an error message displays in the web interface that indicates that the deployment failed.



The screenshot shows the IBM Integration web interface. On the left, a tree view shows the hierarchy: IIBNODE\_NOQM > Servers > server1. The main area displays a red error message: "The deployment of the BAR file 'Admin1.bar' to 'server1' has failed. BIP2087". Below the error message, there are tabs for Overview, Resource Statistics, and Statistics. A Quick View section at the bottom shows a table with one row: Integration Server Name | server2.

- \_\_\_ 9. The error detail is provided in the **Admin Log** in the IBM Integration web interface.
- \_\_\_ a. Click the arrow on the left side of the **Monitoring** folder to show its contents.
  - \_\_\_ b. Click **Admin Log**.
  - \_\_\_ c. Review the messages in the Admin Log.

One of the error messages in the Admin Log describes the reason why the deployment failed. In this case, deployment failed because the message flow in the BAR file requires a default queue manager on the integration node.

The screenshot shows the IBM Integration web interface. On the left, the 'Monitoring' folder is expanded, and 'Admin Log' is selected. The main area displays the 'Admin Log' with a table of messages. The message BIP2683E is highlighted with a red box.

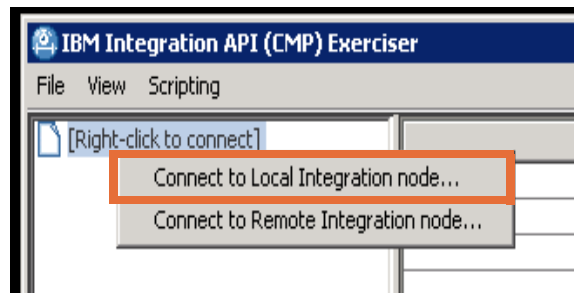
Message	Source	Timestamp	Message Detail
BIP4041E	Runtime Response	2015-10-13 06:03:45.237 Pacific Daylight Time	Integration server 'server1' received an administration request that encountered an exception. While attempting to process an administration request, an exception was encountered. No updates have been made to the configuration of the integration server. Review related error messages to determine why the administration request failed.
BIP2683E	Runtime Response	2015-10-13 06:03:45.237 Pacific Daylight Time	A component was configured to use the queue manager that is specified on the integration node, but no queue manager was found for the integration node. By default, components in IBM Integration Bus use the queue manager that is specified on the integration node to connect to WebSphere MQ. No queue manager is specified on the integration node, so the component cannot resolve the connection to a queue manager. For MQInput, MQOutput, and MQGet nodes, it is possible to specify a queue manager on the message flow node, but for Aggregation, Collector, Sequence, and Resequencing nodes, the queue manager must be specified on the integration node. Specify a queue manager on the integration node, or use a queue manager that is specified on a MQInput, MQOutput, or MQGet node.
BIP2871I	Administration Result	2015-10-13 06:03:45.237 Pacific Daylight Time	The request made by user 'SYSTEM[Default]' to 'deploy' the resource 'Admin1.bar' of type 'BAR' on parent 'server1' of type 'ExecutionGroup' has the status of 'FAILED'.
BIP2087E	Runtime Response	2015-10-13 06:03:45.236 Pacific Daylight Time	Integration node 'IIBNODE_NOQM' was unable to process the internal configuration message. The entire internal configuration message failed to be processed successfully. Use the messages following this message to determine the reasons for the failure.

- \_\_\_ 10. Minimize the IBM Integration web interface browser window.

### Part 3: Use the IBM Integration API Exerciser

In this part of the exercise, you use the IBM Integration API Exerciser to manage an integration node and its resources. You also use the Integration API Exerciser to create an administration script.

- \_\_\_ 1. Start the IBM Integration API (CMP) Exerciser.
  - \_\_\_ a. In Windows Explorer, browse to the `C:\Program Files\IBM\IIB\10.0.0.0\server\sample\IntegrationAPI\` directory.
  - \_\_\_ b. Double-click the `StartIntegrationAPIExerciser.bat` file to start the IBM Integration API Exerciser.
- \_\_\_ 2. To connect to the local integration node, right-click **Right-click to connect** and then click **Connect to Local Integration node**.



- \_\_\_ 3. Select the `IIBNODE_NOQM` integration node, and then click **Submit**.



4. After a connection is established, the status information is displayed in the lower part of the Integration API Exerciser window. The upper part of the window displays information about the integration node.

The screenshot shows the IBM Integration API (CMP) Exerciser window. The left pane displays a tree view of the integration node 'IIBNODE\_NOQM', which includes 'server1' and 'server2'. 'server1' contains several transformation maps and schemas. The right pane displays a table of BrokerProxy methods and their results.

BrokerProxy Method	Result
getAdministrationAPIVersion()	100
getAdvancedProperties()	
platformName	Windows Server 2008 R2 Standard
FixpackCapability	unrestricted
platformVersion	6.1 build 7601 Service Pack 1
platformArchitecture	AMD64
operationMode	advanced
AdminAgentPID	3004
buildLevel	S000-L150316.10572
queueManager	
getBasicProperties()	
AdminSecurity	inactive
version	10000
name	IIBNODE_NOQM
runMode	running
shortDesc	
longDesc	
getBrokerLongVersion()	S000-L150316.10572

The bottom pane displays a log of events:

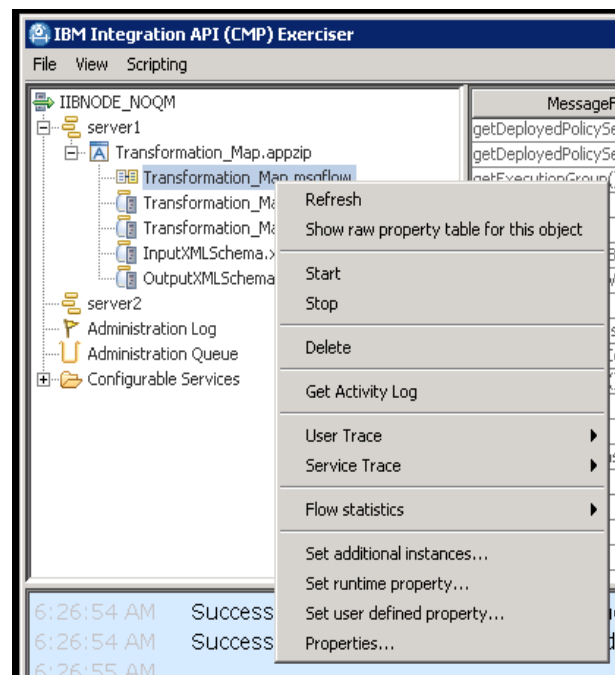
```

6:26:54 AM Successfully registered for updates to the log <Log>.
6:26:54 AM Successfully registered for updates to administration queue <Administration Queue>.
6:26:55 AM 
6:26:55 AM Successfully connected. Click on an object to select it and display its properties.
6:26:55 AM Right-click a selected object to manipulate it.
6:26:55 AM The IBM Integration API (CMP) Exerciser is set to wait for requests to be fully completed by
the integration node before returning; expect pauses while the integration node processes each request.
6:26:55 AM You can change this setting using File -> Set Timeout characteristics.
6:26:55 AM <---- iapi.exerciser.ClassTesterForBrokerProxy.testConnectToLocalBroker
6:26:55 AM

```

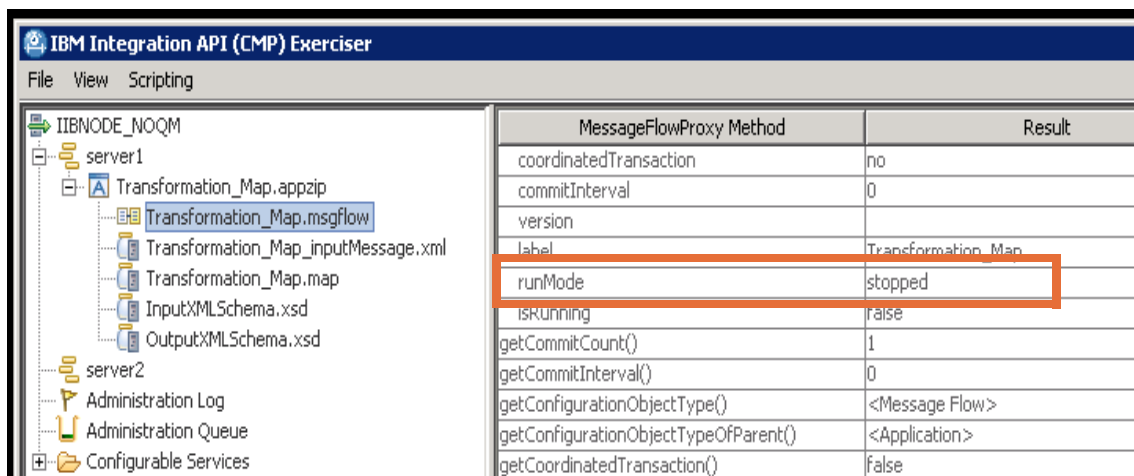
Connected to integration node 'IIBNODE\_NOQM'.

- \_\_\_ 5. Select the message flow **Transformation\_Map.msgflow** that you deployed to the integration server **server1** to display the message flow properties.
- \_\_\_ 6. By right-clicking the message flow, you can turn on tracing, start and stop the message flow, delete the message flow, and complete other operations.



Stop the flow by clicking **Stop** on the menu.

- \_\_\_ 7. Verify that the flow stopped by verifying that the **runMode** property is now *stopped* for the **Transformation\_Map.msgflow**.



You can also use the IBM Integration web interface for IIBNODE\_NOQM to verify the status of the message flow.

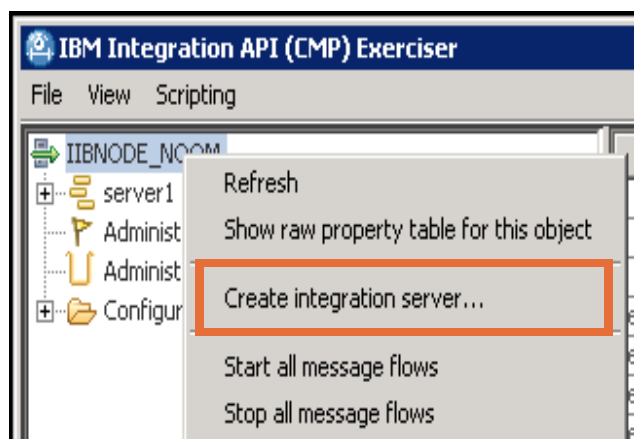
- \_\_\_ 8. Delete the **server2** integration server.

Right-click the **server2** integration server on IIBNODE\_NOQM and then click **Delete**. The delete is completed without confirmation.

- \_\_\_ 9. With the IBM Integration API Exerciser, you can record a script of an operation, which captures it for reuse. You can then run this script against other integration node environments.

Create a script that records the steps for creating an integration server and save it in the C:\labfiles directory.

- \_\_\_ a. From the IBM Integration API Exerciser menu bar, click **Scripting > Record New Script**.
- \_\_\_ b. Browse to the C:\labfiles directory and specify a descriptive name, such as **CreateNewIS.xml**.
- \_\_\_ c. Click **Save**. Scripting is now active.
- \_\_\_ d. Right-click the IIBNODE\_NOQM integration node and then click **Create integration server**.

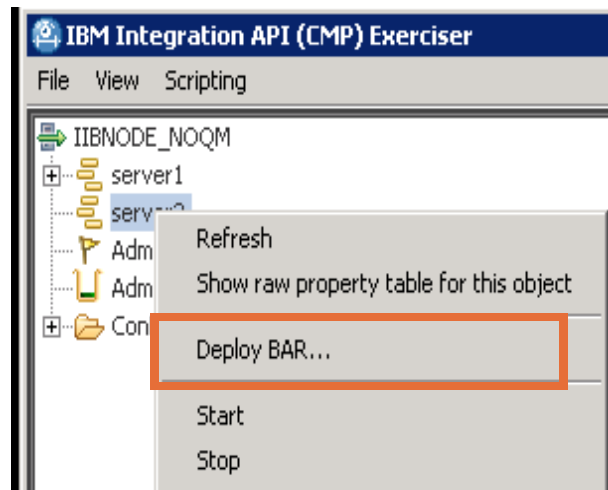


- \_\_\_ e. Name the new integration server **server2**, and then click **Submit**.
  - \_\_\_ f. After you see the new integration server show in the list of integration servers, stop the script recording by clicking **Scripting > Stop Recording**.
- \_\_\_ 10. From the IBM Integration API Exerciser, delete the **server2** integration server.
- \_\_\_ 11. Re-create the **server2** integration server from the script.
- \_\_\_ a. Click **Scripting > Play Back Recorded Script**.
  - \_\_\_ b. Browse to the C:\labfiles directory.
  - \_\_\_ c. Click the XML file that you created in Step 9 and then click **Open**.

The script runs and creates the **server2** integration server again. The integration server is displayed in the list after it is created.

\_\_\_ 12. Verify the **server2** integration server by deploying an Integration Bus application.

- \_\_\_ a. Right-click the **server2** integration server and click **Deploy BAR**.



- \_\_\_ a. Browse to the `C:\labfiles\Lab04-Admin\resources` directory and then select the `AdminApp.bar` file. Click **Open** and wait for the deployment to complete.

- \_\_\_ b. Verify that the message flow was deployed and running.

\_\_\_ 13. Close the IBM Integration API Exerciser.

#### **Part 4: Back up and restore the integration node**

In this part of the exercise, you back up the integration node configuration information.

Normally, you want to back up the integration node configuration information to a separate drive. For this exercise, you create a local folder and back up to that location.

- \_\_\_ 1. Use Windows Explorer to create a directory that is named **IIBNODE\_Backup** in the `C:\labfiles` directory.
- \_\_\_ 2. Open an IBM Integration Console and back up the integration node configuration to the new folder. You can use the `-a` option to name the archive. You do not use it here so that you can see the default file name for the archive.

Type the following command to back up IIBNODE\_NOQM:

```
mqsibbackupbroker IIBNODE_NOQM -d C:\labfiles\IIBNODE_Backup
```

The command should return a message similar to the following example:

```
BIP125I: Creating backup file
'C:\labfiles\IIBNODE_Backup\IIBNODE_NOQM_ccmdd_hhmmss.zip' for integration
node 'IIBNODE_NOQM'.
BIP8071I: Successful command completion.
```

In the response message, `ccmdd` is the year, month, and day, and `hhmmss` is the hour, minute, and second of the backup file creation time.

- \_\_\_ 3. With Windows Explorer, browse to the `C:\labfiles\IIBNODE_Backup` directory and verify that the backup file is in the directory.

You now have all the necessary information to restore your integration node. This file can be copied and stored elsewhere (on some type of external media) or, at a minimum, on a separate drive from where the domain exists.



### Important

The integration node backup command does *not* back up all the resources that message flows require to function correctly. These resources include:

- IBM MQ queues
- Data that is stored in user databases
- Transient information for inflight aggregations or collections
- Runtime code, including resources that are associated with user-defined extensions such as nodes, parsers, and exits

- \_\_\_ 4. Simulate a system failure by deleting the integration node and integration node components. After you complete the system failure simulation, you restore the integration node and components by using the backup file that you created in Step 2.
  - \_\_\_ a. From the IBM Integration web interface for IIBNODE\_NOQM, stop any message flows that are running on **server1** and **server2**.
  - \_\_\_ b. From the IBM Integration web interface, delete the integration servers.
  - \_\_\_ c. From the IBM Integration Console, stop and then delete the integration node IIBNODE\_NOQM and all files in its workpath.

Type:

```
mqsistop IIBNODE_NOQM
mqsdeletebroker IIBNODE_NOQM -w
```



### Note

You can also stop the integration node services by using the Windows Services control panel.

Click **All Program > Administrative Tools > Services**.

Scroll to the IBM Integration Bus component for the integration node and then click **Stop**.

- \_\_\_ 5. Attempt to start the integration node from a command console:

```
mqsistart IIBNODE_NOQM
```

You should receive a message that indicates that the integration node does not exist.

- \_\_\_ 6. Re-create the integration node IIBNODE\_NOQM. Type:

```
mqsicreatebroker IIBNODE_NOQM
```

**Important**

Do not start the **IIBNODE\_NOQM** integration node after you create it. The integration node must be stopped before you can restore it.

- \_\_\_ 7. Using the IBM Integration Console, restore the integration node. Type:

```
mqsirestorebroker IIBNODE_NOQM -d C:\labfiles\IIBNODE_Backup -a archive_file
```

In the command, replace *archive\_file* with the name of the archive file that the **mqsibbackupbroker** command creates in Step 2.

After the restore completes, you receive a message that indicates that the integration node was restored.

- \_\_\_ 8. Start the integration node by using the IBM Integration Console.

- \_\_\_ 9. Get the port number for the IBM Integration web interface for IIBNODE\_NOQM. Type:

```
mqsireportproperties IIBNODE_NOQM -b webadmin -o HTTPConnector -a
```

- \_\_\_ 10. Using the IBM Integration web interface, verify that the integration node is restored correctly. It might be necessary to enter the URL for the IIBNODE\_NOQM integration node again.

You should have two integration servers that are named **server1** and **server2**.

The integration server **server2**, should be running the **Transformation\_Map** application.

## Exercise clean up

- \_\_\_ 1. Stop any message flow applications that are running on **server1** and **server2** on IIBNODE\_NOQM.
- \_\_\_ 2. Delete any message flow applications from **server1** and **server2** on IIBNODE\_NOQM.

## End of exercise

## Exercise review and wrap-up

In this part of the exercise, you used the IBM Integration command interface to check the status and configuration of the integration node, integration servers, and message flows. You also used the IBM Integration command interface to deploy a message flow.

In the second part of the exercise, you used the IBM Integration web interface to check the status and configuration of the integration node, integration servers, and message flows. You also used the IBM Integration command interface to deploy a message flow.

In the third part of the exercise, you used the IBM Integration API Exerciser sample program to manage the integration node, integration server, and integration node.

In the fourth part of the exercise, you backed up the integration node configuration information.

Having complete this exercise, you should be able to:

- Administer IBM Integration Bus components by using the IBM Integration Bus commands, IBM Integration web user interface, and the IBM Integration API Exerciser
- Deploy IBM Integration Bus message flow application by using the IBM Integration Bus commands, IBM Integration web user interface, and the IBM Integration API Exerciser
- Back up and restore an integration node and its configuration





## Exercise 5. Using file-based security to control administration access

### What this exercise is about

In this exercise, you use roles and file-based security to control administration access to the IBM Integration web user interface, integration nodes, integration servers, and message flows.

### What you should be able to do

After completing this exercise, you should be able to:

- Activate administration authority
- Use IBM Integration Bus file-based security to assign permissions for integration nodes, integration servers, and message flows

### Introduction

Administration security controls the rights of users to complete administrative tasks for an integration node and its resources.

IBM Integration Bus V10 can control access to Integration node resources through the IBM Integration web user interface. Different web users can have different access rights across these functions, and access can be granted, denied, or revoked.

The access authorities are defined against a set of user definitions that represent the available security roles. A role is a set of security permissions that control access to an integration node and its resources, and each web user account is associated with a particular role. The permissions are checked to determine a web user's authorization to complete tasks in the web user interface or the REST application programming interface (API). Each web user is then defined to use one or more of these security roles.

In this lab, you use an integration node that does not have a queue manager that is associated with it (IIBNODE\_NOQM) and create three user roles.

**iibRole1** For integration node administration access, read-only

**iibRole2** For integration node administration access, read, write

**iibRole3** For integration administration access, all functions

In the second part of the exercise, you create web users and assign them to one of the defined roles.

Finally, you explore the IBM Integration web user interface as each user.

## Requirements

- IBM Integration Bus V10
- IIBNODE\_NOQM integration node and integration servers that were created in Exercise 1
- Lab files in the C:\labfiles\Lab05-Sec directory

## Exercise instructions

### Exercise set up

- \_\_\_ 1. Check the status of file-based administration on the integration node IIBNODE\_NOQM.

In an IBM Integration Console window, type:

```
mqsireportbroker IIBNODE_NOQM
```

Ensure that you see the line: Administration security = 'inactive'

If security is active on the integration node, deactivate it by typing the following commands:

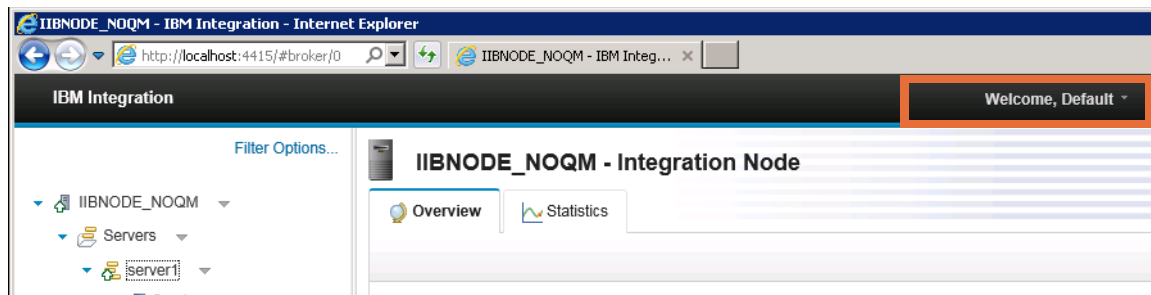
```
mqsistop IIBNODE_NOQM
```

```
mqsichangebroker IIBNODE_NOQM -s inactive
```

```
mqsistart IIBNODE_NOQM
```

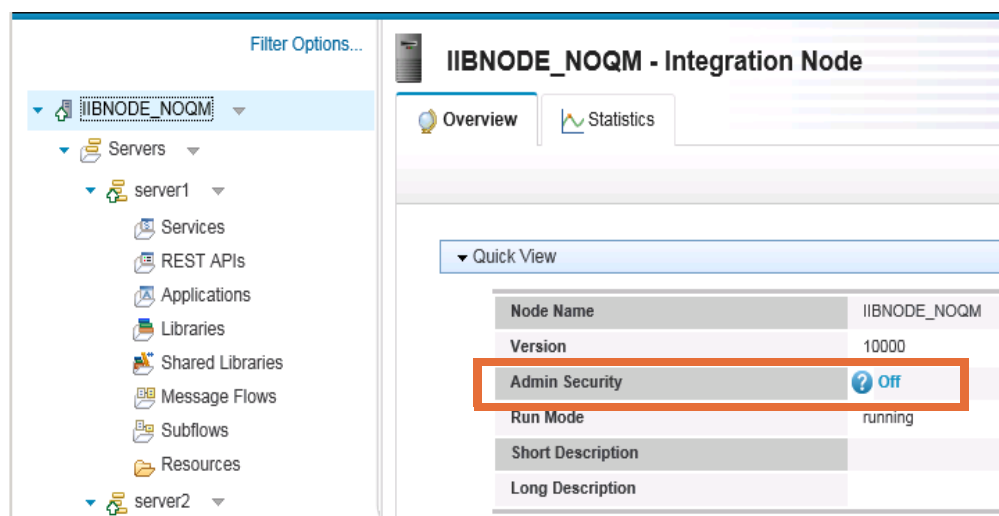
- \_\_\_ 2. If not already open, open a web browser and start the IBM Integration web interface for IIBNODE\_NOQM.

You should see that you are currently logged in as a user this is named **Default**.



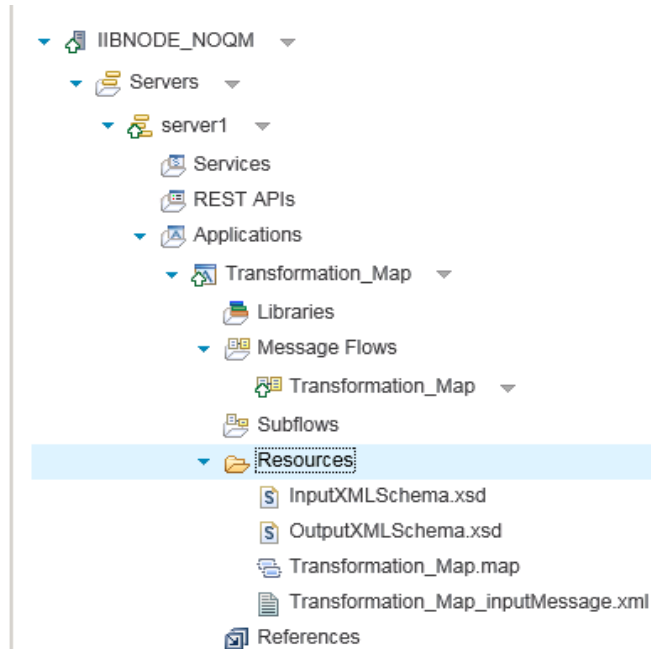
The **Default** user has full update access to all deployed integration node resources.

- \_\_\_ 3. Click **IIBNODE\_NOQM** to display its properties. In the **Quick View** properties, you should see that **Admin Security** is Off.



- \_\_\_ 4. Using the Integration web interface, deploy the file  
C:\labfiles\Lab05-Sec\resources\AdminApp.bar to the **server1** integration server.

After a brief pause, you should see that the **Transformation\_Map** application deployed successfully.



If the integration node web administration HTTP listener is enabled, and security is not active for the integration node, then any user can access the Integration web user interface and control the integration node resources.

## Part 1: Activate file-based security on the integration node

In this part of the exercise, you activate security for the IIBNODE\_NOQM integration node.

- \_\_\_ 1. If an IBM Integration Console is not already open, start an Integration Console session with Windows “Administrator” privileges.
  - \_\_\_ a. Click **Start > All Programs > IBM Integration Bus 10.0.0.0**, right-click **IBM Integration Console 10.0.0.0**, and then click **Run as administrator**.

Optionally, you can right-click the **IBM Integration Console** icon on the desktop and then click **Run as administrator**.

- \_\_\_ b. Click **Yes** to run the IBM Integration Console with “Administrator” privileges.
- \_\_\_ 2. Security can be activated only when the integration node is shut down. Type the following command to stop the integration node:

```
mqsistop IIBNODE_NOQM
```

When you enter the `mqsistop` command, you should see that the IBM Integration web interface page is unavailable with the message *Real-time updates disabled*.

- \_\_\_ 3. In the Integration Console, view the status of administration security and the current authorization mode. Type:

```
mqsireportauthmode IIBNODE_NOQM
```

You should see the following response:

```
BIP8930I: Integration node name 'IIBNODE_NOQM'  
Administration security = 'inactive'  
Authorization mode = 'file'
```

As expected the administration security is returned as `inactive`. Also, this integration node was created without a default queue manager, so default authorization mode is `file`.

- \_\_\_ 4. Activate administration security on IIBNODE\_NOQM. In the IBM Integration Console, type:

```
mqsichangeauthmode IIBNODE_NOQM -s active -m file
```

You should see the following response:

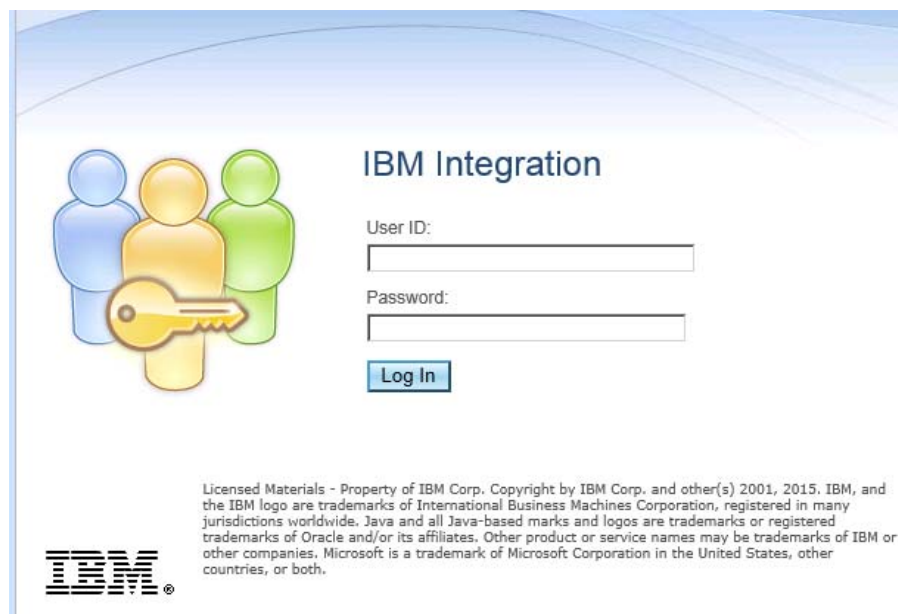
```
BIP8071I: Successful command completion.
```

- \_\_\_ 5. Restart the integration node. In the IBM Integration Console, type:

```
mqsistart IIBNODE_NOQM
```

- \_\_\_ 6. After a short pause, reload the IBM Integration web user interface by clicking the indicator in the address page.

You are now presented with a sign-on screen because administrative security is now active.



## Part 2: Define administration roles and set file-based permissions.

In this part of the exercise, you create administration roles and grant administration authorities to these roles. You reference these roles in the Integration Bus web user definitions that you create later in this exercise.

Three levels of authorization are supported for IBM Integration Bus administration security: read, write, and execute. You can assign permissions to a role (user) by specifying the type of permission followed by a plus (+) to grant permissions, or a minus (-) to revoke permissions.

The following table shows the file-based authorities that are required for different types of users in this exercise:

<u>Role</u>	<u>Authority</u>
<b>iibRole1</b>	Read (view resources)
<b>iibRole2</b>	Read, write (view resources, create integration servers, and modify settings)
<b>iibRole3</b>	Read, write, execute (start, stop, deploy, and modifying resources)

- \_\_\_ 1. Check whether there are any roles that are defined on the integration node. In the Integration Console, type:

```
mqsireportfileauth IIBNODE_NOQM -l
```

You should see the response:

```
BIP8071I: Successful command completion.
```

No defined roles are returned.

- \_\_\_ 2. Create the **iibRole1** role. In the Integration Console, type:

```
mqsichangefileauth IIBNODE_NOQM -r iibRole1 -p read+
```

- \_\_\_ 3. Create the **iibRole2** role. In the Integration Console, type:

```
mqsichangefileauth IIBNODE_NOQM -r iibRole2 -p read+,write+
```

- \_\_\_ 4. Create the **iibRole3** role. In the Integration Console, type:

```
mqsichangefileauth IIBNODE_NOQM -r iibRole3 -p all+
```

- \_\_\_ 5. Enter the command for displaying any defined roles again. In the Integration Console, type:

```
mqsireportfileauth IIBNODE_NOQM -l
```

The returned response should be as follows:

```
BIP8931I: Role = 'iibRole1', Resource = '', Permissions =  
'read+,write-,execute-
```

```
BIP8931I: Role = 'iibRole2', Resource = '', Permissions =  
'read+,write+,execute-
```

```
BIP8931I: Role = 'iibRole3', Resource = '', Permissions =  
'read+,write+,execute+
```

```
BIP8071I: Successful command completion.
```

- \_\_\_ 6. Define an Integration Bus web user that has read-only access.

This user can see deployed applications, but cannot control the status of these applications.

In an Integration Console, type:

```
mgsiwebuseradmin IIBNODE_NOQM -c -u admin1 -a passwd -r iibRole1
```

This command defines a new web user, `admin1`. The user has the security profile that the **iibRole1** role defines. In this case, the user can view the integration node and any deployed applications.

- \_\_\_ 7. Define an Integration Bus web user that has read and write access.

This user can see deployed applications, and can change properties on the integration node and integration server.

In the Integration Console, type:

```
mgsiwebuseradmin IIBNODE_NOQM -c -u admin2 -a passwd -r iibRole2
```

This command defines a new web user, `admin2`. This user has the security profile that the **iibRole2** role defines. In this case, the user can view the integration node and integration servers, and edit their properties. This user can also view the deployed resources.

- \_\_\_ 8. Define an Integration Bus web user for all access.

This user can see deployed applications, and can control all the integration node resources.

In the Integration Console, type:

```
mgsiwebuseradmin IIBNODE_NOQM -c -u admin3 -a passwd -r iibRole3
```

This command defines a new web user, `admin3`. This user has the security profile that the **iibRole3** role defines. In this case, the user can view the integration node, integration servers, and any deployed applications. This user can also control all resources.

- \_\_\_ 9. Display the defined web users. In the Integration Console, type:

```
mgsiwebuseradmin IIBNODE_NOQM -l
```

The response should be similar to the following example:

```
BIP2837I: Web user 'admin1' is defined as having a role of 'iibRole1'.  
BIP2837I: Web user 'admin2' is defined as having a role of 'iibRole2'.  
BIP2837I: Web user 'admin3' is defined as having a role of 'iibRole3'.
```

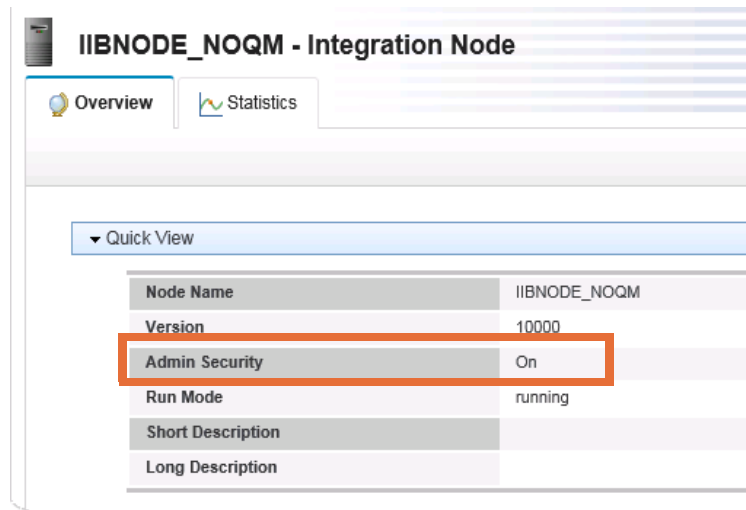
### Part 3: Test the IBM Integration web interface with security

In this part of the exercise, you log in the IBM Integration web interface and test each user role.

- \_\_\_ 1. Log in to the IBM Integration web user interface as the `admin1` user. The role for this user is "read-only".

In the IBM Integration browser window for the `IIBNODE_NOQM` integration node, login with the user ID `admin1` (password is `passw0rd`).

- \_\_\_ 2. Verify that Quick View properties for `IIBNODE_NOQM` show that **Admin Security** is **On**.



- \_\_\_ 3. In the Integration web interface, expand the **Servers** folder.  
You should see the available integration servers but you cannot view any of the resources.
- \_\_\_ 4. You gave the role **iibRole1** 'read' authorities at an integration node level. Setting integration server authority provides more granular approach to authorization. You can authorize each integration server with a separate command.

Also, with Integration Bus file-based authorization, you can change the roles' permissions without a restart of the integration node; changes are recognized dynamically.

Change the integration server authorization so that the `admin1` user with role **iibRole1** can view the integration server resources. In the Integration Console, type:

```
mqsischangeauth IIBNODE_NOQM -e server1 -r iibRole1 -p read+
```

The `-e server1` parameter specifies the integration server, which means that now you are applying the permissions at the integration server level.

- \_\_\_ 5. In the IBM Integration web interface, refresh the page.
- \_\_\_ 6. Expand the **server1** integration server.

You can now view the resources on the integration server.

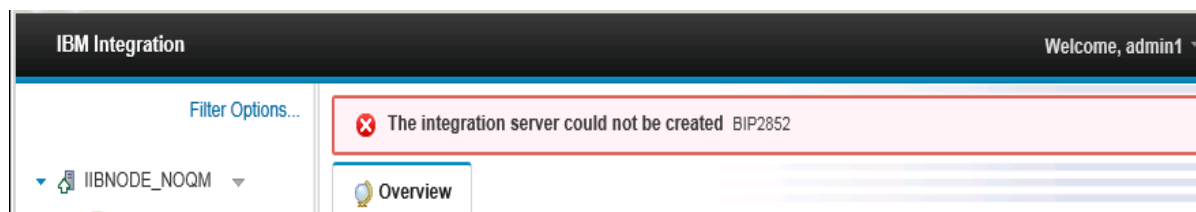
Although you can expand the **server1** resources folders, you do not have permission to start, stop, or other actions to the deployed artifacts.

- \_\_\_ 7. Try to create an integration server.
  - \_\_\_ a. Click the arrow to the right of **Servers** and then click **Create**.



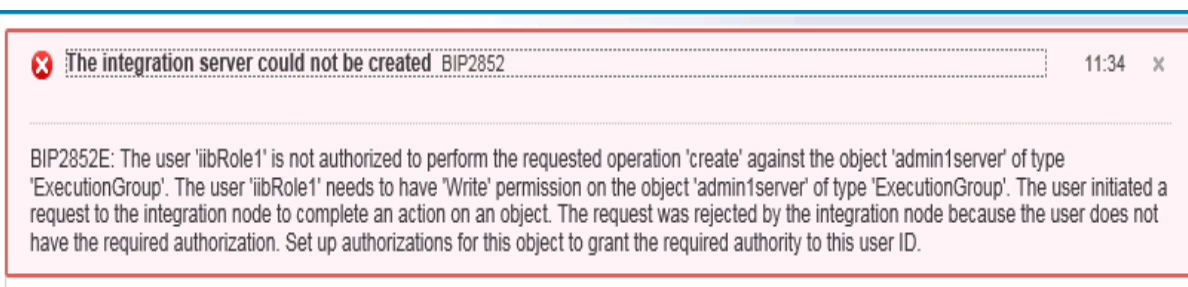
- \_\_ b. In the dialog box, type `admin1server` and then click **OK**.

You should see a message that the integration server could not be created.



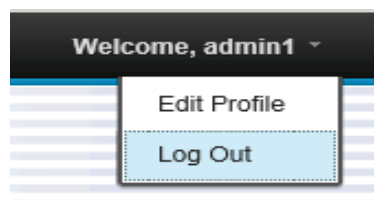
- \_\_ c. Hover with your mouse over the notification message. You should see that this message is a hyperlink. Click the link.

You see a detailed message, explaining why the attempt to create the integration server failed.



- \_\_ 8. After exploring the options for the user `admin1`, log out of the IBM Integration web user interface.

Click the arrow next to the **Welcome, admin1** banner and then click **Log Out**.



- \_\_ 9. Log in to the IBM Integration web user interface as the `admin2` user, which is the user with read and write permissions. The password is `passw0rd`.

This user can view resources, edit integration node properties, and create Integration servers.

- \_\_ 10. Expand **Servers**.

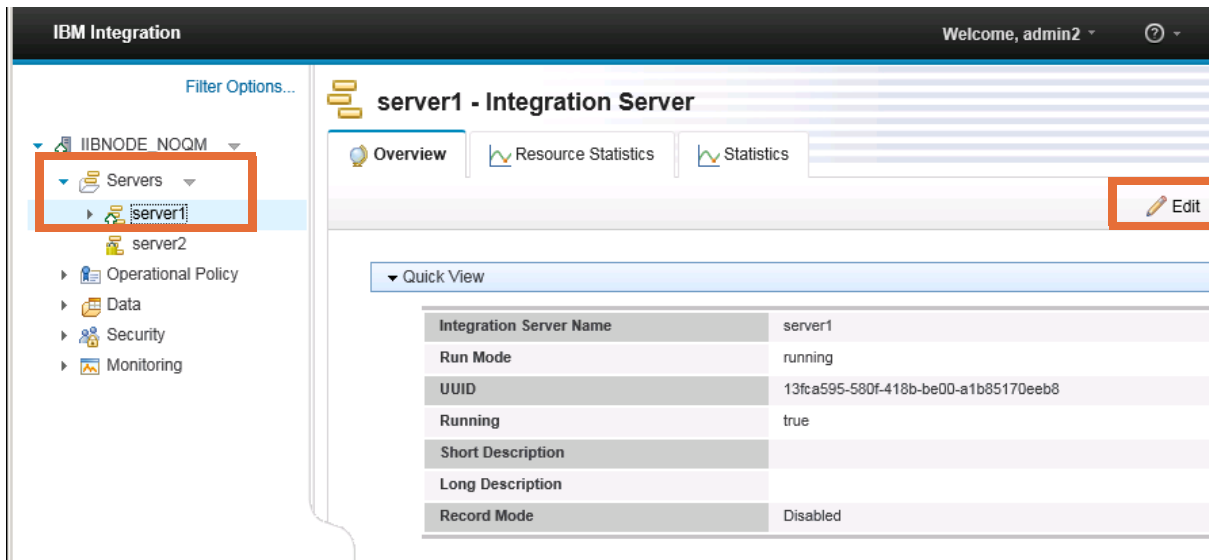
You should see that although you can view the available servers, the role to which `admin2` is associated is not authorized to access the integration server. The role for `admin2` has read and write access for the integration node, but not for any integration servers.

- \_\_ 11. Set read and write access to the **server1** integration server for role **iibRole2**. In the Integration Console, type:

```
msgchangeauth IIBNODE_NOQM -e server1 -r iibRole2 -p read+,write+
```

This command allows this role to view the resources on the **server1** integration server.

- \_\_\_ 12. In the IBM Integration web user interface, refresh the login for `admin2` (press F5 or click **Refresh**).
- \_\_\_ 13. Click the **server1** integration server in the Integration web interface. You now see an **Edit** icon on the properties view (which was not available for user `admin1`).



- \_\_\_ 14. Click **Edit**.
- The table with the properties for the **server1** integration server should open. The user `admin2` should now have the authority to change the integration server properties.
- \_\_\_ 15. You do not change the integration server properties in this exercise. When you are done reviewing the properties, click **Cancel**.
- \_\_\_ 16. The user `admin2` has 'write' authorities, which allows the user to create an integration server on the integration node.
- Try to create an integration server that is named `admin2server`.
- This action should succeed because `admin2` has the required authorization.
- \_\_\_ 17. Even though `admin2` can create the integration server, the administrator must authorize the role to which `admin2` is associated to allow the user to control the new integration server.
- In the Integration Console, type;
- ```
mqsichangefileauth IIBNODE_NOQM -e admin2server -r iibRole2 -p read+,write+
```
- This command sets the permissions for this role on the **admin2server** integration server as read and write.
- \_\_\_ 18. Refresh the IBM Integration web user interface. If necessary, log in as `admin2` again.
- Now, the web user can view the new integration server and its resources (although no resources are currently deployed). The user can also edit the integration server properties.
- \_\_\_ 19. Log out of the IBM Integration web user interface.
- \_\_\_ 20. Log in to the IBM Integration web user interface as `admin3`.

This user has full control for the integration node resources, which includes stopping, starting, and deploying applications and starting and stopping statistics.

- \_\_\_ 21. Although `admin3` has full authorities for the resources on the integration node, the administrator must provide extra permissions to **iibRole3** so that the user can manage the integration server.

In the Integration Console, type:

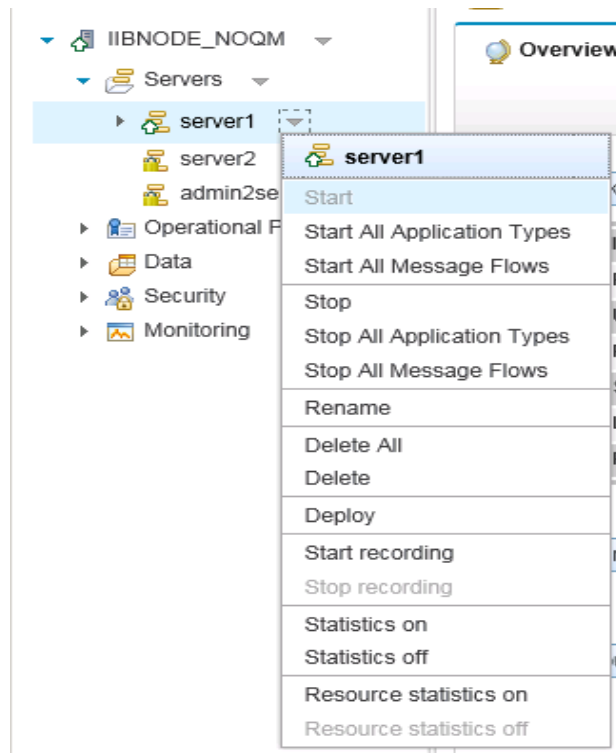
```
mqsichangefileauth IIBNODE_NOQM -e server1 -r iibRole3 -p all+
```

- \_\_\_ 22. Refresh the Integration web interface and then expand the **Servers > server1**.

You should see that now there is an arrow on the right side of **server1**, which was not there for `admin1` and `admin2` web users.

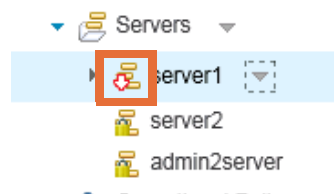
Click the arrow to display the menu.

You should see a menu that contains all the administration tasks that web user `admin3` can complete.

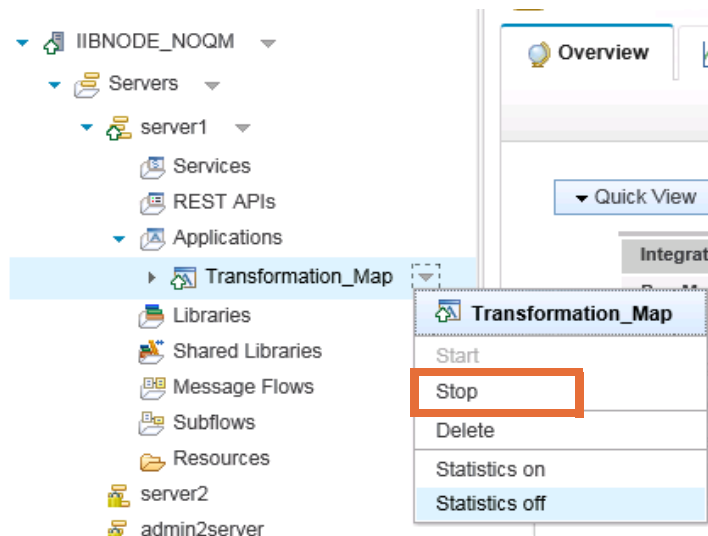


- \_\_\_ 23. Click **Stop** to stop the integration server.

After a few seconds you should see that the integration server stopped, as indicated by the down pointing red arrow.



- \_\_\_ 24. Expand the **server1** menu again and then click **Start** to start the integration server.
- \_\_\_ 25. When the integration server starts, expand the **Applications** group and then click **Transformation\_Map**.
- \_\_\_ 26. Click **Stop** on the Transformation\_Map menu.



The application and all its flows are stopped, as the down pointing red arrows indicated.

- \_\_\_ 27. Delete the **Transformation\_Map** application.
- \_\_\_ 28. Log out user as user admin3.

## Exercise clean up

- \_\_\_ 1. Deactivate administration security on the IIBNODE\_NOQM integration node.

In the Integration Console, type:

```
mqsistop IIBNODE_NOQM
mqsichangebroker IIBNODE_NOQM -s inactive
mqsistart IIBNODE_NOQM
```

- \_\_\_ 2. Delete the integration server **admin2server**.

In the Integration Console, type:

```
mqsdeleteexecutiongroup IIBNODE_NOQM -e admin2server
```

## End of exercise

## Exercise review and wrap-up

In the first part of this exercise, you activated file-based security and then created three user roles.

In the second part of this exercise, created web users and assigned them to one of the defined roles.

In the third part of this exercise, you explored the IBM Integration web user interface as each user and configured security authorizations for integration servers.

Having completed this exercise, you should be able to:

- Activate administration authority
- Use IBM Integration Bus file-based security to assign permissions for integration nodes, integration servers, and message flows



## Exercise 6. Using queue-based security to control administration access

### What this exercise is about

In this exercise, you use the IBM Integration Bus security queues on IBM MQ to set and test IBM Integration Bus access control for the integration nodes, integration servers, and message flows.

### What you should be able to do

After completing this exercise, you should be able to:

- Activate queue-based administration authority
- Use the IBM Integration Bus security queues on IBM MQ to assign permissions for integration nodes, integration servers, and message flows

### Introduction

IBM Integration Bus can use a special set of IBM MQ SYSTEM.BROKER queues to control access to each entity that must be secured in terms of administrative access. Each administrative action, such as creating an integration server, and deploying or stopping a message flow, is mapped to one of three categories; read, write, or run. These categories are mapped to IBM MQ security properties as follows:

- *Inquire* maps to *Read*
- *Put* maps to *Write*
- *Set* maps to *Run*

One or more of the IBM MQ security permissions are set on a particular queue to control access to the object that queue represents; an integration node or an integration server. This design uses a familiar, already present mechanism (queue security) and provides excellent granularity and flexibility.

In the first part of this exercise, you set up user and group authorities for a non-privileged Windows user (*No\_Auth\_ID*). That same user ID is used in testing the next parts of this lab.

In the second part of the exercise, you activate IBM Integration Bus security. You then use the same non-privileged user ID (*No\_Auth\_ID*) to show that the user cannot use any functions in the IBM Integration Bus environment.

In the third part of the exercise, IBM Integration Bus security is activated. The same non-privileged user ID is used to demonstrate that the user cannot use

any functions in the Integration Bus environment until authorization is configured on the IBM MQ authorization queues.



#### Note

This exercise uses the IBM Integration API Exerciser to test queue-based security. It is important that you understand that all Integration Bus applications are subject to security authentication when administrative security is active. Those applications include the IBM Integration Toolkit and the IBM Integration web interface.

## Requirements

- IBM Integration Bus V10 and IBM MQ V8
- The IIBNODE\_WITHQM integration node and the IIBMQ IBM MQ manager that were created in Exercise 2
- Lab files in the C:\labfiles\Lab06-QSec directory
- Microsoft Windows user that is named No\_Auth\_ID and group that is named Security\_Test with IBM Integration Bus permissions as shown in Part 1 of the exercise



## Exercise instructions

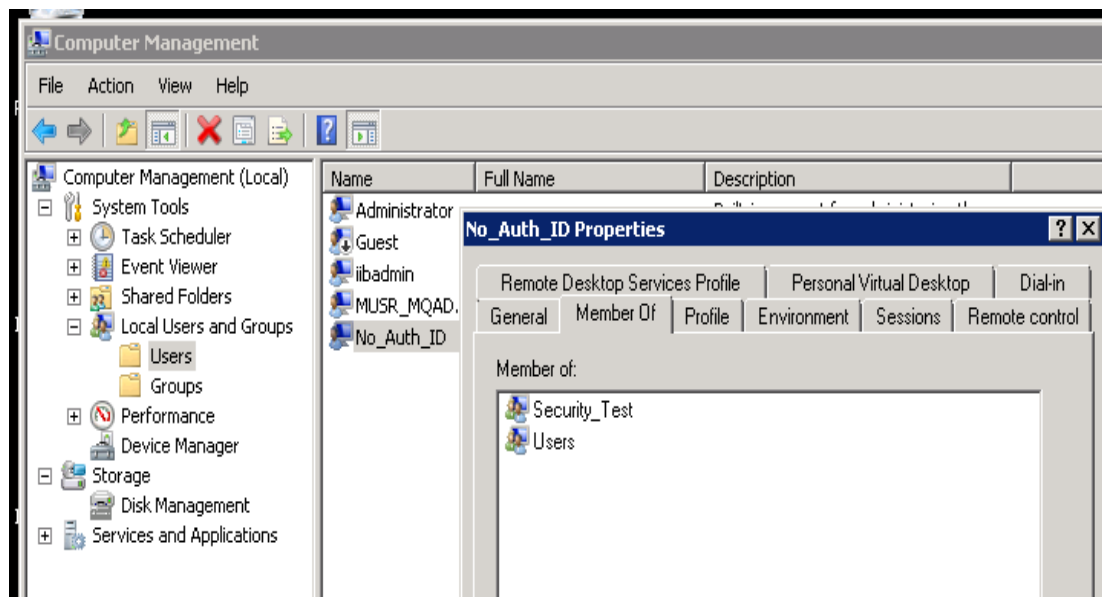
### Exercise set up

- \_\_\_ 1. To effectively test administrative security, you must run under a different user ID than the administrator ID `iibadmin` because it has full authority over the integration node environment. Logging out of that account and logging in as another user is not a good solution on Windows because that starts a different or unique configuration of installed software. The solution is to use the Windows option to **Run as different user**. With this option, you can run a program with different permissions than the currently logged-on user ID provides.

In this exercise, you use the **Run as different user** option in this lab to run the IBM Integration API Exerciser under the `No_Auth_ID` user ID. The `No_Auth_ID` is a member of the group `Security_Test`. The `No_Auth_ID` user ID is not a member of the `mqbrkrs` or `mqm` groups.

Verify that you have a user ID that is named `No_Auth_ID` user ID and that this user is member of the group `Security_Test`.

- \_\_\_ a. Start the Windows Computer Management application by clicking **Start > Administrative Tools > Computer Management**.
- \_\_\_ b. In the Computer Management window, expand **Local Users and Groups**.
- \_\_\_ c. Click **Users**. A list of all users and groups is displayed in the right pane.
- \_\_\_ d. In the right pane, double-click the user ID `No_Auth_ID`.
- \_\_\_ e. Click the **Member of** tab and review the groups to which the user ID belongs. Notice that the user ID is *not* a member of the `mqbrkrs`, `mqm`, or **Administrators** groups.
- \_\_\_ f. Click **Cancel** to exit without saving any changes to the user ID.



- \_\_\_ g. Close the Computer Management window.

- \_\_\_ 2. In this exercise, you use the IIBNODE\_WITHQM integration node with the default queue manager IIBQM that were created in Exercise 2.

Verify that the IIBNODE\_WITHQM integration node and IIBQM are created and running.

- \_\_\_ a. Start IBM MQ Explorer if it is not already running. Verify that the queue manager IIBQM is defined and running.
- \_\_\_ b. In an IBM Integration Console, type the following command to verify that the integration node IIBNODE\_WITHQM is running.

```
mqsilist IIBNODE_WITHQM
```

- \_\_\_ 3. Start the IBM Integration web interface for IIBNODE\_WITHQM.

If you do not know the HTTP Connector port for the web interface, type the following command in the Integration Console.

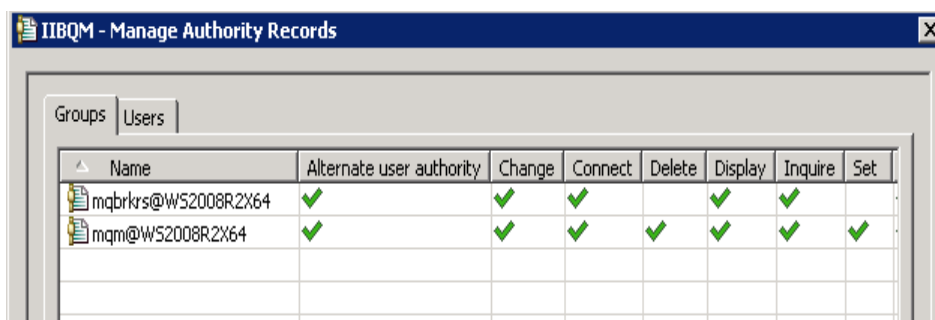
```
mqsiireportproperties IIBNODE_WITHQM -b webadmin -o HTTPConnector -a
```

## Part 1: Set IBM MQ permissions for users

If you are using any Integration Bus functions that require access to IBM MQ, permissions are required to enable users to connect to the integration node.

In this part of this exercise, you set up IBM MQ permissions for a non-administrator users (No\_Auth\_ID) that is a member of the Security\_Test group. That same user ID is used in testing the next parts of this exercise.

- \_\_\_ 1. In IBM MQ Explorer, grant the group Security\_Test **Connect** and **Inquire** privileges for the integration node queue manager IIBQM.
- \_\_\_ a. Under the **Queue Managers** folder, right-click **IIBQM** and then click **Object Authorities > Manage Queue Manager Authority Records**. The Manage Authority Records window is displayed.



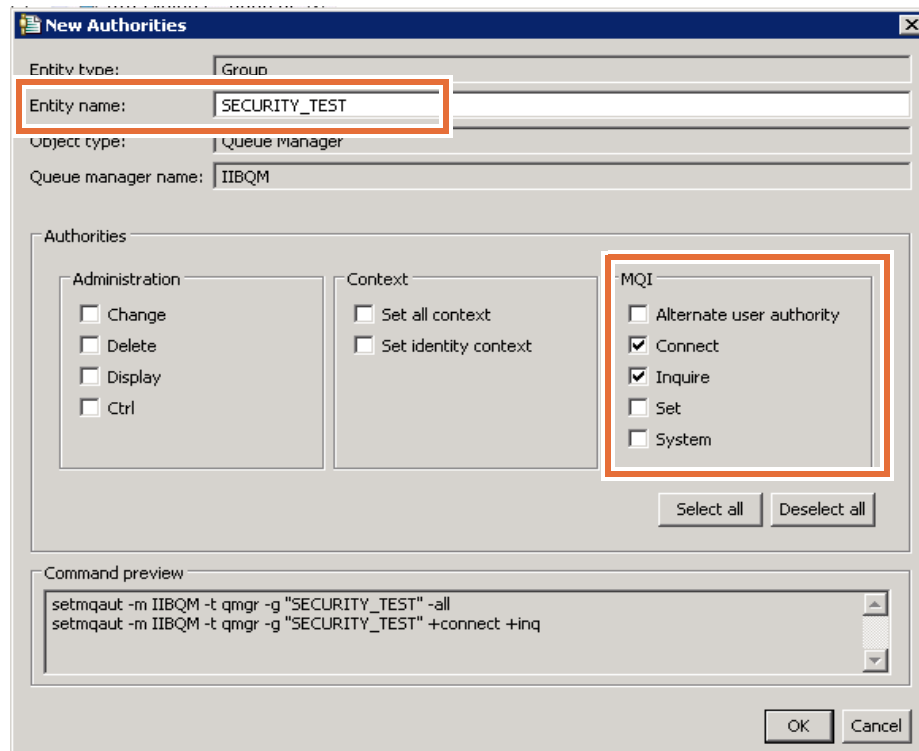
- \_\_\_ b. Review the existing groups and users by clicking the **Groups** tab and then the **Users** tab.

The only two groups that you see are mqm and mqbrkrs.

The only user that you see is iibadmin.

- \_\_\_ c. On the **Groups** tab, click **New** (at the bottom of the window). The **New Authorities** dialog box is displayed.
- \_\_\_ d. For **Entity name**, type: Security\_Test

- \_\_\_ e. Under the **MQI** authorities options, click **Connect** and **Inquire**.

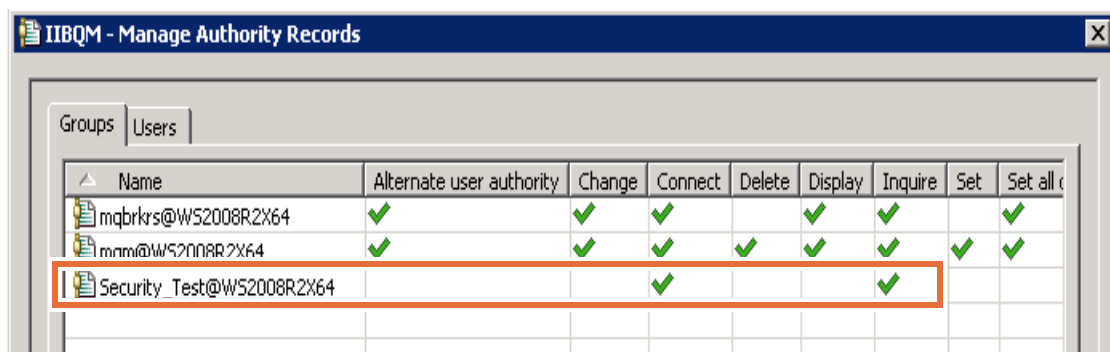


The 'New Authorities' dialog box is shown. The 'Entity name' field is set to 'SECURITY\_TEST'. The 'MQI' section is highlighted with a red box, showing 'Connect' and 'Inquire' checked. The 'Command preview' section shows the following commands:

```
setmqaut -m IIBQM -t qmgr -g "SECURITY_TEST" -all
setmqaut -m IIBQM -t qmgr -g "SECURITY_TEST" +connect +inq
```

- \_\_\_ f. Click **OK**, and then click **OK** again on the result dialog box.

The `Security_Test` group is displayed in the **Groups** list with **Connect** and **Inquire** authority.



The 'IIBQM - Manage Authority Records' window is shown. The 'Groups' tab is selected. The table below shows the authority records for the groups listed.

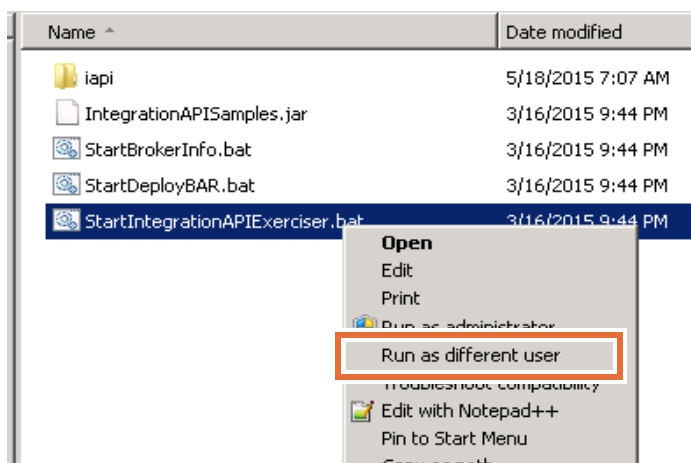
| Name                      | Alternate user authority | Change | Connect | Delete | Display | Inquire | Set | Set all c |
|---------------------------|--------------------------|--------|---------|--------|---------|---------|-----|-----------|
| mqbrkrs@WS2008R2X64       | ✓                        | ✓      | ✓       |        | ✓       | ✓       |     | ✓         |
| mnm@WS2008R2X64           | ✓                        | ✓      | ✓       | ✓      | ✓       | ✓       | ✓   | ✓         |
| Security_Test@WS2008R2X64 |                          |        | ✓       |        |         | ✓       |     |           |

- \_\_\_ g. Close the Manage Authority Records window.

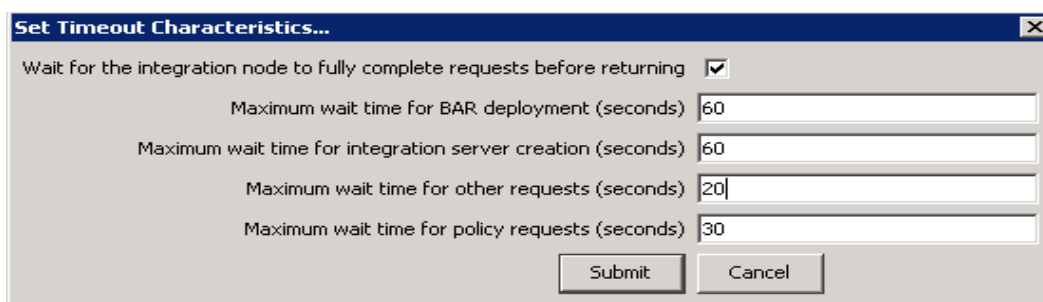
## Part 2: Test the default unsecured environment

In this part of the exercise, you verify that permissions you configured in Part 1 of this exercise are correct and that the No\_Auth\_ID user can successfully connect to the IIBNODE\_WITHQM integration node.

- \_\_\_ 1. So that you can run as another user without having to log off and back on, use the **Run as different user** option in Windows and start the Integration API Exerciser.
  - \_\_\_ a. In Windows Explorer, browse to the C:\Program Files\IBM\IIB\10.0.0.0\server\sample\IntegrationAPI directory.
  - \_\_\_ b. While holding down the Shift key, right-click the **StartIntegrationAPIExerciser.bat** file and then click **Run as different user**.



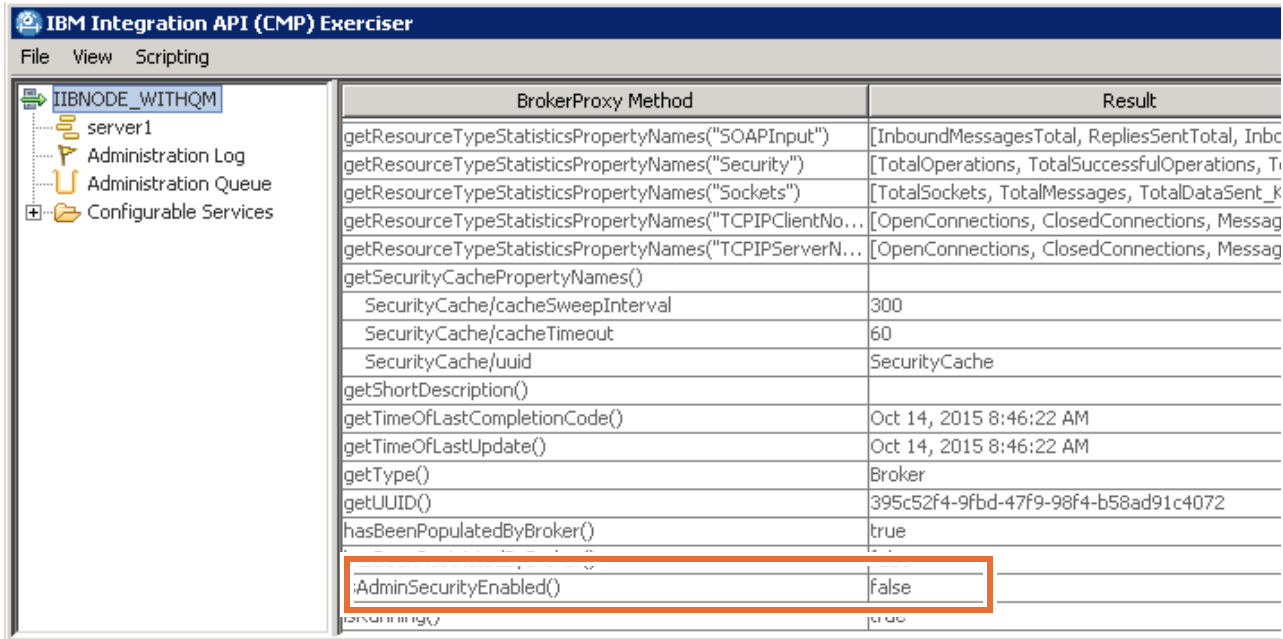
- \_\_\_ c. A login window opens and you are prompted to enter the user ID and password.  
 For the user ID, type: No\_Auth\_ID  
 For the password, type: web1sphere  
 After a few moments, the Integration API Exerciser starts.
- \_\_\_ 2. Adjust the timeout values for the Integration API Exerciser.
  - \_\_\_ a. Click **File > Set Timeout Characteristics**.
  - \_\_\_ b. Modify the timeout values as follows:
    - Set **Maximum wait time for BAR deployment** to 60
    - Set **Maximum wait time for integration server creation** to 60
    - Set **Maximum wait time for other requests** to 20



- \_\_\_ c. Click **Submit**.
- \_\_\_ 3. The next steps assume that the IIBNODE\_WITHQM integration node is running.  
If the integration node IIBNODE\_WITHQM is not running, start it now.
- \_\_\_ 4. The Integration API Exerciser is running under No\_Auth\_ID and gets all permissions from the settings for that user and its group, which is Security\_Test.  
Connect the Integration API Exerciser to the integration node.
- \_\_\_ a. Right-click **Right-click to connection** and then click **Connect to Remote Integration node**.
- \_\_\_ b. Specify the connection values.
- For the **Host Name**, type: 127.0.0.1
  - For the **Port**, type: 4416
  - For the **User Name**, type: No\_Auth\_ID
  - For the **Password**, type: web1sphere

- \_\_\_ c. Click **Submit**.  
After a few seconds, the Integration API Exerciser connects to the integration node and displays the integration node properties.
- \_\_\_ 5. Using the Integration API Exerciser, verify that Administrative Security is turned off in IIBNODE\_WITHQM.
- \_\_\_ a. With the **IIBNODE\_WITHQM** integration node selected in the Integration API Exerciser navigator (left pane), scroll to the bottom of the data window (right pane).

- \_\_ b. Verify that **isAdminSecurityEnabled()** shows false, indicating that administrative security is not enabled.



| BrokerProxy Method                                       | Result   |
|--|--|
| getResourceTypeStatisticsPropertyNames("SOAPInput")      | [InboundMessagesTotal, RepliesSentTotal, Inbc  |
| getResourceTypeStatisticsPropertyNames("Security")       | [TotalOperations, TotalSuccessfulOperations, T |
| getResourceTypeStatisticsPropertyNames("Sockets")        | [TotalSockets, TotalMessages, TotalDataSent_k  |
| getResourceTypeStatisticsPropertyNames("TCPIPClientNo... | [OpenConnections, ClosedConnections, Messag    |
| getResourceTypeStatisticsPropertyNames("TCPIPServerN...  | [OpenConnections, ClosedConnections, Messag    |
| getSecurityCachePropertyNames()                          |  |
| SecurityCache/cacheSweepInterval                         | 300  |
| SecurityCache/cacheTimeout                               | 60   |
| SecurityCache/uuid                                       | SecurityCache                                  |
| getShortDescription()                                    |  |
| getTimeOfLastCompletionCode()                            | Oct 14, 2015 8:46:22 AM                        |
| getTimeOfLastUpdate()                                    | Oct 14, 2015 8:46:22 AM                        |
| getType()  | Broker   |
| getUUID()  | 395c52f4-9fbd-47f9-98f4-b58ad91c4072           |
| hasBeenPopulatedByBroker()                               | true   |
| isAdminSecurityEnabled()                                 | false  |

When the Integration API Exerciser is connected to the integration node, information about the integration node is displayed under the integration node in the navigation pane.

The display of the integration server **server1** in the Integration API Exerciser implies that some level of access is granted to the user. In this case, the No\_Auth\_ID user has “read” access to the integration node.

- \_\_ 6. Attempt to deploy a BAR as the No\_Auth\_ID user.
- \_\_ a. Click the **server1** integration server in the Integration API Exerciser navigator.
- \_\_ b. Right-click **server1** and then click **Deploy BAR** from the menu.



#### Note

Be sure to select an entity before right-clicking it in the IBM Integration API Exerciser.

- \_\_ c. Browse to the C:\labfiles\Lab06-QSec\resources directory, select Admin\_Security\_Test1.bar, and then click **Open**.

The request succeeds, as indicated in the status log in the bottom pane of the IBM Integration API Exerciser.

```

10:06:45 AM DeployResult.getDeployStartTime() = Oct 14, 2015 10:06:32 AM
10:06:45 AM DeployResult.getDeployStopTime() = Oct 14, 2015 10:06:45 AM (12s)
10:06:45 AM DeployResult.getDeployResponses() : BIP2871I: The request made by user 'SYSTEM[Default]' to 'deploy'
'C:\labfiles\Lab06-QSec\resources\Admin_Security_Test1.bar' of type 'BAR' on parent 'server1' of type 'ExecutionGroup'
'COMPLETE'.
10:06:45 AM
10:06:45 AM SUCCESS: The integration node completed the request.
10:06:45 AM
10:06:45 AM <---- iapi.exerciser.ClassTesterForExecutionGroupProxy.testDeployBAR
10:06:45 AM

```

- \_\_\_ 7. Determine whether this user can stop a message flow.
  - \_\_\_ a. Fully expand the **server1** integration server in the navigator.
  - \_\_\_ b. Right-click **Admin\_Security1\_MsgFlow.cmf** and click **Stop** from the menu.
  - \_\_\_ c. Verify that the message flow stopped by selecting the message flow and checking the **runMode** status, which should now be set to **stopped**.

| MessageFlowProxy Method              | Result  |
|--------------------------------------|---|
| traceLevel                           | none  |
| additionalInstances                  | 0   |
| startMode                            | Maintained  |
| coordinatedTransaction               | no  |
| commitInterval                       | 0   |
| version                              |   |
| <b>runMode</b>                       | <b>stopped</b>  |
| running                              | false   |
| getCommitCount()                     | 1   |
| getCommitInterval()                  | 0   |
| getConfigurationObjectType()         | <Message Flow>  |
| getConfigurationObjectTypeOfParent() | <Execution Group>   |
| getCoordinatedTransaction()          | false   |
| getDeployProperties()                |   |
| modifyTime                           | 2009-06-18 16:36:40.000 -0700                             |
| deployTime                           | 2015-10-14 10:06:32.925 -0700                             |
| barFileName                          | C:\labfiles\Lab06-QSec\resources\Admin_Security_Test1.bar |

- \_\_\_ 8. Restart the flow by right-clicking **Admin\_Security1\_MsgFlow.cmf** and clicking **Start** from the menu.
- \_\_\_ 9. Verify that the flow is started by reviewing the status log. The **runMode** status should now be set to **running**.
- \_\_\_ 10. Disconnect the Integration API Exerciser from the integration node by right-clicking **IIBNODE\_WITHQM** in the navigation pane and clicking **Disconnect**.

This step completes the testing in the unsecured environment. You saw that, given a minimum set of authorizations, the **No\_Auth\_ID** has access to the integration node.

Do not exit from the IBM Integration API Exerciser. You use it again in the next part of the exercise.

### **Part 3: Activate administration security and test the secured environment**

In this part of the lab, IBM Integration Bus security is activated, and the same non-privileged user ID is used to demonstrate that the user cannot use any functions in the IBM Integration Bus environment.

\_\_ 1. Activate integration node administration security.

\_\_ a. Start an IBM Integration Console session as the administrator.

\_\_ b. Stop the integration node so that you can make the necessary configuration change.  
Type:

```
mqsisstop IIBNODE_WITHQM
```

\_\_ c. View the status of administration security and the current authorization mode. Type:

```
mqsireportauthmode IIBNODE_WITHQM
```

You should see the following response:

```
BIP8930I: Integration node name 'IIBNODE_WITHQM'  
Administration security = 'inactive'  
Authorization mode = 'mq'
```

As expected the administration security is returned as *inactive*. Also, this integration node was created with a default queue manager, so default authorization mode is *mq*.

\_\_ d. Activate administration security on IIBNODE\_WITHQM. Type:

```
mqsichangeauthmode IIBNODE_WITHQM -s active -m mq
```

\_\_ e. Restart the integration node. Type:

```
mqsisstart IIBNODE_WITHQM
```

\_\_ 2. When you activate administration security, the Integration web interface requires a user name and password to log in.

\_\_ a. Enable the Integration web interface for the administration user *iibadmin*. In the Integration Console, type:

```
mqsiwebuseradmin IIBNODE_WITHQM -c -u iibadmin -a web1sphere -r iibadmin
```

\_\_ b. Log in to the Integration web interface with the administrator credentials.

For the user ID, type: *iibadmin*

For the password, type: *web1sphere*

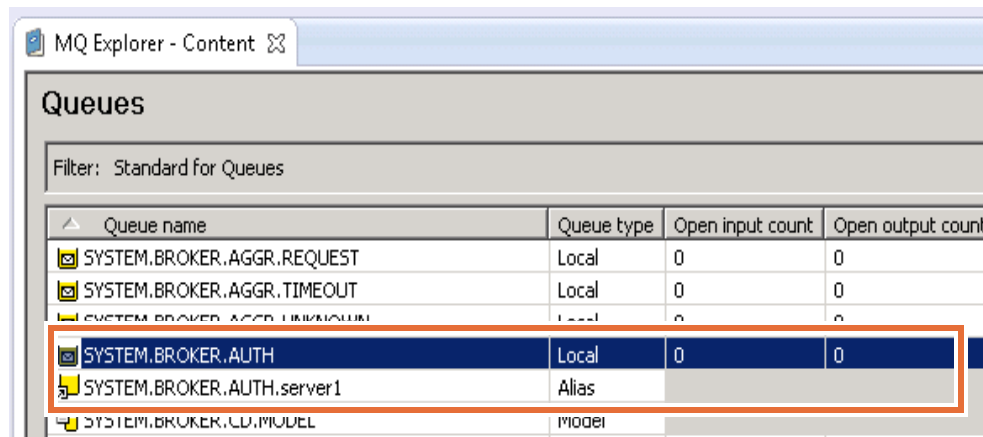
\_\_ 3. You must also enable the Integration web interface for the *No\_Auth\_ID* so that this user can connect to the integration nodes port.

In the Integration Console, type:

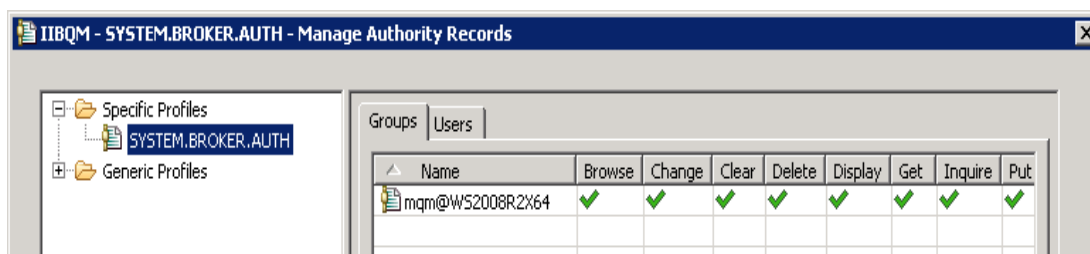
```
mqsiwebuseradmin IIBNODE_WITHQM -c -u No_Auth_ID -a web1sphere -r No_Auth_ID
```



- \_\_\_ 4. Verify that the authorization queues exist on the queue manager.
- \_\_\_ a. In IBM MQ Explorer, click **Queues** under the IIBQM queue manager to display the **Queues** content view.
  - \_\_\_ b. In the Queues content view, click the **Show System Object** icon to display the SYSTEM queues.
  - \_\_\_ c. Confirm that the following authorization queues exist:
    - SYSTEM.BROKER.AUTH (the authorization queue for the integration node)
    - SYSTEM.BROKER.AUTH.server1 (the authorization queue for integration server)



- \_\_\_ 5. The SYSTEM.BROKER.AUTH queue controls authorization for the integration. Give the group Security\_Test "Inquire" authority on the queue so that member of that group can view the integration node properties.
- \_\_\_ a. In the IBM MQ Explorer, right-click the SYSTEM.BROKER.AUTH queue and then click **Object Authorities > Manage Authority Records** from the menu.
  - \_\_\_ b. Expand **Specific Profiles** and select **SYSTEM.BROKER.AUTH**.



Security\_Test is not displayed in the profiles. It does not have permissions on the SYSTEM.BROKER.AUTH queue. The group must be defined to the queue and it must have Read ("inquire") authority.

- \_\_\_ c. Click the **Groups** tab, and then click **New**.
- \_\_\_ d. For the **Entity name**, type: Security\_Test
- \_\_\_ e. Under the **MQI** authorities options, click **Inquire**.
- \_\_\_ f. Click **OK**, and then click **OK** again on the result window.  
The Security\_Test group is displayed in the **Groups** list with **Inquire** authority.

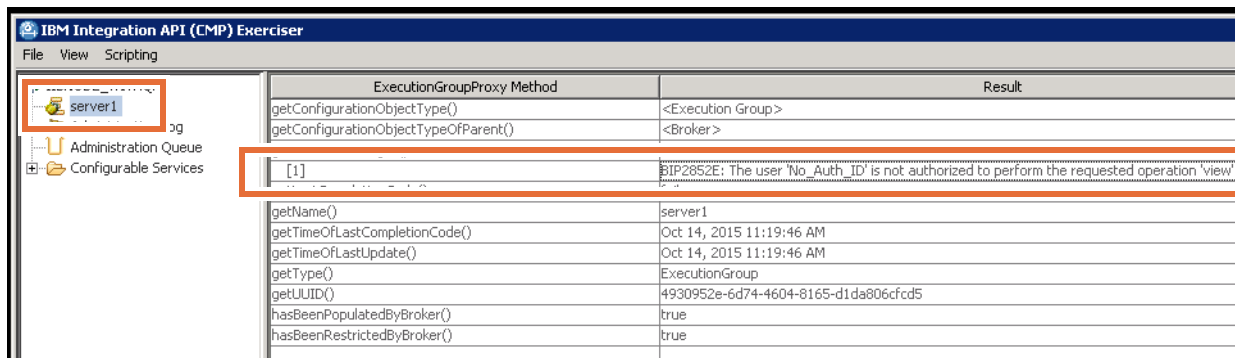
- \_\_\_ g. Click **Close** to close the Manage Authority Records window.
- \_\_\_ 6. In the IBM Integration API Exerciser, connect to IIBNODE\_WITHQM.

With the security changes, the queue and integration server shows in the navigation pane.

You can see the integration server **server1** but it has a lock icon next to it. You cannot see the integration server properties or the message flow that is running on the server. Granting authority to the integration node did not grant authority to the integration servers contained within it.

- \_\_\_ 7. Click the **server1** integration server. You should see the BIP2852 error message in the **Result** field that indicates the user is not authorized.

To list the contents of an integration server, the user ID must have Read ("inquire") access to the integration server.



- \_\_\_ a. In IBM MQ Explorer, **Queues** content view, right-click **SYSTEM.BROKER.AUTH.server1** queue and then click **Object Authorities > Manage Authority Records** from the menu.
- \_\_\_ b. Expand **Specific Profiles** and then click **SYSTEM.BROKER.AUTH.server1**.
- \_\_\_ c. Click the **Groups** tab, and then click **New**.
- \_\_\_ d. Enter `Security_Test` as the **Entity name**.
- \_\_\_ e. Under the **MQI** authorities options, select **Inquire**.
- \_\_\_ f. Click **OK**, and then click **OK** again on the result dialog box.  
The `Security_Test` group is displayed in the **Groups** list with **Inquire** authority.
- \_\_\_ g. Click **Close** to close the **Manage Authority Records** dialog box.
- \_\_\_ 8. In the Integration API Exerciser, right-click the **server1** integration server and then click **Refresh** from the menu.
- You now see the properties of the **server1** integration server. You can also see the message flow that is running on the integration server.
- \_\_\_ 9. Try to deploy a BAR file to server1 from the Integration API Exerciser.
- \_\_\_ a. Right-click the **server1** integration server and click **Deploy BAR** from the menu.
- \_\_\_ b. Go to the `C:\labfiles\Lab04-QSec\resources` directory and then select `Admin_Security_Test2.bar`.

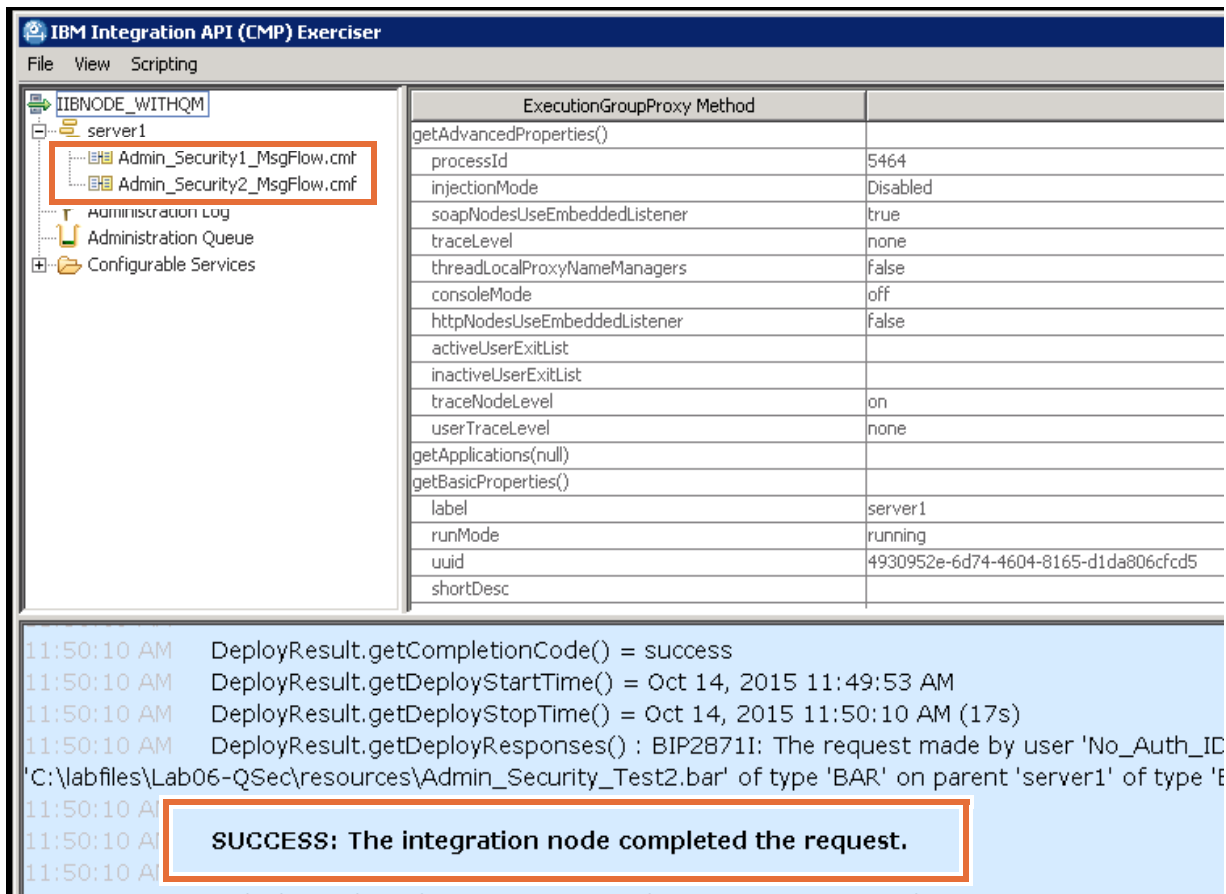
- \_\_\_ c. The deployment fails. The message in the Integration API Exerciser indicates that the user does not have authorization and must have “write” permissions on the integration server.

The ability to deploy requires Write (Put) permission on the integration server.

- \_\_\_ 10. Add Write (Put) permissions for the `Security_Test` group on the integration server **SYSTEM.BROKER.AUTH.server1** queue.
  - \_\_\_ a. In IBM MQ Explorer, **Queues** content view, right-click **SYSTEM.BROKER.AUTH.server1** queue and then click **Object Authorities > Manage Authority Records** from the menu.
  - \_\_\_ a. Expand **Specific Profiles** and then select **SYSTEM.BROKER.AUTH.server1**.
  - \_\_\_ b. Click the **Groups** tab, and then select the **Security\_Test** row in the table.
  - \_\_\_ c. Click **Edit**.
  - \_\_\_ d. Under the **MQI** authorities options, select **Put**.

The `Security_Test` group now has both Inquire (Read) and Put (Write) permissions.
  - \_\_\_ e. Click **OK**, and then click **OK** again on the result dialog box.
  - \_\_\_ f. Close the **Manage Authority Records** dialog box.
- \_\_\_ 11. Try to deploy the `Admin_Security_Test2.bar` in the Integration API Exerciser.

Verify that the flow was successfully deployed by reviewing the messages in the Integration API Exerciser status log. You should also see the message flow in the Integration API Exerciser navigator pane.



| ExecutionGroupProxy Method   |                                      |
|------------------------------|--------------------------------------|
| getAdvancedProperties()      |                                      |
| processId                    | 5464                                 |
| injectionMode                | Disabled                             |
| soapNodesUseEmbeddedListener | true                                 |
| traceLevel                   | none                                 |
| threadLocalProxyNameManagers | false                                |
| consoleMode                  | off                                  |
| httpNodesUseEmbeddedListener | false                                |
| activeUserExitList           |                                      |
| inactiveUserExitList         |                                      |
| traceNodeLevel               | on                                   |
| userTraceLevel               | none                                 |
| getApplications(null)        |                                      |
| getBasicProperties()         |                                      |
| label                        | server1                              |
| runMode                      | running                              |
| uuid                         | 4930952e-6d74-4604-8165-d1da806cfdc5 |
| shortDesc                    |                                      |

```

11:50:10 AM DeployResult.getCompletionCode() = success
11:50:10 AM DeployResult.getDeployStartTime() = Oct 14, 2015 11:49:53 AM
11:50:10 AM DeployResult.getDeployStopTime() = Oct 14, 2015 11:50:10 AM (17s)
11:50:10 AM DeployResult.getDeployResponses() : BIP2871I: The request made by user 'No_Auth_ID
'C:\labfiles\Lab06-QSec\resources\Admin_Security_Test2.bar' of type 'BAR' on parent 'server1' of type 'E
11:50:10 AM
11:50:10 AM
11:50:10 AM

```

**SUCCESS: The integration node completed the request.**

- \_\_\_ 12. Stop the `Admin_Security2_MsgFlow.cmf` message flow from the Integration API Exerciser.

Right-click `Admin_Security2_MsgFlow.cmf` in the Integration API Exerciser and then click **Stop**.

The stop command fails because the `Security_Test` group does not have Set (Run) authority on the integration server, which is required for a “stop” or “start” command.

- \_\_\_ 13. In IBM MQ Explorer, modify the permissions for the `Security_Test` group to include Set (Run) permissions on the **SYSTEM.BROKER.AUTH.server1** queue.

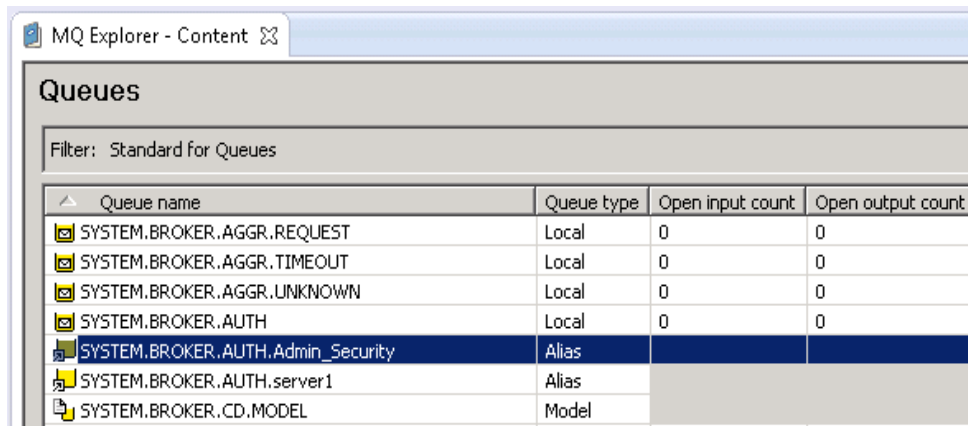
Review the substeps in step 10 if you need help. This time, select **Set** authority in addition to the authorities that are already set.

- \_\_\_ 14. In the Integration API Exerciser, try to stop the message flow now that the permissions are modified.

- \_\_\_ 15. To verify that the `Admin_Security2_MsgFlow.cmf` message flow stopped, scroll to the last entry in the **MessageFlowProxy Method** list. The last entry in the **Result** column is the *isRunning* method with a result of `false`, which indicates that the flow is stopped.

| MessageFlowProxy Method              | Result  |
|--------------------------------------|---|
| traceLevel                           | none  |
| additionalInstances                  | 0   |
| startMode                            | Maintained  |
| coordinatedTransaction               | no  |
| commitInterval                       | 0   |
| version                              |   |
| runMode                              | stopped   |
| isRunning                            | false   |
| getCommitCount()                     | 1   |
| getCommitInterval()                  | 0   |
| getConfigurationObjectType()         | <Message Flow>  |
| getConfigurationObjectTypeOfParent() | <Execution Group>   |
| getCoordinatedTransaction()          | false   |
| getDeployProperties()                |   |
| modifyTime                           | 2009-06-18 15:58:08.000 -0700                             |
| deployTime                           | 2015-10-14 11:49:53.819 -0700                             |
| barFileName                          | C:\labfiles\Lab06-QSec\resources\Admin_Security_Test2.bar |

- \_\_\_ 16. Attempt to create an integration server in the Integration API Exerciser.
- \_\_\_ a. Right-click **IIBNODE\_WITHQM** and then click **Create integration server** from the menu.
  - \_\_\_ b. Enter `Admin_Security` for the **New integration server name** and then click **Submit**.  
The requested action fails because the `Security_Test` group does not have Write (Put) permission on the integration node authorization queue.
- \_\_\_ 17. In IBM MQ Explorer, add Write (Put) permissions to the `SYSTEM.BROKER.AUTH` queue for `Security_Test`.  
When you complete this step, the `Security_Test` group should have both Read (Inquire) and Write (Put) permissions.
- \_\_\_ 18. Attempt to add the **Admin\_Security** integration server to `IIBNODE_WITHQM` again. Refer to step 16 if necessary.  
This time, the command is successful.
- \_\_\_ 19. Attempt to deploy the `Admin_Security_Test2.bar` file that is in the `C:\labfiles\Lab06-QSec\resources` directory to the **Admin\_Security** integration server.  
The deployment fails because the group `Security_Test` does not have sufficient authority to deploy to the new integration server.  
The process of creating the integration server automatically creates the queue **SYSTEM.BROKER.AUTH.Admin\_Security**, but it does not automatically create the required authorities.



MQ Explorer - Content

Queues

Filter: Standard for Queues

| Queue name                        | Queue type | Open input count | Open output count |
|-----------------------------------|------------|------------------|-------------------|
| SYSTEM.BROKER.AGGR.REQUEST        | Local      | 0                | 0                 |
| SYSTEM.BROKER.AGGR.TIMEOUT        | Local      | 0                | 0                 |
| SYSTEM.BROKER.AGGR.UNKNOWN        | Local      | 0                | 0                 |
| SYSTEM.BROKER.AUTH                | Local      | 0                | 0                 |
| SYSTEM.BROKER.AUTH.Admin_Security | Alias      |                  |                   |
| SYSTEM.BROKER.AUTH.server1        | Alias      |                  |                   |
| SYSTEM.BROKER.CD.MODEL            | Model      |                  |                   |

The deployment operation requires that you add Put (Write) authority on the **SYSTEM.BROKER.AUTH.Admin\_Security** queue for the *Security\_Test* group.

## Exercise clean up

- \_\_\_ 1. In the IBM Integration API Exerciser, disconnect from the integration node by right-clicking **IIBNODE\_WITHQM** and then clicking **Disconnect**.
- \_\_\_ 2. Close the IBM Integration API Exerciser.
- \_\_\_ 3. From the IBM Integration web interface:
  - \_\_\_ a. Delete the integration server **Admin\_Security**.
  - \_\_\_ b. Stop all message flow on the **server1** integration server.
  - \_\_\_ c. Delete all flows from the **server1** integration server.
- \_\_\_ 4. Disable administrative security on IIBNODE\_WITHQM from a command in the IBM Integration Console.
  - \_\_\_ a. Stop the integration node and verify that it is stopped. Type:
 

```
mqsistop IIBNODE_WITHQM
```
  - \_\_\_ b. Disable administration security. Type:
 

```
mqsichangeauthmode IIBNODE_WITHQM -s inactive -m mq
```
  - \_\_\_ c. Restart the integration node. Type:
 

```
mqsistart IIBNODE_WITHQM
```

## End of exercise

## Exercise review and wrap-up

In the first part of this exercise, you set up user and group authorities for a non-privileged Windows user (No\_Auth\_ID).

In the second part of the exercise, you added basic authorizations for a non-administrative user.

In the third part of the exercise, you activated Integration Bus administration security. The same non-privileged user ID was used to demonstrate that the user cannot use any functions in the Integration Bus environment until authorization was configured on the IBM MQ authorization queues.

Having completed this exercise, you should be able to:

- Activate queue-based administration authority
- Use the IBM Integration Bus security queues on IBM MQ to assign permissions for integration nodes, integration servers, and message flows





# Exercise 7. Implementing web services and web services security

## What this exercise is about

In this exercise, you configure the integration node runtime environment to support web services. You use the supplied tools to set up a web service that can accept information from, or deliver it to, a transport with secure HTTP (HTTPS).

## What you should be able to do

After completing this exercise, you should be able to:

- Configure the integration node HTTP listener
- Implement message flows that provide and use web services
- Implement a web service message flow application that uses HTTPS for secure transport

## Introduction

A web service is a callable program that is at a web location, and can create unlimited extensions to the classic remote procedure call (RPC) paradigm. As standards stabilize, web services of all types are becoming common.

In the first part of this exercise, you configure an integration node HTTP listener.

In the second part of this exercise, you review the web services messages flow and build and deploy a BAR file.

In the third part of this exercise, you build the BAR file and then use the Integration Toolkit BAR file editor to modify the configurable properties.

In the fourth part of this exercise, you test the web service.

In the fifth part of this exercise, you implement SSL for the web service application.

## Required materials

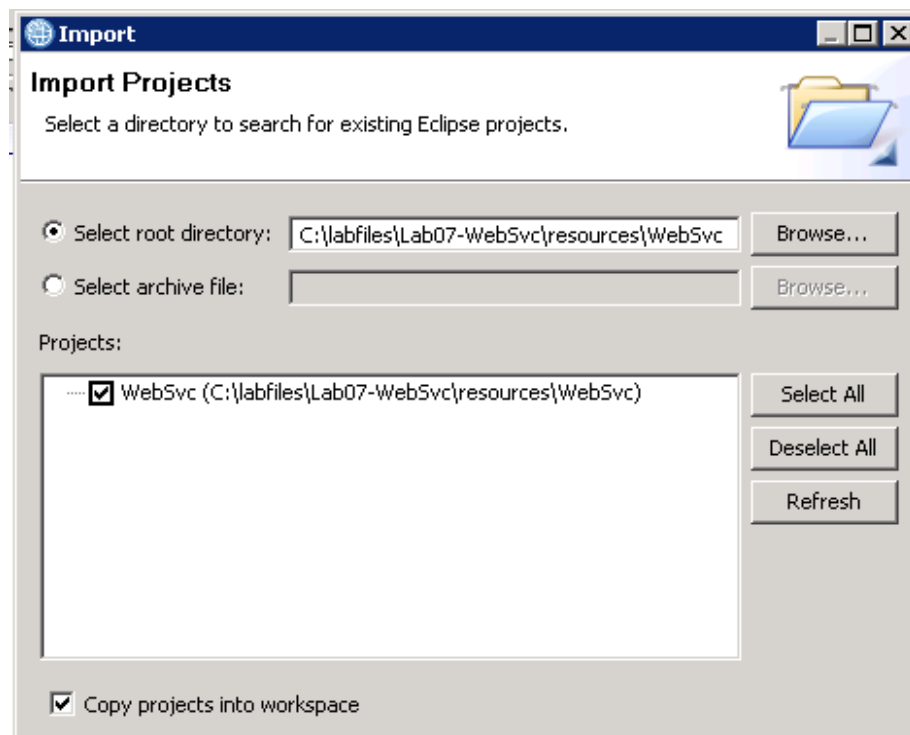
- IBM Integration Bus V10 and IBM MQ V8
- The IIBNODE\_WITHQM integration node and the IIBQM queue manager that were created in Exercise 2

- Lab files in the C:\labfiles\Lab07-WebSvc directory

## Exercise instructions

### Exercise set up

- \_\_\_ 1. If it is not already open, open IBM Integration Toolkit.
- \_\_\_ 2. To prevent any conflicts with any existing message flow applications, switch to a new workspace that is named `C:\Workspace\WebSvc`.
  - \_\_\_ a. In the Integration Toolkit, click **File > Switch Workspace > Other**.
  - \_\_\_ b. For the **Workspace**, type: `C:\Workspace\WebSvc`
  - \_\_\_ c. Click **OK**. After a brief pause, the IBM Integration Toolkit restarts in the new workspace.
  - \_\_\_ d. Close the Welcome window to go to the **Integration Development** perspective.
- \_\_\_ 3. Import an existing project that contains the message flows for this exercise.
  - \_\_\_ a. From the menu bar, click **File > Import**. The **Import** selection dialog box is displayed.
  - \_\_\_ b. Expand **General**.
  - \_\_\_ c. Select **Existing Projects into Workspace**, and then click **Next**.
  - \_\_\_ d. To the right of **Select root directory**, click **Browse**, browse to the `C:\labfiles\Lab07-WebSvc\resources\WebSvc` folder, and then click **OK**.
  - \_\_\_ e. Ensure that `WebSvc` is selected in the Projects window.
  - \_\_\_ f. Select the **Copy projects into workspace** option.



- \_\_\_ g. Click **Finish**. The **WebSvc** project is imported into the **Independent Resources** folder in the **Application Development** view.

- \_\_\_ 4. In the IBM Integration Toolkit **Integration Nodes** view, verify that the integration node IIBNODE\_WITHQM is running.
- \_\_\_ 5. Create the local queues (PING\_IN, PING\_OUT, and FAILURE) that are required for this exercise on the IIBQM queue manager.
  - \_\_\_ a. In an IBM Integration Console, type the following command:  

```
runmqsc IIBQM < C:\labfiles\Lab07-WebSvc\Q_Defs.mqsc
```
  - \_\_\_ b. Open IBM MQ Explorer and verify that the local queues were created on the IIBQM queue manager.

## **Part 1: Configure the integration node HTTP listener**

In this part of the exercise, you configure the integration node HTTP listener.

The integration node HTTP listener requires that you use an integration node that is configured with a default queue manager. In this exercise, you use the integration node IIBNODE\_WITHQM that you created in Exercise 2. It is defined to use the IIBQM queue manager as its default queue manager.

- \_\_\_ 1. In this exercise, you use port 7085 for the integration node HTTP listener. Verify that port 7085 is not already in use by another application.
  - \_\_\_ a. From a Windows command prompt, type the following command to get the network information and save it to a text file. Type:  

```
netstat -an > C:\labfiles\network.txt
```
  - \_\_\_ b. Open the text file in Notepad and examine the list of assigned and active port numbers. Verify that port number 7085 is not already in use.
- \_\_\_ 2. Configure the integration node HTTP listener to use port 7085.
  - \_\_\_ a. Check the current port for the HTTP listener on the integration node. In the IBM Integration Console, type:  

```
mqsireportproperties IIBNODE_WITHQM -b httplistener -o HTTPConnector -a
```

The results should show that the current port is 7800.
  - \_\_\_ b. Change the port for the integration node listener to 7085. In the IBM Integration Console, type:  

```
mqsichangeproperties IIBNODE_WITHQM -b httplistener -o HTTPConnector  
-n port -v 7085
```
  - \_\_\_ c. Verify that the HTTPConnector port is now 7085. In the IBM Integration Console, type:  

```
mqsireportproperties IIBNODE_WITHQM -b httplistener -o HTTPConnector -a
```

The results should show that the current port is now 7085.
- \_\_\_ 3. For the changes to take effect, the integration node must be stopped and restarted after entering the `mqsichangeproperties` command.
  - \_\_\_ a. Stop the integration node. In the Integration Console, type:  

```
mqsisstop IIBNODE_WITHQM
```

After a few moments, a confirmation message is displayed.

- \_\_ b. Restart the integration node. In the Integration Console, type:

```
mqsistart IIBNODE_WITHQM
```

After a few moments, a confirmation message is displayed.



### Note

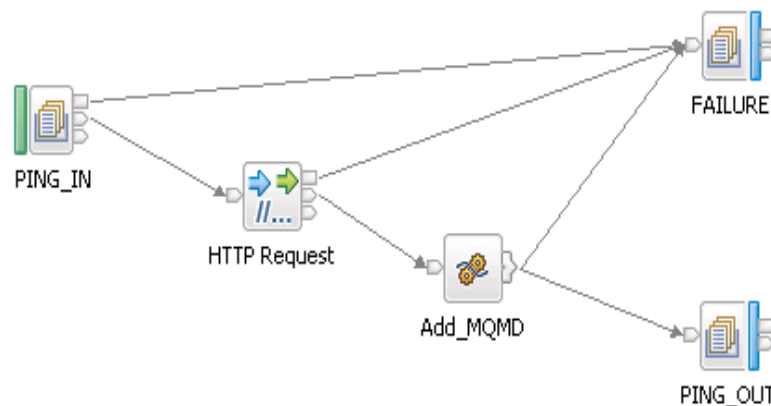
The `mqsistop` command waits for a signal that the integration node was shut down successfully before it displays the confirmation message.

The `mqsistart` command returns a confirmation message when you enter start commands, but it does not wait for the integration node and its components to signal that they are ready. As a result, if an error occurs during start, it is possible for the `mqsistart` confirmation message to show even though the components did not start.

Always check the status of the integration node and its components to ensure that they started correctly.

## Part 2: Review the message flows

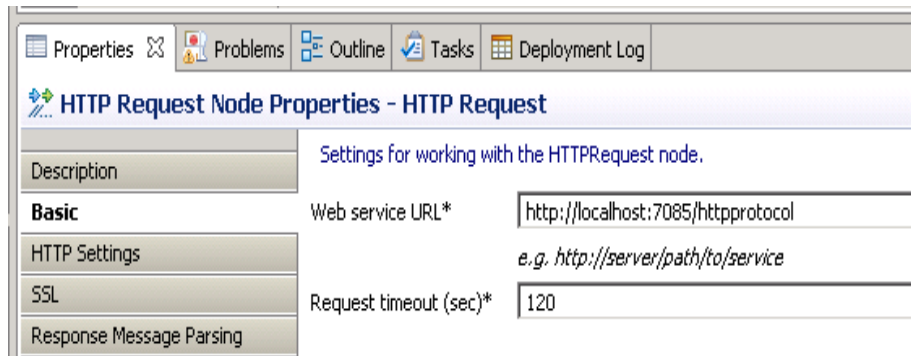
- \_\_ 1. In the IBM Integration Toolkit, review the **CallWebSvc** message flow.
- \_\_ a. In the Application Development view, expand **Independent Resources > WebSvc > Flows**.
- \_\_ b. Double-click **CallWebSvc.msgflow** to open it in the Message Flow editor.



- \_\_ 2. Notice the names of the MQInput node (PING\_IN) and MQOutput nodes (FAILURE and PING\_OUT). The node names are the same as the queue names that are specified as properties and should match the queues that you created in the **Exercise set up**.
- If you want to confirm these queue names, right-click those nodes and click **Properties**.
- \_\_ 3. Click the **HTTP Request** node to view its properties in the **Properties** view.
- In the **Basic** properties view, view the entry for **Web service URL**. The format of the URL is:

```
http://server:inode_listener_port/service
```

The value of `service` must agree with a similar value that is specified in the properties of the HTTPInput node in the *target* web service message flow.



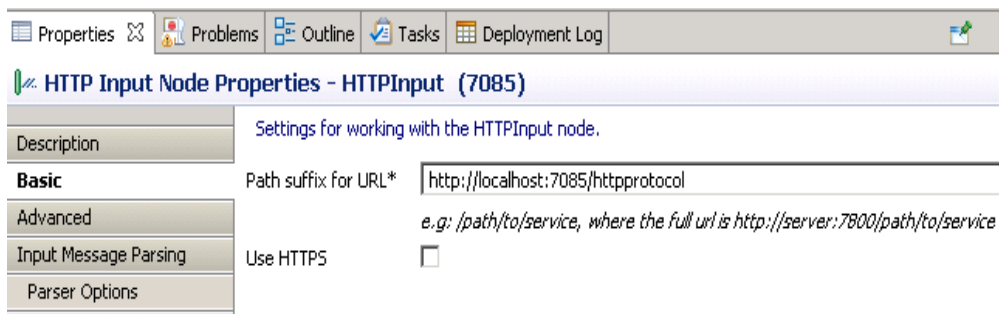
Do not change any of the properties, but do remember them.

- \_\_\_ 4. Open the **HTTPProtocol** message flow in the Message Flow editor by double-clicking **HTTPProtocol.msgflow** in the Application Development view. It should contain three nodes as shown in the following figure.



- \_\_\_ 5. Click the **HTTPInput (7085)** node to see its properties.

The **Path suffix for URL** value on the Basic tab exactly matches the URL on the previous flow (the HTTPRequest node in that flow is starting this flow by using that URL).

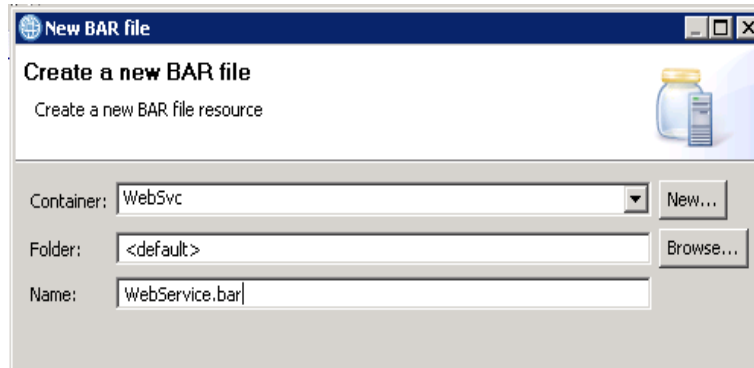


- \_\_\_ 6. Close both of the message flows.

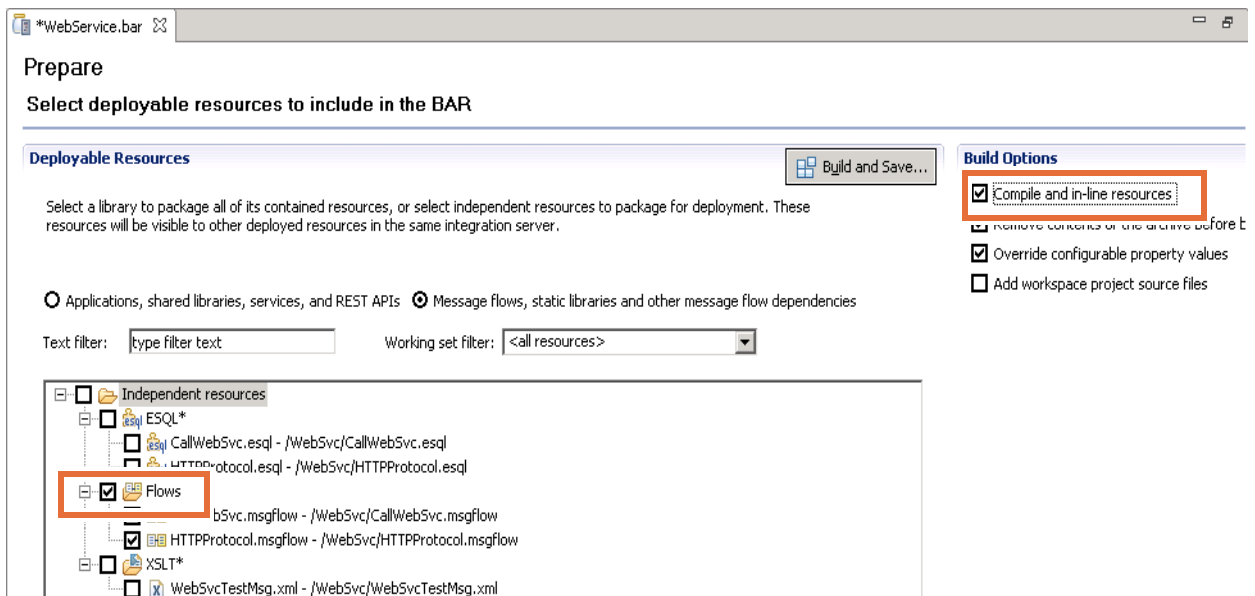
## Part 3: Create and deploy a BAR file

In this part of the exercise, you create and deploy a BAR file that contains the web services message flows.

- \_\_\_ 1. In the IBM Integration Toolkit, create a BAR file that is called `WebService.bar`.
  - \_\_\_ a. From the menu bar, click **File > New > BAR file**.
  - \_\_\_ b. For **Container**, select: **WebSvc**.
  - \_\_\_ c. For **Folder**, use the `<default>` value.
  - \_\_\_ d. For the **Name**, type: `WebService.bar`

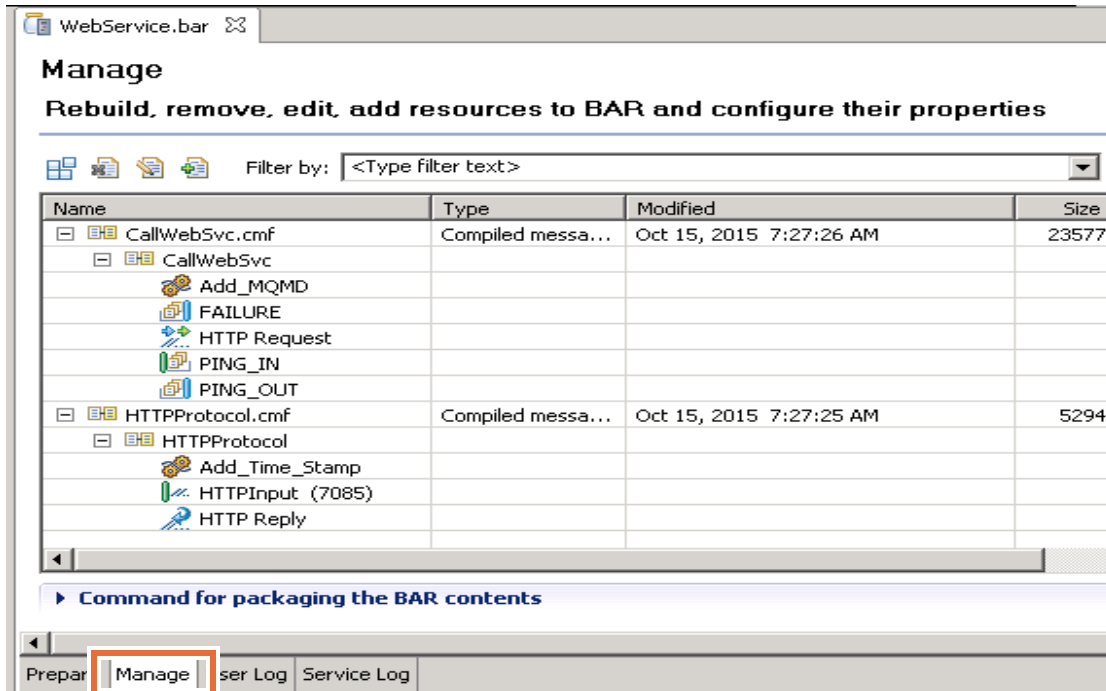


- \_\_\_ e. Click **Finish**. The new BAR file opens in the BAR File editor.
- \_\_\_ f. Select the **Flows** check box to add both message flows.
- \_\_\_ g. Under the **Build Options**, select **Compile and in-line resources**.



- \_\_\_ h. Click **Build and Save**.
- \_\_\_ i. Click **OK** on the success message. The message flows are added to the BAR file.

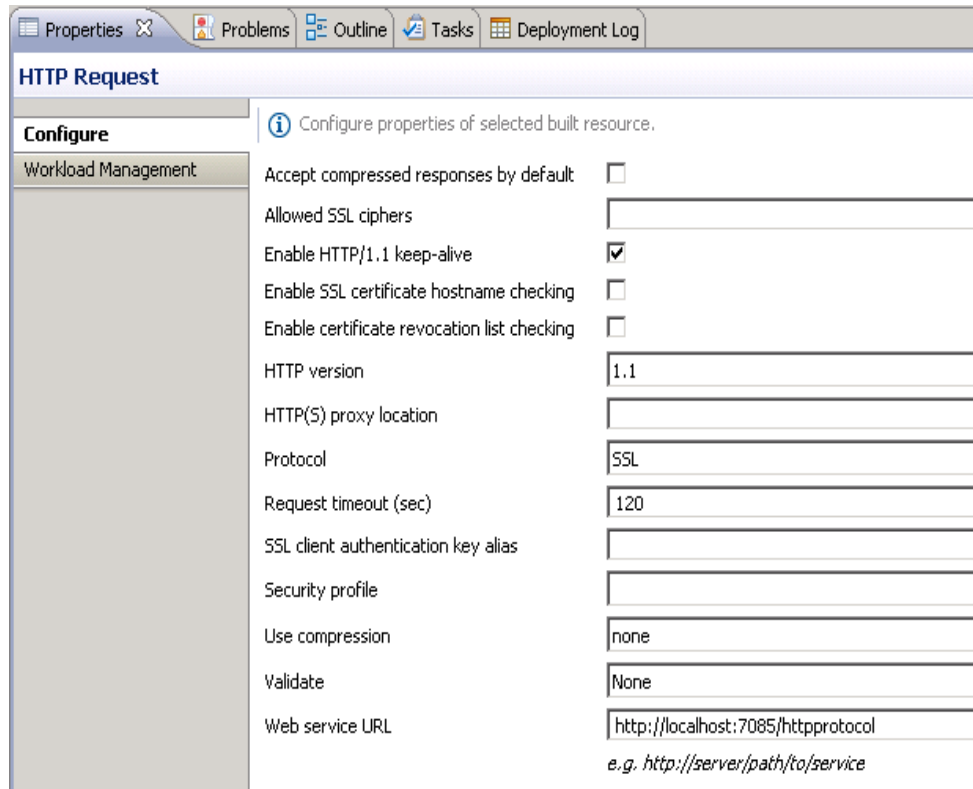
- \_\_\_ 2. You can view and modify the message flow configurable properties, such as the queue names, by editing the BAR on the BAR File editor Manage tab.
- \_\_\_ a. In the BAR File editor, click the **Manage** tab (at the bottom of the BAR File editor window).
- \_\_\_ b. Fully expand the two compiled message flows ( .cmf files).



- \_\_\_ c. Click the **HTTPRequest** and the **HTTPInput (7085)** nodes in the tree to view their configurable properties.



The properties for the HTTPRequest node are shown here.



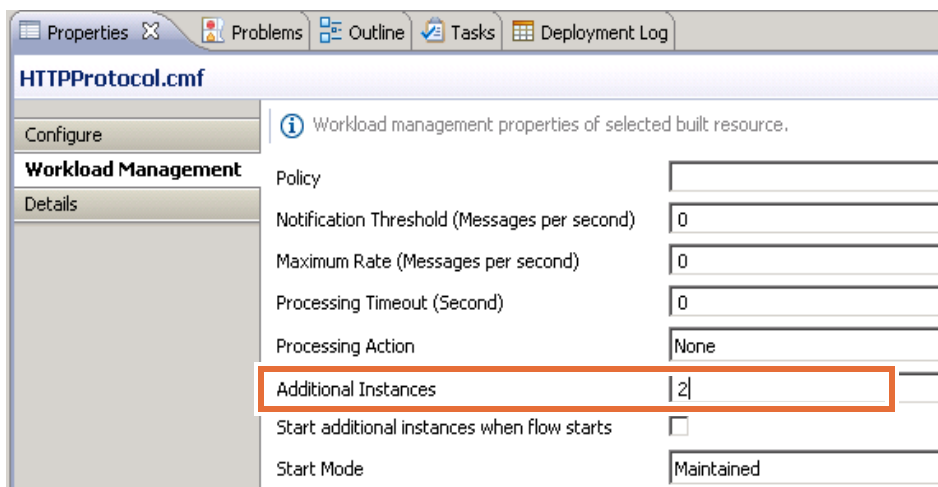
Before you deploy the BAR file, you can change the URL for the target web service, which overrides the value that was used for development testing. Similarly, a developer can test locally by simulating a remote web service.

To start the actual target web service, no changes need to be made to the calling message flow. You need only to specify a different URL by using the BAR File editor **Manage** tab and then redeploy the BAR file.

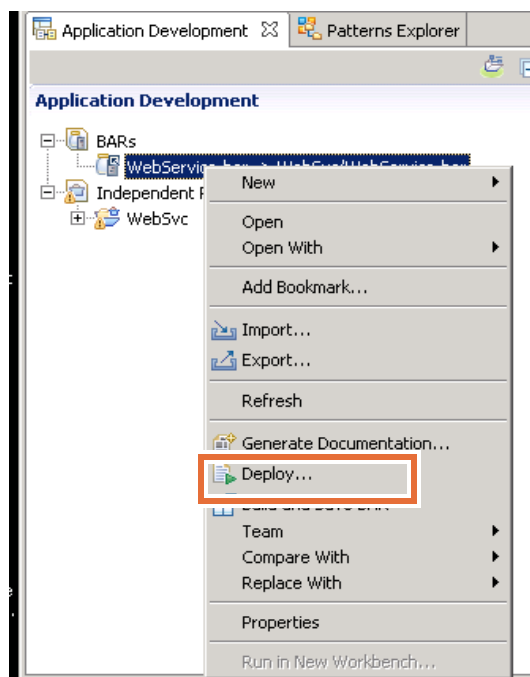
In this exercise, you use the provided values for the connections. Do not change the configurable properties for message flow nodes.

- \_\_\_ 3. For this exercise, you allow for more than one instance of the flow to run.
  - \_\_\_ a. While still on the BAR File editor **Manage** tab, click `HTTPProtocol.cmf` to display the message flow configurable properties.
  - \_\_\_ b. In the **Properties** view, click **Workload Management** to display the **Workload Management** configurable properties.

- \_\_\_ c. Change the **Additional Instances** property value to **2**.



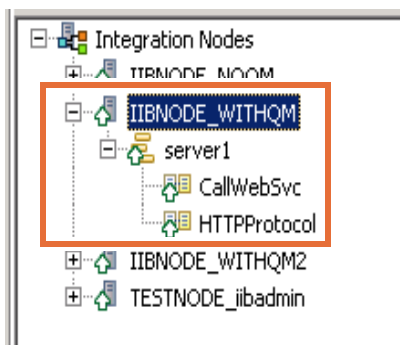
- \_\_\_ d. Similarly, change the **Additional Instances** property for the `CallWebSvc.cmf` message flow to **2**.
- \_\_\_ e. Save the BAR file updates by typing **Ctrl+S**, or by clicking **File > Save** from the menu bar.
- \_\_\_ f. Close the BAR file editor.
- \_\_\_ 4. Deploy the BAR to the **server1** integration server on **IIBNODE\_WITHQM**.
- \_\_\_ a. In the **Application Development** view, right-click the `WebService.bar` file and then click **Deploy** from the menu.



- \_\_\_ b. In the **Deploy** window, click **server1** (under **IIBNODE\_WITHQM**), and then click **Finish**.

The message flows should show under the **server1** integration server in the **Integration Nodes** view.

- \_\_\_ c. Verify that the deployment was successful by reviewing the **Deployment Log** and the **Integration Nodes** view.



- \_\_\_ 5. Deploying the message flows should activate the HTTP port 7085. Verify the port 7085 is in a LISTENING state.
- \_\_\_ a. From a Windows command prompt, type:
- ```
netstat -an > C:\labfiles\network.txt
```
- \_\_\_ b. Examine this list in Notepad, the listener that uses port 7085 should now be displayed as LISTENING.

network.txt - Notepad

File Edit Format View Help

Active Connections

Proto	Local Address	Foreign Address	State
TCP	0.0.0.0:88	0.0.0.0:0	LISTENING
TCP	0.0.0.0:135	0.0.0.0:0	LISTENING
TCP	0.0.0.0:389	0.0.0.0:0	LISTENING
TCP	0.0.0.0:445	0.0.0.0:0	LISTENING
TCP	0.0.0.0:464	0.0.0.0:0	LISTENING
TCP	0.0.0.0:523	0.0.0.0:0	LISTENING
TCP	0.0.0.0:593	0.0.0.0:0	LISTENING
TCP	0.0.0.0:636	0.0.0.0:0	LISTENING
TCP	0.0.0.0:2414	0.0.0.0:0	LISTENING
TCP	0.0.0.0:3268	0.0.0.0:0	LISTENING
TCP	0.0.0.0:3269	0.0.0.0:0	LISTENING
TCP	0.0.0.0:3389	0.0.0.0:0	LISTENING
TCP	0.0.0.0:4414	0.0.0.0:0	LISTENING
TCP	0.0.0.0:4416	0.0.0.0:0	LISTENING
TCP	0.0.0.0:5722	0.0.0.0:0	LISTENING
TCP	0.0.0.0:5800	0.0.0.0:0	LISTENING
TCP	0.0.0.0:5900	0.0.0.0:0	LISTENING
TCP	0.0.0.0:7085	0.0.0.0:0	LISTENING
TCP	0.0.0.0:9389	0.0.0.0:0	LISTENING
TCP	0.0.0.0:47001	0.0.0.0:0	LISTENING

The listener for port 7085 is now active because you deployed a web service message flow that contains an HTTPInput node. The integration node compared the HTTP port number (7085) with the port that is assigned to the integration node. Because they matched, the integration node listener became active.

**Note**

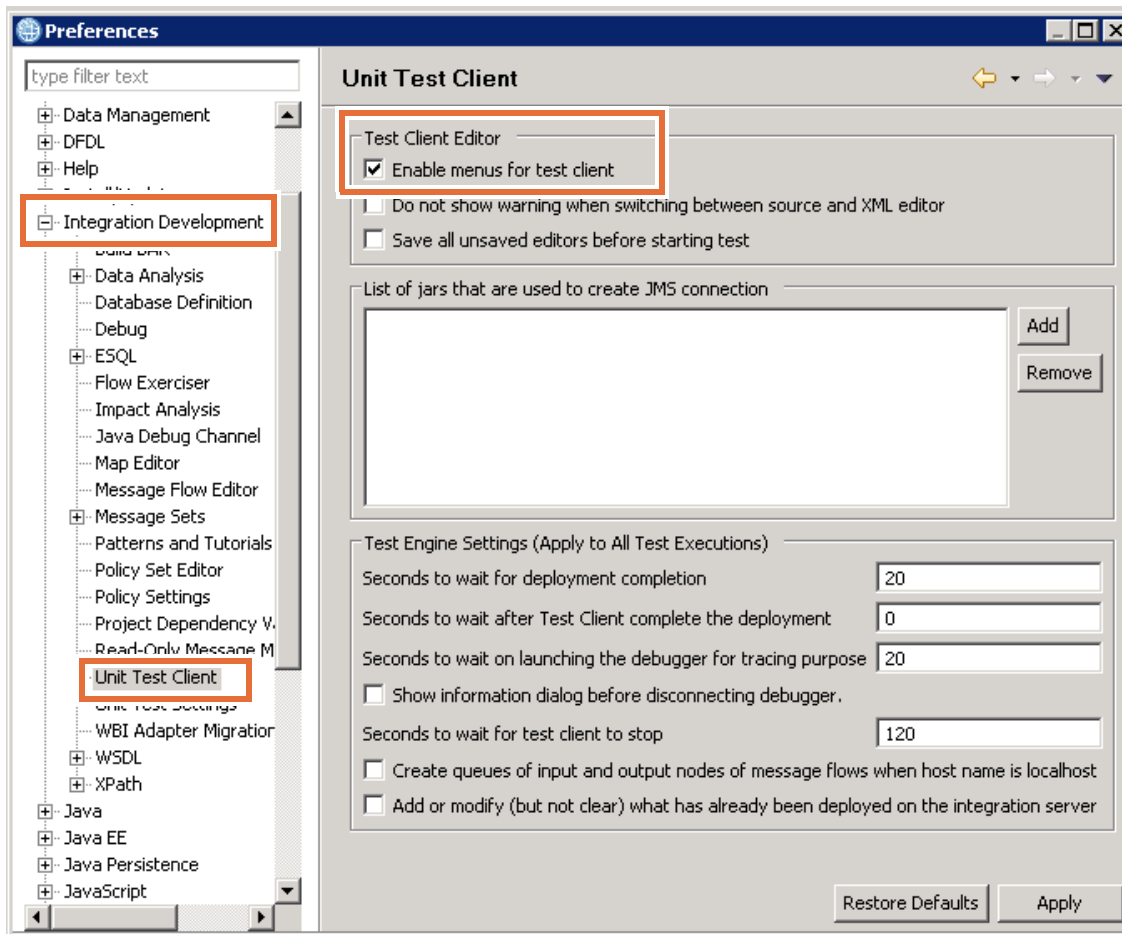
The `mqsiccreatebroker` and `mqsicchangebroker` commands can also be used to assign or change the integration node listener port number by using the `-P HTTPPort` parameter. Be sure to use an uppercase “P” when entering this command as lowercase “p” is also a valid parameter.

Later in this exercise, you learn how to configure SSL, which uses a different port.

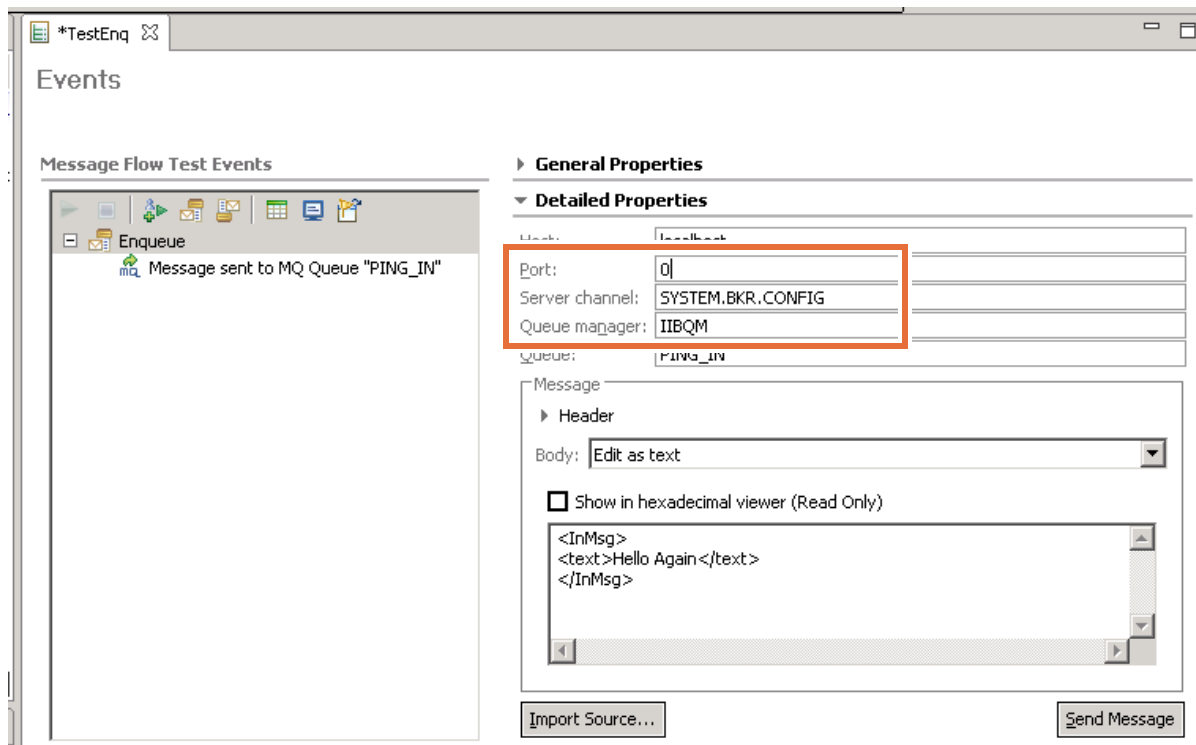
## Part 4: Test the web service application

In this part of the exercise, you use the Integration Toolkit Unit Test Client and a pre-defined test to test the web service.

- \_\_\_ 1. Enable the Integration Toolkit Unit Test Client.
  - \_\_\_ a. In the Integration Toolkit, click **Window > Preferences**.
  - \_\_\_ b. Expand **Integration Development**.
  - \_\_\_ c. Click **Unit Test Client**.
  - \_\_\_ d. Select **Enable menus for test client** and then click **OK**.



- \_\_\_ 2. In the **Application Development** view of the IBM Integration Toolkit, expand **Independent Resources > WebSvc > Flow Tests**.
- \_\_\_ 3. Double-click the file `TestEnq.mbttest` to open it in the Test Client.
- \_\_\_ 4. Under **Detailed Properties**, change the properties in the Test Client as follows:
  - \_\_\_ a. Change the **Queue manager** to match your integration node queue manager name (**IIBQM**).
  - \_\_\_ b. Change the Port to **0** (this port is not the HTTP port that you set earlier in the exercise by using the `mqsischangeproperties` command).



- \_\_\_ c. Type **Ctrl+S** to save the file.
- \_\_\_ 5. In the Unit Test Client, click **Send Message** to put a message on the PING\_IN queue.  
The message flow starts, calls the `WebService` flow, obtains the response, adds a message descriptor, and delivers it to the output queue called PING\_OUT.



### Note

The only result that you see is *Message Sent to Queue*. You do not see the response message until you complete the next step.

- \_\_\_ 6. Use the IBM MQ Explorer to see whether the message is on the output queue **PING\_OUT**.  
You can right-click the queue and click **Browse Messages** to view the message on the queue.

The message data should contain the following text:

```
<OutMsg>
<NewText>The date/time is:</NewText>
<DateTime>date_and_timestamp when the flow was run</DateTime>
</OutMsg>
```

- \_\_\_ 7. Close the Unit Test Client in the IBM Integration Toolkit. Do not save the changes when prompted.

## ***Part 5: Implement SSL for the web service application***

It is possible that a developer tested in an unsecured environment so that when the application is moved to the production environment you must take steps to secure it. HTTP can be run over SSL to ensure that the data is not altered while it is in transit.

Imagine, a scenario where a flow that calls a web service is being allowed to flow across non-SSL HTTP connections. However, other users might start that same web service by using an HTML form in a browser. In that case, you want to allow only trusted access. There is no requirement to change the message flows. You set up a new archive file to make necessary changes to one message flow to allow this scenario to work.

First, you must set up SSL for use. In a real-world environment, it is likely that you would obtain certificates from an external certificate authority. However, for the purposes of testing or internal only use, the **keytool** utility is included with the IBM Integration Bus. With the **keytool** utility, you can generate certificate keys to use with SSL for testing. You use it for this exercise.

- \_\_\_ 1. Open an Integration Console as an Administrator user, if one is not already open.
- \_\_\_ 2. Change directories to the **keytool** utility directory:
- ```
cd C:\Program Files\IBM\WebSphere MQ\java\jre\bin
```
- \_\_\_ 3. Run the keytool utility without any options to display a list of all the possible parameters:

```
keytool
```

Briefly review the parameters and options.

- \_\_\_ 4. Create a keystore.
- \_\_\_ a. Type the following command in the Integration Console:
- ```
keytool -genkey -keypass devkey -keyalg RSA -alias tomcat -keystore
dev.keystore -storepass devkey -validity 360 -keysize 2048
```

- \_\_ b. You are asked a series of questions for the key generation process; respond to them as shown here. You can use any name when prompted for first and last names.

What is your first and last name?

[Unknown]: **Mary Smith**

What is the name of your organizational unit?

[Unknown]: **WM646**

What is the name of your organization?

[Unknown]:

What is the name of your City or Locality?

[Unknown]:

What is the name of your State or Province?

[Unknown]:

What is the two-letter country code for this unit?

[Unknown]: **US**

Is CN=Mary Smith, OU=WM645, O=Unknown, L=Unknown, ST=Unknown, C=US correct? (type "yes" or "no")

[no]: **yes**

The keystore is created with the values that you specified. The keystore file is written to the C:\Program Files\IBM\WebSphere MQ\java\jre\bin directory.

- \_\_ 5. Use Windows Explorer to create a directory that is named **C:\labfiles\keystores**.
- \_\_ 6. Copy the **dev.keystore** file from the **C:\Program Files\IBM\WebSphere MQ\java\jre\bin** directory to the **C:\labfiles\keystores** directory.
- \_\_ 7. Make the integration node **IIBNODE\_WITHQM** aware of the keystore file, and configure the HTTPListener to use the SSL port. This port is not the default HTTP port that you used earlier.

Type the following commands in the order shown.

*These commands are case-sensitive.*



### Hint

To reduce the amount of typing you must do, you can press the up arrow key on your keyboard in the Integration Console to retrieve a previous command. You can then edit the portion of command that changed, by using the left arrow, right arrow, Delete, and Backspace keys.

Press Enter to run the command after you finish editing it.

- \_\_ a. Enable SSL for the integration node:

```
mqsichangeproperties IIBNODE_WITHQM -b httplistener -o HTTPListener -n
enableSSLConnector -v true
```

- \_\_ b. Configure the integration node with the keystore information:

```
mqsichangeproperties IIBNODE_WITHQM -b httplistener -o HTTPSConnector -n
keystoreFile -v C:\labfiles\keystores\dev.keystore
```

- \_\_\_ c. Set the keystore password in the integration node:
- ```
mqsichangeproperties IIBNODE_WITHQM -b httplistener -o HTTPSConnector -n keystorePass -v devkey
```
- \_\_\_ d. Set the port for HTTP SSL connections:
- ```
mqsichangeproperties IIBNODE_WITHQM -b httplistener -o HTTPSConnector -n port -v 7083
```
- \_\_\_ e. Configure the integration node with the truststore information:
- ```
mqsichangeproperties IIBNODE_WITHQM -b httplistener -o HTTPSConnector -n truststoreFile -v C:\labfiles\keystores\dev.keystore
```
- \_\_\_ f. Configure the truststore password in the integration node:
- ```
mqsichangeproperties IIBNODE_WITHQM -b httplistener -o HTTPSConnector -n truststorePass -v devkey
```
- \_\_\_ g. Disable client authentication in the integration node:
- ```
mqsichangeproperties IIBNODE_WITHQM -b httplistener -o HTTPSConnector -n clientAuth -v false
```
- \_\_\_ 8. Use the `mqsireportproperties` command to verify the configuration changes that you made:
- ```
mqsireportproperties IIBNODE_WITHQM -b httplistener -o HTTPSConnector -a
```
- Verify the values that are reported for `uuid`, `keystoreFile`, `keystorePass`, `truststoreFile`, `truststorePass`, `clientAuth`, and `port`.
- ```
uuid='HTTPSConnector'  
clientAuth='false'  
keystoreFile='C:\labfiles\keystores\dev.keystore'  
keystorePass='*****'  
truststoreFile='C:\labfiles\keystores\dev.keystore'  
truststorePass='*****'  
port='7083'
```
- \_\_\_ 9. Restart the integration node from the Integration Console so that the configuration changes take effect.
- Type:
- ```
mqsistop IIBNODE_WITHQM  
mqsistart IIBNODE_WITHQM
```
- \_\_\_ 10. You are now ready to deploy message flows that use SSL. The message flows must request the correct SSL port to function correctly. For testing purposes in this exercise, you use only one of the two message flows.
- In the IBM Integration Toolkit, create a BAR file that is called `SSLArchive.bar`.
- Add only the **HTTPProtocol** message flow to the BAR file; do not add the **CallWebSvc** message flow.
- \_\_\_ a. Right-click the **BARs** folder in the **Application Development** view.



- \_\_\_ b. Click **New > BAR file**.
  - \_\_\_ c. For **Container**, select **WebSvc**.
  - \_\_\_ d. Use the **<default>** value for **Folder**.
  - \_\_\_ e. For **Name**, enter **SSLArchive.bar**
  - \_\_\_ f. Click **Finish**. The new BAR file is opened in the BAR File editor.
  - \_\_\_ g. Select the check box for **HTTPProtocol.msgflow** (do *not* include the **CallWebSvc** message flow).
  - \_\_\_ h. Click **Build and Save**. The message flow is added to the BAR file.
  - \_\_\_ i. Click **OK** to the confirmation dialog box when the build completes.
- \_\_\_ 11. Modify the message flow properties to use SSL.
- \_\_\_ a. In the **Manage** tab of the BAR File editor, expand **HTTPProtocol.msgflow**.
  - \_\_\_ b. Expand **HTTPProtocol**.
  - \_\_\_ c. Click the **HTTPInput (7085)** node. The node properties are displayed in the **Properties** view.
  - \_\_\_ d. Modify the **Path suffix for URL** to use port **7083** (SSL) instead of **7085**.
  - \_\_\_ e. Select the **Use HTTPS** check box.

**Manage**  
Rebuild, remove, edit, add resources to BAR and configure their properties

Filter by: <Type filter text>

Name	Type	Modified	Size	Path
HTTPProtocol.esql	ESQL file	Oct 15, 2015 9:41:05 AM	599	
HTTPProtocol.msgflow	Message flow	Oct 15, 2015 9:41:05 AM	2527	
HTTPProtocol				
Add_Time_Stamp				
HTTPInput (7085)				
HTTP Reply				

Command for packaging the BAR contents

Prepare Manage User Log Service Log

Properties Problems Outline Tasks Deployment Log

**HTTPInput (7085)**

Configure

Workload Management

Configure properties of selected built resource.

Path suffix for URL:   
*e.g.: /path/to/service, where the full uri is http://server:7800/path/to/service*

Decompress input message: ☐

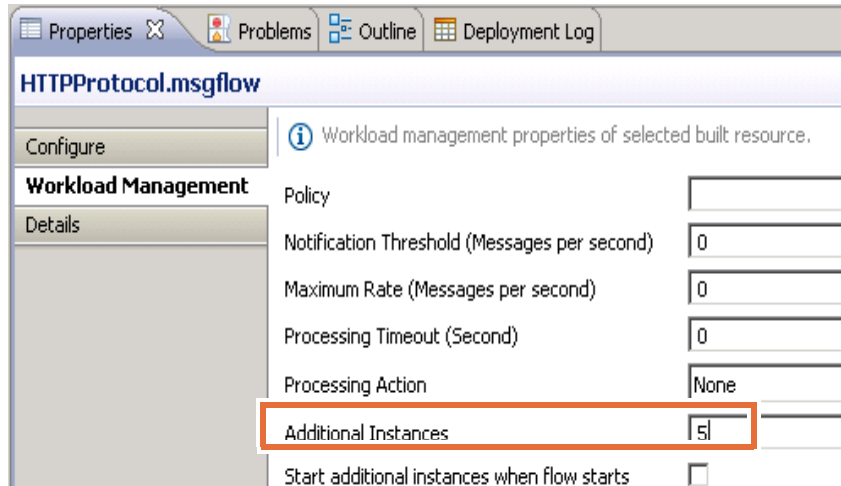
Fault format:

Security profile:

Maximum client wait time (sec):

Use HTTPS: ☒

- \_\_\_ 12. Change the number of instances to 5 to allow multiple copies of the flow to run to increase message throughput.
  - \_\_\_ a. In the **Manage** tab of the BAR File editor, select `HTTPProtocol.msgflow`.
  - \_\_\_ b. Select the **Workload Management** section on the **Properties** tab.
  - \_\_\_ c. Change **Additional Instances** to 5.



- \_\_\_ 13. Save the changes by typing Ctrl+S or clicking **File > Save**.
- \_\_\_ 14. Create an integration server that is named `SSLWebServices` on `IIBNODE_WITHQM`.
  - \_\_\_ a. In the **Integration Nodes** view of the Integration Toolkit, right-click `IIBNODE_WITHQM`.
  - \_\_\_ b. Click **New Integration Server**.
  - \_\_\_ c. For the integration server name, type: `SSLWebServices`
  - \_\_\_ d. Click **OK**.
- \_\_\_ 15. Deploy the BAR file `SSLArchive.bar` to the **SSLWebServices** integration server.



### Important

Do *not* deploy the `SSLArchive.bar` file to the same integration server where the non-SSL flow is running (the **default** integration server).

- \_\_\_ 16. Verify that deployment was successful by reviewing the **Deployment Log**.
- \_\_\_ 17. Verify that a listener for port 7083 is now active by entering the `netstat -an` command from the IBM Integration Console session.
- \_\_\_ 18. Test the web service message flow.
  - \_\_\_ a. In Windows Explorer, browse to the `C:\labfiles\Lab07-webSvc\resources\WebSvc` folder.
  - \_\_\_ b. Right-click the `webSvcform.html` file and then click **Open with > Firefox**.

- \_\_\_ c. In the browser window, click **Obtain Date/Time**.



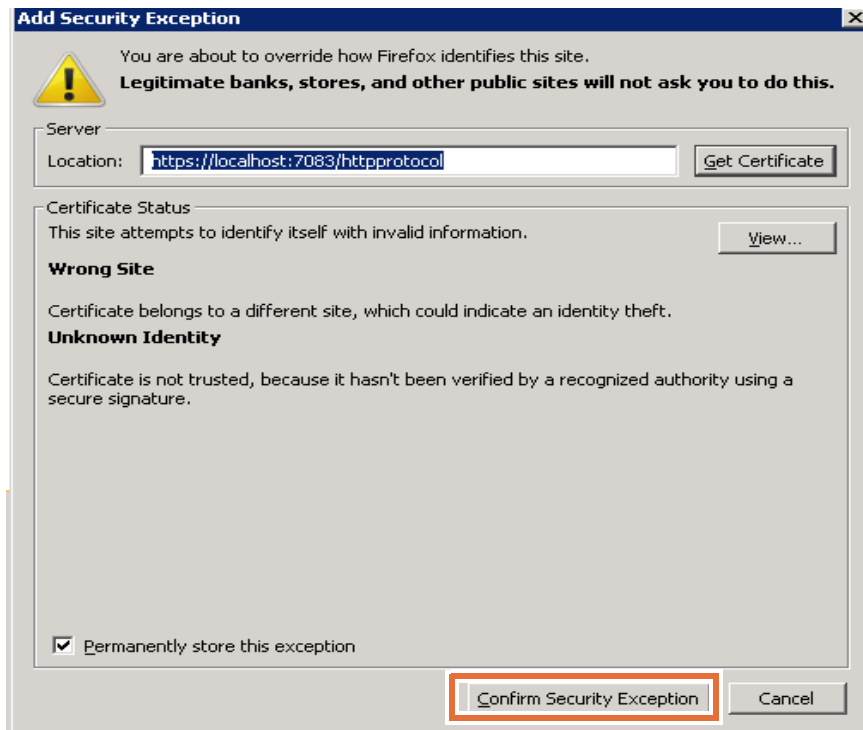
If you set up the keystore correctly, you get a security message about the certificate that is being used. It might take a few moments before the browser responds.



#### Note

If you use Internet Explorer to open the `WebSvcform.html` file, your output results are different because of the differences in the browsers.

- \_\_\_ d. Expand **I Understand the Risks** and then click **Add Exception**. The Add Security Exception window opens.



- \_\_\_ e. Click **Confirm Security Exception**. A confirmation window opens.
- \_\_\_ f. Click **Resend** on the confirmation window. You should receive the response from the web service in the form of an XML file with the date and time that is similar to the example.

```
<OutMsg>
<NewText>The date/time is: </NewText>
<DateTime>2015-10-19 05:51:04.286131</DateTime>
</OutMsg>
```

While it might not be obvious, the combination of the two flows used in this exercise approximates to a “*What time is it?*” application. The “calling” flow can be altered to call different web services by changing the URL value in the BAR file before deployment.

## Exercise clean up

- \_\_\_ 1. In the IBM Integration Toolkit **Integration Nodes** view:
  - \_\_\_ a. Right-click the **SSLWebServices** integration server and then click **Delete > All flows and resources**.
  - \_\_\_ b. Delete the **SSLWebServices** integration server.
  - \_\_\_ c. Right-click the **server1** integration server and then click **Delete > All flows and resources**.
- \_\_\_ 2. Close any open windows and programs.

## End of exercise

## Exercise review and wrap-up

As you saw, it is possible to change how a message flow that behaves as a web service can be called in different ways. You also configured an integration node to use SSL and non-SSL HTTP transports.

After completing this exercise, you should be able to:

- Configure the integration node HTTP listener
- Implement message flows that provide and use web services
- Implement a web service message flow application that uses HTTPS for secure transport



## Exercise 8. Using problem diagnosis tools

### What this exercise is about

In this exercise, you enable and disable message flow Trace nodes, and use a user trace and service trace to capture runtime information. You also configure and activate an integration server JVM Debug Port.

### What you should be able to do

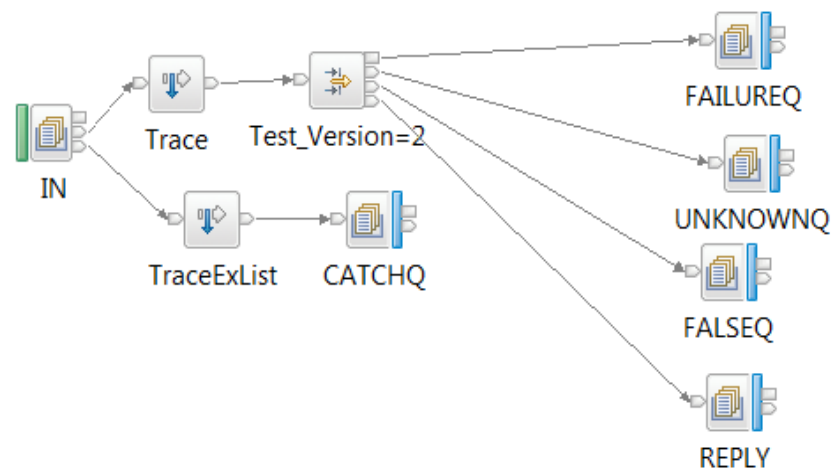
After completing this exercise, you should be able to:

- Activate a user trace and service trace to capture information
- Administer Trace nodes in a message flow
- Activate the integration server JVM Debug Port

### Introduction

In this exercise, you use various problem determination tools to administer message flows with trace nodes and administration tools to generate and control tracing.

This exercise uses a flow that is similar to the message flow that is used in a previous exercise but with the addition of a trace node.



In the first part of the exercise, you import a BAR file that contains a Trace node into IBM Integration web interface. You examine the Trace node properties and then deploy the BAR file to the IIBNODE\_WITHQM integration node.

In the second part of this exercise, you use the IBM MQ SupportPac RfhUtil to put messages on the message flow input queue and test the message flow.

In the third part of this exercise, you control Trace nodes and enable a user trace and a service trace. A service trace provides more detailed tracing, and generates a large amount of output.

In the fourth part of this exercise, you activate an integration server flow debug port.

## Requirements

- IBM Integration Bus V10 and IBM MQ Explorer V8
- The integration node IIBNODE\_WITHQM and queue manager IIBQM that were created in Exercise 2
- The lab exercise files in the C:\labfiles\Lab08-Trc folder
- The IBM MQ SupportPac IH03 (RfhUtil) in the C:\Software\ih03 folder



## Exercise instructions

### Exercise set up

Use the IBM MQ `runmqsc` command to define the queues that are required for this exercise.

\_\_\_ 1. Start an IBM Integration Console session if one is not already running.

\_\_\_ 2. Type following command:

```
runmqsc IIBQM < C:\labfiles\Lab08-Trc\resources\Q_Defs.txt
```

\_\_\_ 3. Using IBM MQ Explorer, verify that the following local queues are defined on queue manager IIBQM: CATCH, FAILURE, FALSE, IN, REPLY, and UNKNOWN.

### Part 1: *Examine and deploy the BAR file*

In this part of the exercise, you deploy an existing BAR file by using the IBM Integration web interface for the IIBNODE\_WITHQM integration node. The BAR file contains a message flow that contains a Trace node.

\_\_\_ 1. Open the IBM Integration web interface for the IIBNODE\_WITHQM integration node if it is not already open.

\_\_\_ 2. Expand **Servers**.

\_\_\_ 3. Click the down arrow to the left of the **server1** integration server and then click **Deploy**.

\_\_\_ 4. Click **Browse** and go to the `C:\labfiles\Lab08-Trc\resources` directory.

\_\_\_ 5. Click `PD.bar` and then click **Open**.

\_\_\_ 6. Examine the Trace node properties in the `PD.bar` file.

\_\_\_ a. The BAR file contains an application that is named `SIMPLEFLOW_WITH_TRACE`. In the Deploy BAR File window, expand the application until you see the message flow nodes.

\_\_\_ b. Expand the Trace node to display the Trace node configurable properties.

## Deploy BAR File

Select a BAR file to deploy. Optionally, provide an overrides file.

BAR file:

PD.bar

Browse

Deploy preview:

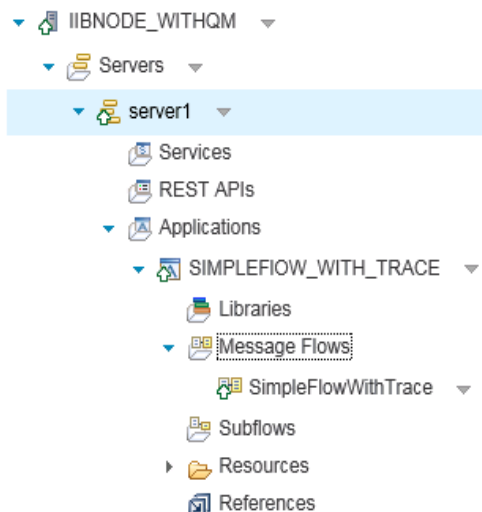
Content	Value
▶ FALSEQ	...
▶ CATCHQ	...
▶ Test_Version=2	...
▶ TraceExList	...
▶ FAILUREQ	...
▼ Trace	...
filePath	C:\labfiles\Lab08-Trc\SimpleFlowtrace.txt
▶ REPLY	...

Overrides

The developer specified that the Trace node should create a trace file. Based on this setting at design time, the only property that the administrator can change is the location of the trace file. In this BAR file, the trace file destination (**filePath**) is set to:  
**C:\labfiles\Lab08-Trc\SimpleFlowtrace.txt.**

### 7. Click **Deploy**.

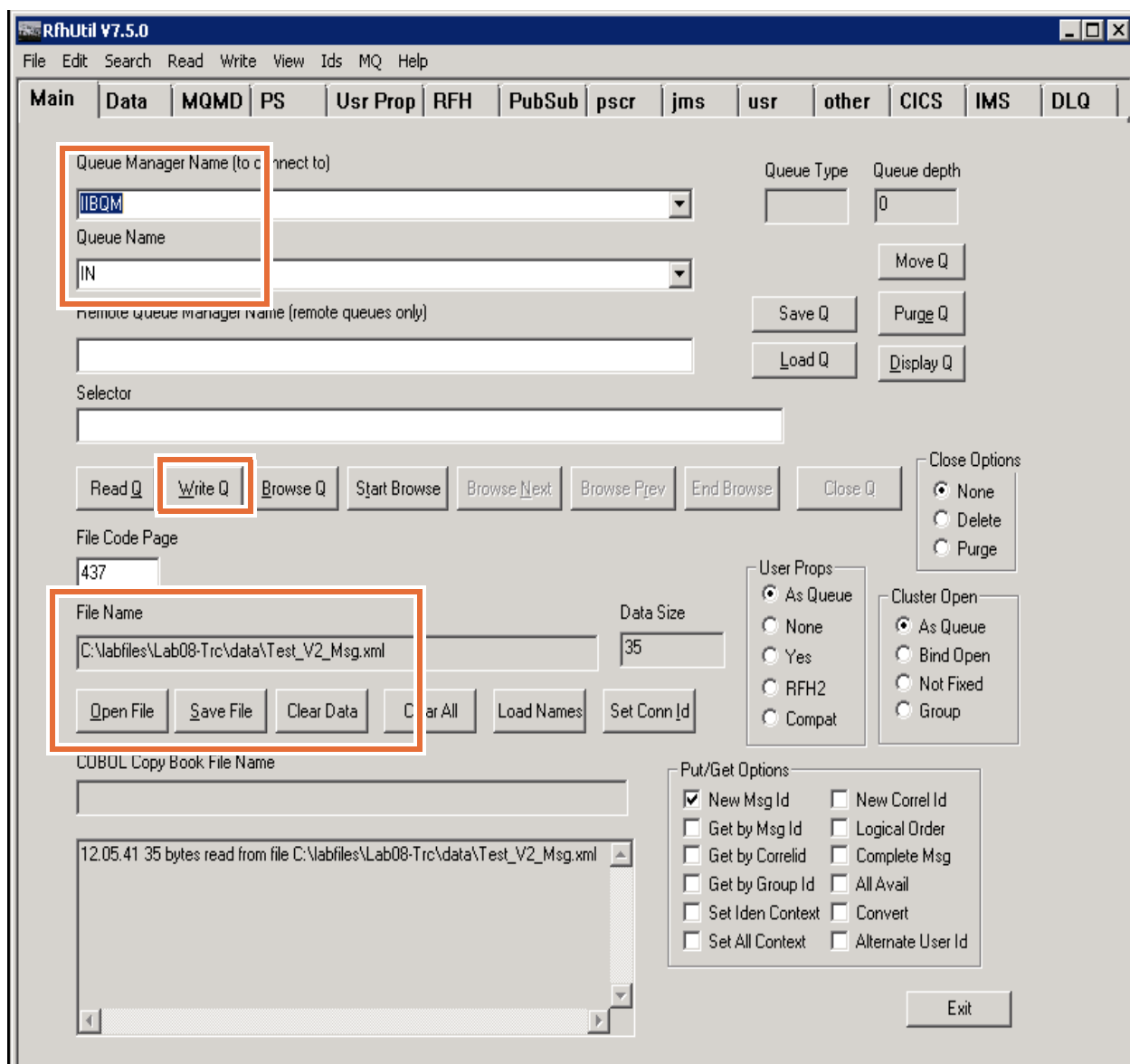
You should see a message flow that the deployment was successful. You should also see that the message flow **SimpleFlowWithTrace** is running under the **server1** integration server.



## Part 2: Test the message flow

In this part of the exercise, you test the message flow by using the IBM MQ SupportPac RfhUtil.

- \_\_\_ 1. Start RfhUtil.
  - \_\_\_ a. In Windows Explorer, browse to the C:\Software\ih03 directory.
  - \_\_\_ b. Double-click the **rfhutil.exe** file.
- \_\_\_ 2. Put a test message on the IN queue on the IIBQM queue manager.
  - \_\_\_ a. On the **Main** tab in RfuUtil, select **IIBQM** for the **Queue Manager Name**.
  - \_\_\_ b. Select **IN** for the **Queue Name**.
  - \_\_\_ c. Click **Open File**. A file selection window is displayed.
  - \_\_\_ d. Go to C:\labfiles\Lab08-Trc\data and then select the file Test\_V2\_msg.xml. Click **Open**.



- \_\_\_ e. Click **Write Q** to send the test message to the **IN** queue.
- \_\_\_ 3. Open another RfhUtil instance and check the **REPLY** queue.
  - \_\_\_ a. On the **Main** tab in RfuUtil, select **IIBQM** for the **Queue Manager Name**.
  - \_\_\_ b. Select **REPLY** for the **Queue Name**.
  - \_\_\_ c. Click **Read Q**.
  - \_\_\_ d. Switch to the **Data** tab and click **XML** as the **Data Format**. The message should show as follows:

```
<OutMsg>  
<Version>2</Version>  
</OutMsg>
```
- \_\_\_ 4. Do not close the RfhUtil session, as you use it later in this lab.

### ***Part 3: Manage traces and Trace nodes***

In this part of the exercise, you use the IBM Integration Console to activate a user trace and manage the Trace node.

- \_\_\_ 1. In the IBM Integration web interface, verify the current state of the Trace nodes on the integration server.
  - \_\_\_ a. Click **server1** to display the integration server properties in the **Overview** tab.
  - \_\_\_ b. In the **Overview** tab, expand the **Advanced Properties** section.
  - \_\_\_ c. The **Trace Node Level** property identifies the current state of the Trace nodes. Verify that the **Trace Node Level** is **on**.

**server1 - Integration Server**

Overview | Resource Statistics | Statistics

▼ Quick View

Integration Server Name	server1
Run Mode	running
UUID	4930952e-6d74-4604-8165-d1da806cfd5
Running	true
Short Description	
Long Description	
Record Mode	Disabled

▼ Advanced Properties

Injection Mode	Disabled
Process ID	6904
Trace Level	none
Soap Nodes use Embedded Listener	true
Thread Local Proxy Name Managers	false
Console Mode	off
HTTP Nodes use Embedded Listener	false
Inactive User Exit List	
Trace Node Level	on
User Trace Level	none

- \_\_\_ 2. In the IBM Integration web interface, verify the current state of the **User Trace Level** for the **SimpleFlowWithTrace** message flow that is running in the IIBNODE\_WITHQM **server1** integration server.
- \_\_\_ a. Expand the application under the **server1** integration server to show the message flow.
- \_\_\_ b. Click the **SimpleFlowWithTrace** application flow to show its properties in the **Overview** tab.

The message flow **Advanced Properties** indicate that user trace is not active (**User Trace Level** is set to none).

**SimpleFlowWithTrace - Message Flow**

Overview | Statistics | Operational Policy | Edit

▼ Quick View

Message Flow Name	SimpleFlowWithTrace
Version	
UUID	fdcc19a7-e8f0-42e0-83d3-2074d5c50e99
Service Trace Level	none
Commit Count	1
Additional Instances	0
Start Mode	Maintained
Coordinated Transaction	no
Commit Interval	0
Running	true
Run Mode	running

▼ Advanced Properties

User Trace Level	none
Active User Exit List	
Inactive User Exit List	

- \_\_\_ 3. In an IBM Integration Console, activate a debug level user trace for the **SimpleFlowWithTrace** message flow.
- Type:
- ```
mqsischangetrace IIBNODE_WITHQM -e sever1 -u -f SimpleFlowWithTrace -k  
SIMPLEFLOW_WITH_TRACE -l debug
```
- \_\_\_ 4. In the Integration web interface, verify that the **User Trace Level** for the message flow is now set to debugTrace.
- \_\_\_ 5. Send another test message to the IN queue by clicking **Write Q** in the first RfhUtil session that you started.
- \_\_\_ 6. Verify that the reply message was put on the REPLY queue on queue manager IIBQM.
- \_\_\_ 7. Retrieve the user trace data and deactivate tracing by running the GETTRACE script.

This command file runs several IBM Integration Bus commands:

- **mqsireadlog** reads the binary trace data that the trace node generated
- **mqsiformatlog** translates the binary trace data into human-readable format
- **mqsischangetrace** disables tracing (and can also be used to start tracing or change trace parameters)

The command file then starts Windows Notepad to display the trace output.

In the Integration Console, run the GETTRACE script. Enter the commands:

```
cd C:\labfiles\Tools
gettrace IIBNODE_WITHQM server1
```

- \_\_\_ 8. Review the user trace output. You should find all the runtime events (or errors) in the trace file.

The trace shows the path that each message took through the message flow, and how each node responded to the message. The trace listing also shows a detailed listing of all ESQL statements that were run in the flow.

When you are done reviewing the trace listing, close the Notepad window.

A copy of user trace file `server1.txt` is provided for your reference in the `C:\labfiles\Lab08-Trc\solution` directory.

- \_\_\_ 9. Review the output from the Trace node in the file you configured in the Trace node properties (in this case, `C:\labfiles\Lab08-Trc\SimpleFlowTrace.txt`).
- \_\_\_ a. Use Windows Explorer to locate the Trace node output file.
  - \_\_\_ b. Open the file and examine the contents. The entire message that the Trace node received is displayed in the output because the developer specified `${Root}` for the trace pattern.



#### Note

A copy of the trace file that the Trace node generates is provided for your reference in the `C:\labfiles\Lab08-Trc\solution` directory.

- \_\_\_ 10. In the Integration Console, disable the user trace on the **SimpleFlowWithTrace** message flow. Type:

```
mqsichangetrace IIBNODE_WITHQM -e sever1 -u -f SimpleFlowWithTrace -k
SIMPLEFLOW_WITH_TRACE -l none
```

- \_\_\_ 11. Trace nodes generate trace records that you can use to monitor the behavior of a message flow. After a developer finishes testing a message flow, the developer or the administrator must disable the Trace nodes to avoid the performance impact that tracing causes. In Integration Bus, you can turn off the Trace nodes without removing them from the message flow.

Disable the Trace nodes on the **server1** integration server.

In the Integration Console, type:

```
mqsichangetrace IIBNODE_WITHQM -n off -e server1
```

- \_\_\_ 12. Using the Integration web interface, verify that the **User Trace Level** on the message flow is now `none` and the **Trace Node Level** on the integration server is `off`.

- \_\_\_ 13. A service trace provides more detailed tracing, and generates a large amount of output. In this part of the exercise, you can enable and disable a service trace by using the `mqsischangetrace` command.



### Important

You typically generate a service trace under the direction of IBM Support.

- \_\_\_ a. In the Integration Console, type the following command to start a service trace for the **server1** integration server. The **-r** flag resets trace to delete any previous trace data that was collected.

```
mqsischangetrace IIBNODE_WITHQM -t -e server1 -l debug -r
```

The character before `debug` in the command is a lowercase “l”, not a “1”.

- \_\_\_ b. Type the following command to verify the trace settings on the integration server.

```
mqsireporttrace IIBNODE_WITHQM -t -e server1
```

You should see a response that indicates that the trace level is “debug”.

- \_\_\_ 14. Go to the first RFHUtil session and send another message to the IN queue.

- \_\_\_ 15. Turn off the service trace. In the Integration Console, type:

```
mqsischangetrace IIBNODE_WITHQM -t -e server1 -l none
```

- \_\_\_ 16. After you generate a service trace, you must retrieve the service trace data into a file. For this exercise, the command file `getsvctrace.cmd` in the `C:\labfiles\Tools` directory retrieves the service trace data. The command file is similar to the `gettrace.cmd` that you used when retrieving the user trace data.

- \_\_\_ a. Retrieve the service trace data from an IBM Integration Console session.

In the Integration Console, type the commands:

```
cd C:\labfiles\Tools
getsvctrace IIBNODE_WITHQM server1
```

- \_\_\_ b. Review the service trace output in the Notepad session that the `getsvctrace.cmd` opens.

The information that is captured in the service trace is more detailed. It includes trace information about the interactions of the various components (such as the toolkit and the integration nodes).

Close the Notepad window when you are done reviewing the service trace information.



### Note

A copy of the service trace file is provided for your reference in the `C:\labfiles\Lab08-Trc\solution` directory.



## Part 4: Administer the message flow debugger

Developers can use the message flow debugger in the IBM Integration Toolkit to help diagnose unexpected behavior within message flows. Breakpoints can be set to pause the flow and examine and change the contents of the message, Extended Structured Query Language (ESQL), and Java variables and mappings. While developers normally do these tasks, an administrator might be called upon to help with setting up the debugging environment.

Since the Java runtime debug environment is used for all message flow debugging, you must configure the JVM debug port for the integration server in which the message flow is running.

The debug port can be configured in the development environment by using the IBM Integration Toolkit, or in the administration environment by using IBM Integration web interface or a command. As an administrator, you must also be aware of the status of the debugger because of the potential impact on overall performance. In this exercise, you configure and manage the message flow debugger from IBM Integration web interface.



### Note

You must configure the JVM debug port once for each integration server. The integration server **QuickView** in the IBM Integration web user interface identifies whether a debug port is configured for an integration server.

- \_\_\_ 1. Determine whether a debug port is defined in the **server1** integration server on the IIBNODE\_WITHQM queue manager.
  - \_\_\_ a. In the Integration web interface for IIBNODE\_WITHQM, click **server1** to display its properties on the **Overview** tab.
  - \_\_\_ b. Expand the **Resource Manager Properties** section.
  - \_\_\_ c. Verify that the current value for the **JVM Manager - JVM Debug Port** property is 0.

**server1 - Integration Server**

Overview | Resource Statistics | Statistics

► Quick View

► Advanced Properties

▼ Resource Manager Properties

|                                     |      |
|-------------------------------------|------|
| JVM Manager - JVM Debug Port        | 0    |
| JVM Manager - JVM Min Heap Size     | -1   |
| JVM Manager - JVM Max Heap Size     | -1   |
| JVM Manager - JVM Native Stack Size | -1   |
| JVM Manager - JVM Verbose Option    | none |
| JVM Manager - JVM System Property   |      |

- \_\_\_ 2. Using the IBM Integration web interface, set the JVM debug port to 9997.
- \_\_\_ a. Click **Edit** on the **Overview** tab for the **server1** integration server.
- \_\_\_ b. Scroll down until you see the **JVM Manager - JVM Debug Port** property.

| server1 - Integration Server                 |       |     |
|--|-------|-----|
| Overview   Resource Statistics   Statistics  |       |     |
| Save Cancel                                  |       |     |
| Cache Manager - Domain Name                  |       | ... |
| Cache Manager - Clients Default to SSL       |       | ... |
| Cache Manager - SSL Protocol                 |       | ... |
| Cache Manager - SSL Alias                    |       | ... |
| Content Based Filtering - Enabled            | false | ... |
| Content Based Filtering - Validation Threads | 1     | ... |
| Content Based Filtering - Evaluation Threads | 1     | ... |
| JVM Manager - JVM Debug Port                 | 0     | ... |

- \_\_\_ c. Click the **edit** icon (the icon in the same row as the property).
- \_\_\_ d. In the **Edit Value** window, type 9997 and then click **OK**.
- \_\_\_ e. Click **Save**.



### Note

You can also use the `mqsichangeproperties` command to configure the JVM debug port:

```
mqsichangeproperties <intNodeName> -e <intServerName> -o ComIbmJVMManger
-n jvmDebugPort -v <portNumber>
```

If you use `mqsichangeproperties` to set the debug port, you must also manually stop and restart the integration server by using the `mqsistop` and `mqsistart` commands.

- \_\_\_ 3. The integration server Resource Manager Properties update to show that a debug port is enabled.

The developer can now start the debugger from the Integration Toolkit. When the developer is done debugging the message flow, you should disable the JVM debug port.

- \_\_\_ 4. Disable the integration server JVM debug port by changing the **JVM Resource Manager - JVM Debug Port** property to 0.

## Exercise Clean up

- \_\_\_ 1. Delete the deployed message flow application from the **server1** integration server on IIBNODE\_WITHQM integration node.
- \_\_\_ 2. Clear the messages from the REPLY queue on IIBQM.
- \_\_\_ 3. Close the RfhUtil windows.

## End of Exercise

## Exercise review and wrap-up

In this exercise, you used some of the troubleshooting tools available to developers and administrators, which include the Trace node, user trace, and service trace. This exercise also showed you how to use the IBM Integration web interface and command to monitor and manage the troubleshooting tools. Managing the state of the troubleshooting tools in a production environment is especially important as many of these tools negatively affect performance.

In this exercise, you worked with a local integration node. The same tools are also available for managing remote integration nodes after a connection is established to the remote integration node from the IBM Integration web interface.

Having completed this exercise, you should be able to:

- Activate a user trace and service trace to capture information
- Administer Trace nodes in a message flow
- Activate the integration server JVM Debug Port



# Exercise 9. Identifying runtime problems

## What this exercise is about

In the first part of this exercise, you analyze a series of message flows that produce runtime errors and determine where the incoming message is found after the errors occur. In the second part of the exercise, you analyze and correct error situations that involve an IBM Integration Bus integration node that connects to an IBM MQ queue manager.

## What you should be able to do

After completing this exercise, you should be able to:

- Describe the symptoms of runtime and environmental problems
- Isolate a problem to a particular component
- Determine the cause of a problem and correct it

## Introduction

In the first part of the exercise, you examine message flows that cause a runtime exception. You evaluate the runtime exceptions and identify what happens to the message when the designated processing node generates an exception.

In the second part of this exercise, you are given a preconfigured BAR file and some information about its content and what to expect in terms of what constitutes successful completion of the flow. You deploy and run the message flow. The message flow fails with one or more errors. You identify the errors and propose a correction to the problem.

## Requirements

- A lab environment with IBM Integration Bus V10 and IBM MQ V8
- Successful completion of Exercises 2
- The exercise files in the C:\labfiles\Lab09-PD and C:\labfiles\Tools directories
- The IBM MQ SupportPac IH03 (RfhUtil) in the C:\Software\ih03 directory

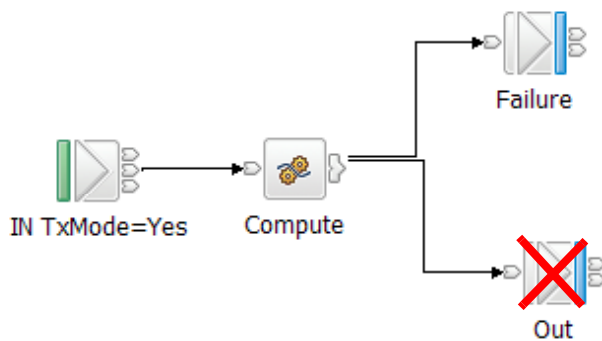
## Exercise instructions

### Part 1: Analyze message flow runtime failures

This part of the exercise presents some message flows that connect to IBM MQ. Each message flow shows a runtime exception. In each of the following scenarios, determine what happens to the message that is being processed when the designated processing node generates an exception.

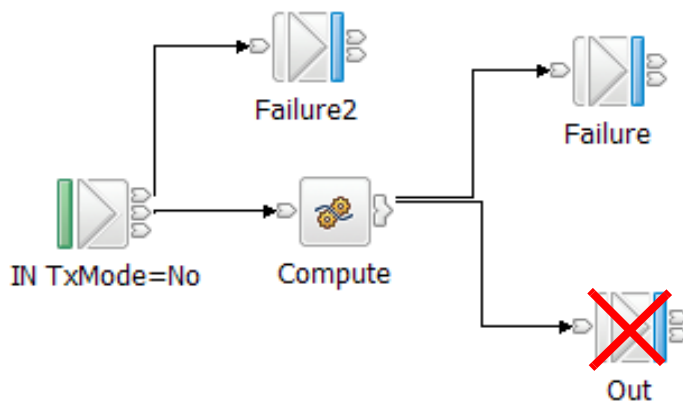
In each of the flows, **TxMode** is the state of the **Transaction mode** property for the input message processing node.

- \_\_\_ 1. In this example, the error occurs on the Out node. The queue manager has a defined dead-letter queue (**DLQ**) and a backout queue for the input queue (**IN\_BOQ**).



Record your answer here:

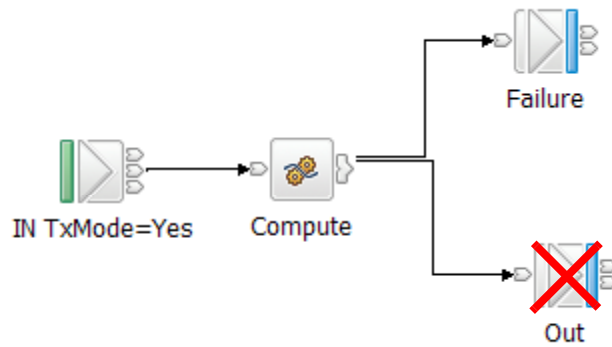
- \_\_\_ 2. Queue manager dead-letter queue = **DLQ**; IN queue backout queue = **IN\_BOQ**



Record your answer here:

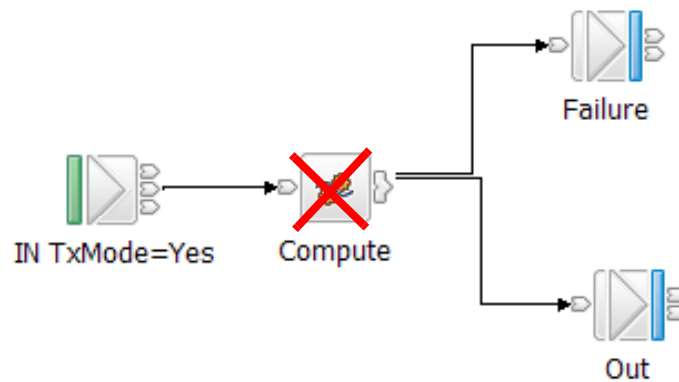
- \_\_\_ 3. Queue manager dead-letter queue = **DLQ**; IN queue has no backout queue.





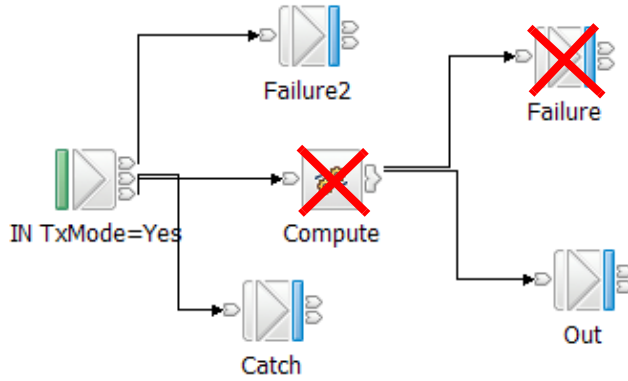
Record your answer here:

\_\_\_ 4. Queue manager dead-letter queue = **DLQ**; IN queue backout queue = **IN\_BOQ**



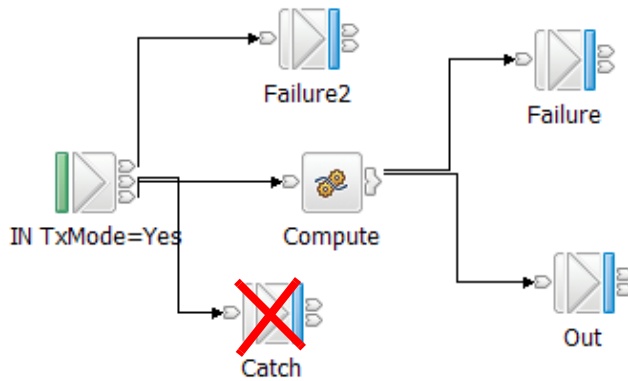
Record your answer here:

\_\_\_ 5. Queue manager dead-letter queue = **DLQ**; IN queue backout queue = **IN\_BOQ**



Record your answer here:

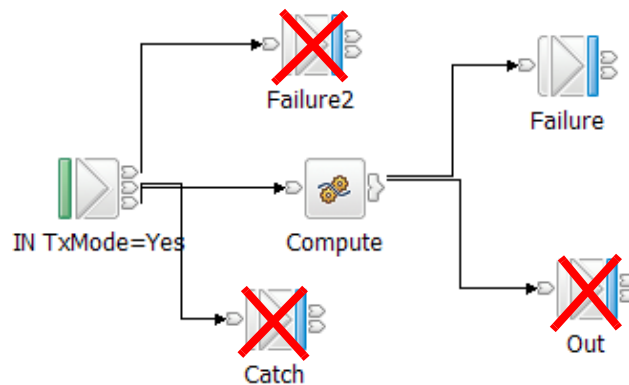
\_\_\_ 6. Queue manager dead-letter queue = **DLQ**; IN queue backout queue = **IN\_BOQ**



Record your answer here:

\_\_\_ 7. Queue manager dead-letter queue = **DLQ**; IN queue backout queue = **IN\_BOQ**

**Hint:** The **Out** node generated the original exception, but more exceptions occurred during explicit exception handling.



Record your answer here:

## Part 2: Isolate and identify problems

In this part of the exercise, you apply your accumulated knowledge to deploy and test a message flow and then identify the problems. The instructions in this exercise are intentionally brief.

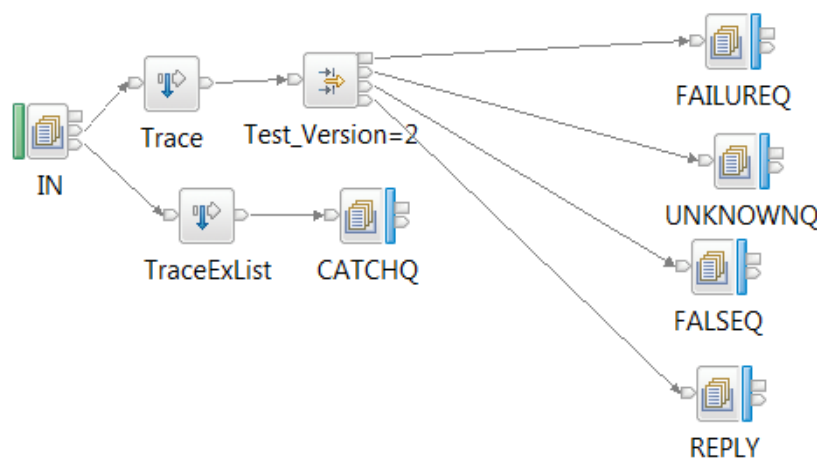
In this part of the exercise, you are given a preconfigured BAR file. You are also given some information about its content and what to expect, in terms of what constitutes successful completion of the flow.

Some general hints and tips are provided.

- What is (or are) the expected result (or results) of the BAR file content? This result is likely to be the initial criterion to determine whether all is well (or not).
- Recall that several components can participate in the IBM Integration Bus environment: the integration node, integration server, and in some case, IBM MQ queue managers, and queues. When, and if, something goes wrong, error messages are likely to be generated. The question to ask is, from which component?
- What do you know, or what can you determine, about the message flow and its use of **CATCH** or **FAILURE** logic? What do you know about the integration node queue manager in terms of a dead-letter queue? Consider these factors when you answer the question “*Where is the message?*” in the context of the default behaviors that were described in the lecture.
- Even though this course is about IBM Integration Bus, there are cases where the interaction, and dependence between the integration node and the queue manager are significant. When it comes to problem recognition and problem resolution, there is can be overlap between the two products.
- Although you have access to the IBM Integration Toolkit in this class, the focus of the exercise is to use the IBM Integration web user interface, IBM MQ Explorer, and system tools.

You can find solutions to this error scenario in Appendix A.

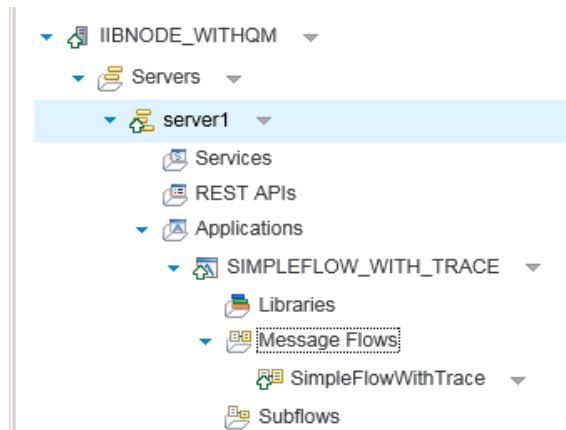
The message flow in this error scenario triggers when a message is put on the IBM MQ IN queue. The Filter node in the message flow routes the message to an output queue based on the message content.



- \_\_\_ 1. In the IBM MQ Explorer, verify that the following local queues exist on the IIBQM queue manager:
- CATCH
  - IN
  - REPLY
  - FAILURE
  - FALSE
  - UNKNOWN

If any of the queues are missing, create them as local queues on IIBQMGR.

- \_\_\_ 2. Delete any deployed message flows from the **server1** integration server on **IIBNODE\_WITHQM** integration node.
- \_\_\_ 3. Deploy `FindTheError.bar` (in the `C:\labfiles\Lab09-PD\resources` directory) to the **server1** integration server.
- \_\_\_ 4. Verify that the message flow **SimpleFlowWithTrace** message flow is deployed and running.



- \_\_\_ 5. Use RfhUtil to test the message flow.
- Use test message `Test_V2_Msg.xml` that is in the `C:\labfiles\Lab09-PD\data` directory. Write the message to queue IN in queue manager IIBQM.



#### Hint

See Exercise 8 if you need detailed instructions for the steps to test the message flow.

- \_\_\_ 6. Evaluate the results. Where was the message sent?
- Use whatever problem determination tools you think are required to understand exactly what is taking place. Apply corrective action as you see fit. If there is a problem, can you correct it more than one way?
- Consider this exercise complete when you identify why the message is not being put on the REPLY queue.

## Exercise clean up

- \_\_\_ 1. In IBM MQ Explorer, clear any remaining messages from the queues.
- \_\_\_ 2. Close any open windows and programs (such as RFHUtil). You can leave the IBM MQ Explorer open.
- \_\_\_ 3. Delete the message flow that is running on in the **server1** integration server on IIBNODE\_WITHQM.
- \_\_\_ 4. If you enabled any traces in this exercise, turn them off.

## End of Exercise

## Exercise review and wrap-up

The first part of the exercise helped you to understand integration node default behavior if an error (that is, an exception) occurs in a message flow. Message flow design and IBM MQ queue definitions and specifications also affect where the messages are ultimately found.

Part two simulated, to a degree, the nature of runtime and environmental errors that are likely to be encountered in a typical installation. Probably the most important thing to remember is the close relationship between the IBM Integration Bus and the queue manager for some message flows. Although a problem might manifest itself in one product, the contributing cause might be the other.

Having completed this exercise, you should be able to:

- Describe the symptoms of runtime and environmental problems
- Isolate a problem to a particular component
- Determine the cause of a problem and correct it





# Exercise 10. Viewing runtime statistics

## What this exercise is about

In this exercise, you enable integration node statistics collection and view the collected statistics with the IBM Integration web user interface. You also create a subscription for the integration node statistics and write them to a queue.

## What you should be able to do

After completing this exercise, you should be able to:

- Enable integration node resource and message flow statistics
- View integration node statistics in the IBM Integration web user interface
- Subscribe to integration node statistics from IBM MQ

## Introduction

In this first part of this exercise, you use the IBM Integration web interface to enable resource statistics collection on an integration server and message flow statistics collection on a message flow.

In the second part of this exercise, you use the IBM Integration web interface to examine the statistics data.

In the third part of this exercise, you use IBM MQ Explorer to subscribe to an IBM Integration Bus statistics publication. You then use the IBM MQ SupportPac RfhUtil to retrieve and view the statistics data.

## Requirements

- IBM Integration Bus V10 and IBM MQ V8
- The integration node IIBNODE\_WITHQM and the queue manager IIBQM that were created in Exercise 2
- The lab exercise files in the C:\labfiles\Lab10-Stats\resources directory
- The IBM MQ IH03 SupportPac in the C:\software\ih03 directory

## Exercise instructions

### Exercise set up

Import and run a message flow application by using the Flow Exerciser in the IBM Integration Toolkit if it is not already running.

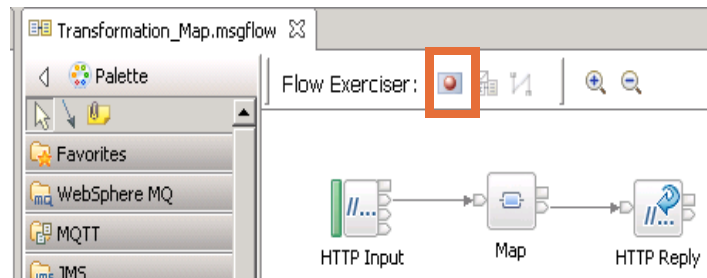
- \_\_\_ 1. Start the IBM Integration Toolkit, if it is not already running.
- \_\_\_ 2. In the IBM Integration Toolkit, switch to a new workspace.
  - \_\_\_ a. To prevent any conflicts with any existing message flow applications, switch to a new workspace that is named `C:\Workspace\Stats`.
  - \_\_\_ a. In the Integration Toolkit, click **File > Switch Workspace > Other**.
  - \_\_\_ b. For the **Workspace**, type: `C:\Workspace\Stats`
  - \_\_\_ c. Click **OK**. After a brief pause, the IBM Integration Toolkit restarts in the new workspace.
  - \_\_\_ d. Close the Welcome window to go to the **Integration Development** perspective.
- \_\_\_ 3. Import the `SimpleFlowWithMap_PI.zip` project interchange file from the `C:\labfiles\Lab10-Stats\resources` directory.
  - \_\_\_ a. From the Integration Toolkit menu bar, click **File > Import**.

The Import window is displayed.
  - \_\_\_ b. Verify that **IBM Integration > Project Interchange** is selected, and then click **Next**.
  - \_\_\_ c. To the right of **From zip file**, click **Browse**.
  - \_\_\_ d. Browse to the `C:\labfiles\Lab10-Stats\resources` directory.
  - \_\_\_ e. Click `SimpleFlowWithMap_PI.zip` and then click **Open**.
  - \_\_\_ f. Click **Transformation\_Map** (if it is not already selected) and then click **Finish**.

The Integration Toolkit imports the application and constructs the workspace.

The **Transformation\_Map** application is now displayed in the **Application Development** view.
- \_\_\_ 4. Open the `Transformation_Map.msgflow` message flow in the message flow editor.
  - \_\_\_ a. Fully expand the **Transformation\_map** application by clicking the plus (+) signs and examine its contents.
  - \_\_\_ b. Double-click the `Transformation_Map.msgflow` message flow to open it in the message flow editor.

- \_\_\_ 5. Start the Flow Exerciser and deploy the message flow to the **server1** integration server on IIBNODE\_WITHQM.
  - \_\_\_ a. In the Message Flow editor, click the **Start Flow exerciser** icon to deploy the message flow and start the Flow exerciser.

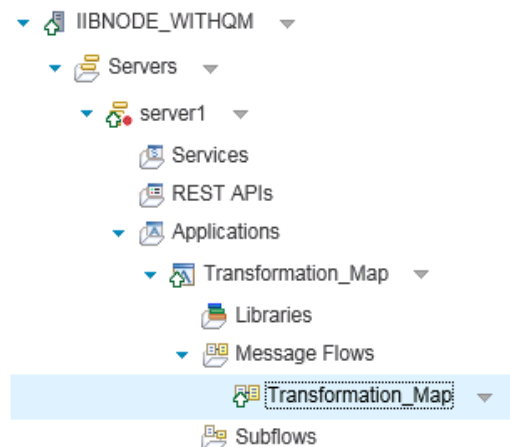


- \_\_\_ b. Select the **server1** integration server on the IIBNODE\_WITHQM integration node and then click **Finish**.
- \_\_\_ c. The application is deployed to the **server1** integration server on IIBNODE\_WITHQM. You should see a window that indicates that the message flow is being deployed.
- \_\_\_ d. Click **Close** on the Record Message window.

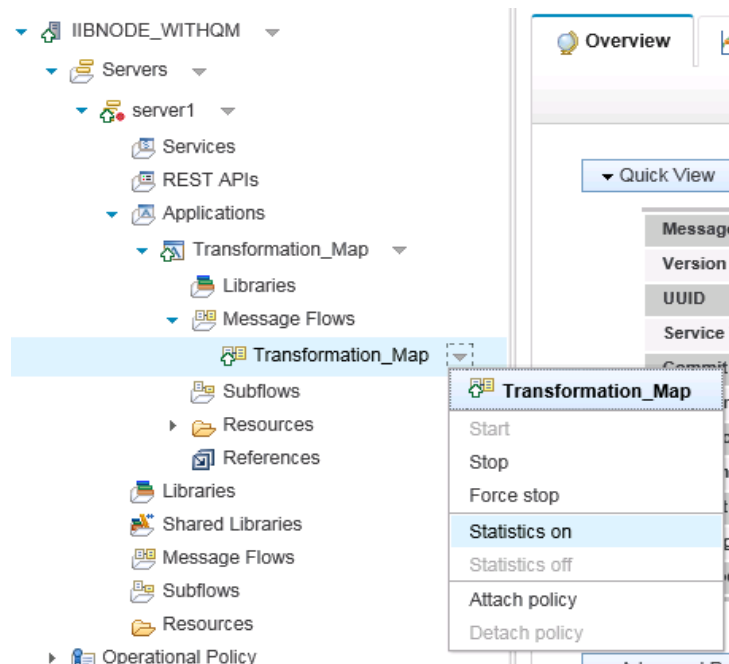
### Part 1: Enable message flow and resource statistics

In this part of the exercise, you enable *message flow statistics* and *resource statistics* from IBM web user interface for the integration node IIBNODE\_WITHQM.

- \_\_\_ 1. If it is not already open, open the IBM Integration web user interface for the IIBNODE\_WITHQM integration node.
- \_\_\_ 2. Verify that the **Transformation\_Map** message flow is running on **server1**.



- \_\_\_ 3. Enable message flow statistics collection on the **Transformation\_Map** message flow.
  - \_\_\_ a. Click the down pointing arrow on the right side of the **Transformation\_Map** message flow and then click **Statistics on**.



- \_\_\_ b. A confirmation message that indicates that message flow statistics are enabled is displayed.



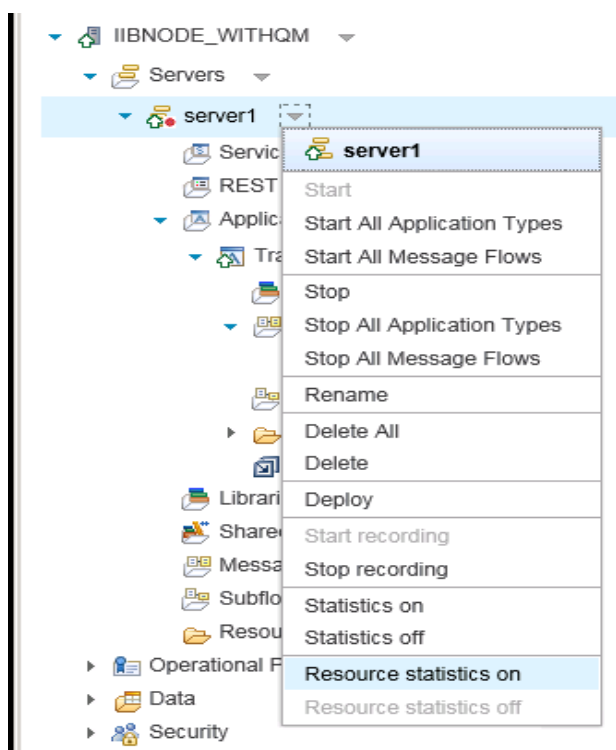
### Note

You can view the message flow statistics collection status by running the `mqsireportflowstats` command from an IBM Integration Console.

For example, to view the message flow statistics collection status for the message flows that are running on the **server1** integration server, type:

```
mqsireportflowstats IIBNODE_WITHQM -a -e server1 -j
```

- \_\_\_ 4. Enable resource statistics collection on the **server1** integration server that is running on IIBNODE\_WITHQM.
- \_\_\_ a. Click the down pointing arrow on the right side of the **server1** integration server and then click **Resource statistics on**.
- \_\_\_ b. A confirmation message that indicates that resource statistics are enabled is displayed.



### Note

You can also view the resource statistics collection status by running the `mqsireportresourcestats` command from an IBM Integration Console.

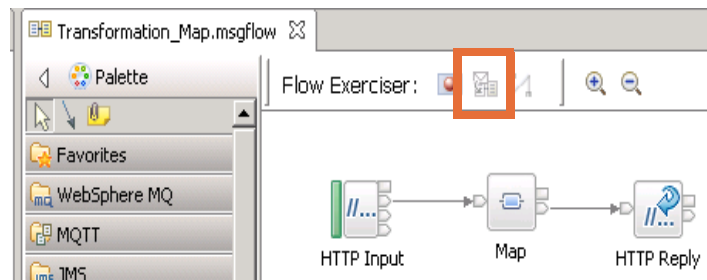
For example, to view the resource statistics collection status for the **server1** integration server that is running on the IIBNODE\_WITHQM enter:

```
mqsireportresourcestats IIBNODE_WITHQM -e server1
```

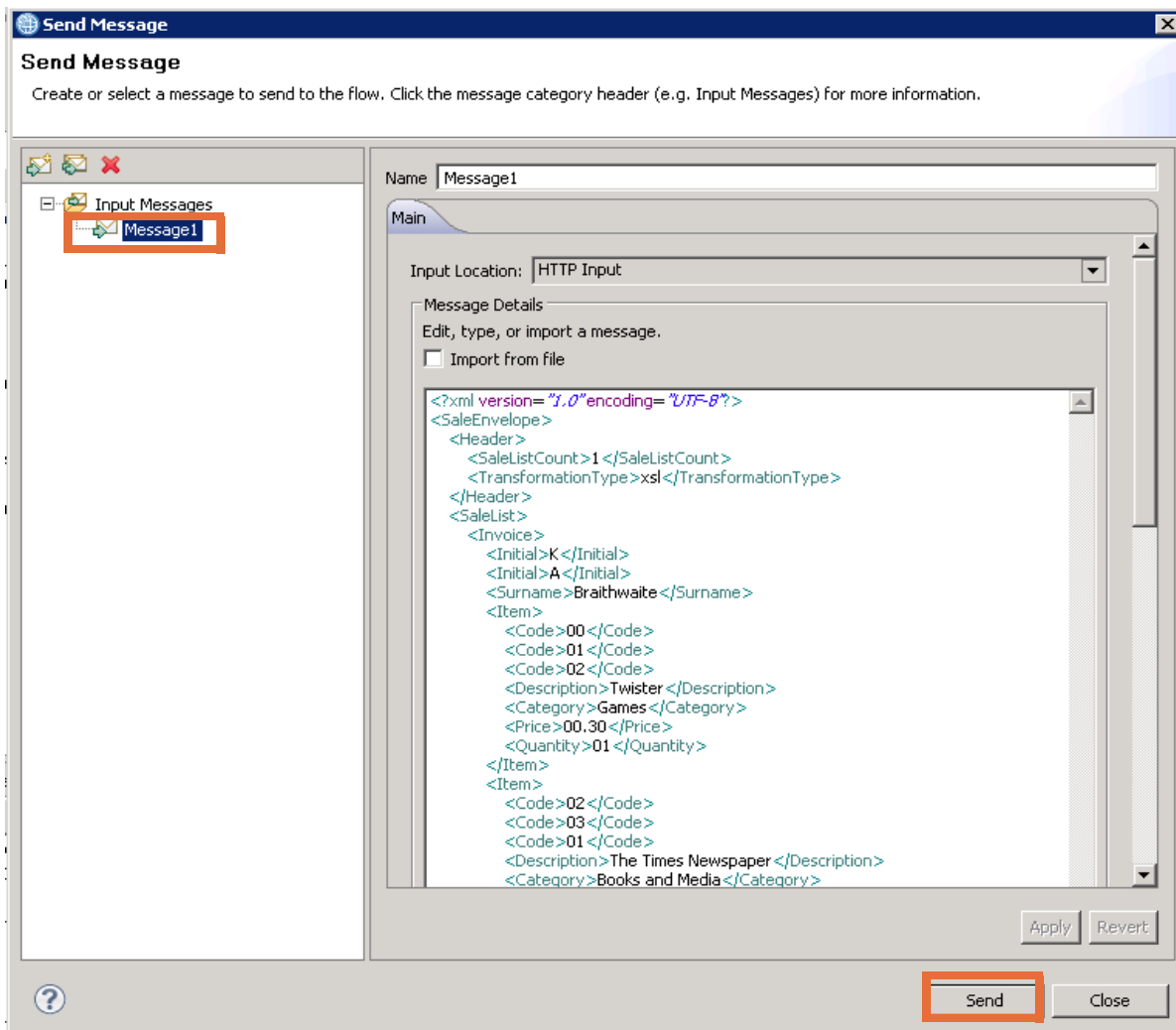
## Part 2: View message flow and resource statistics

In this part of the exercise, you use the IBM Integration web interface to view the integration server resource statistics and then message flow statistics.

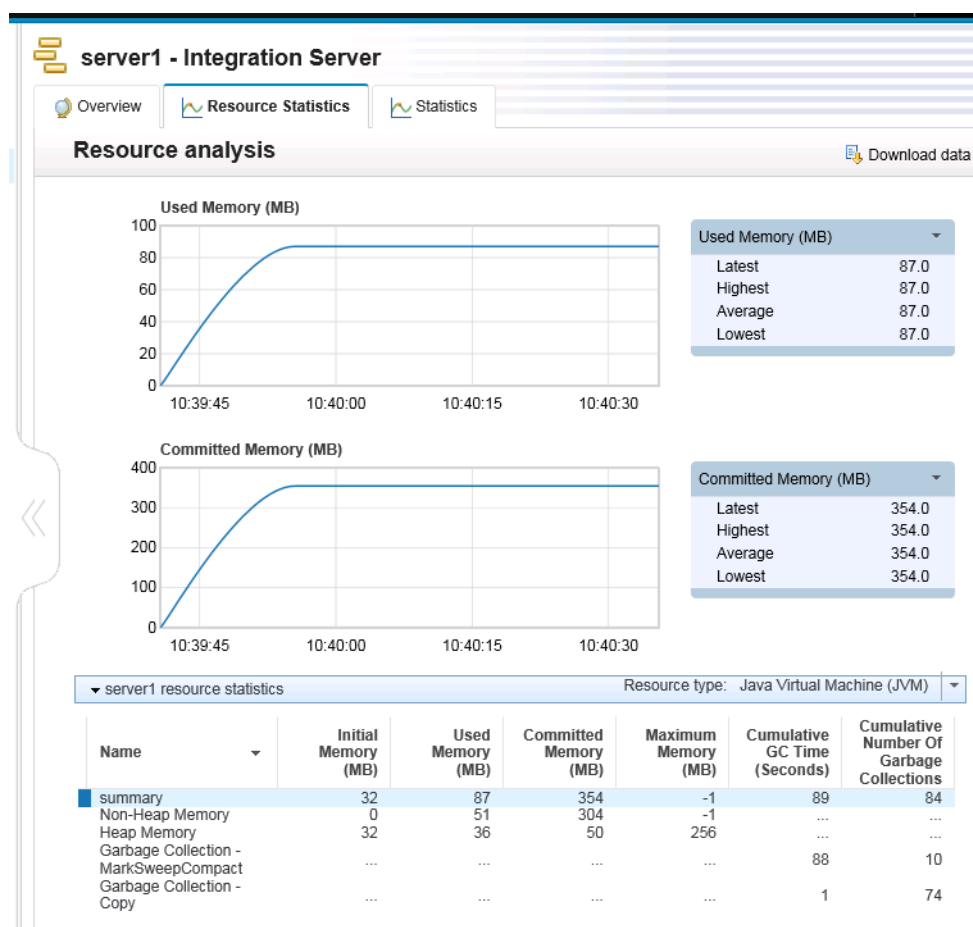
- \_\_\_ 1. Send a test message to the flow by using the Integration Toolkit Flow Exerciser.
  - \_\_\_ a. Click the **Send a message to the flow** icon in the Flow Exerciser toolbar.



- \_\_\_ b. In the Send Message window, click **Message1** in the **Input Messages** folder.
- \_\_\_ c. Click **Send**.



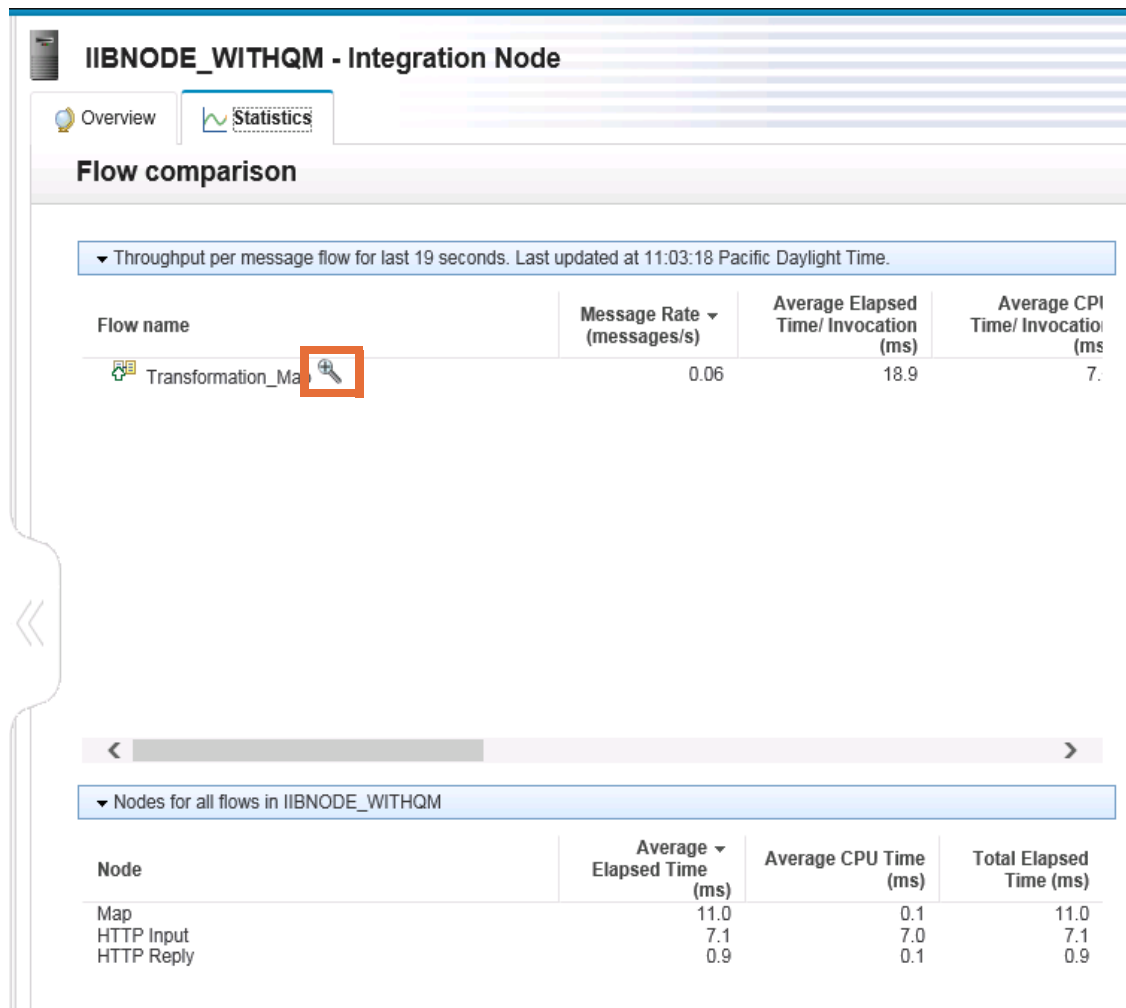
- \_\_\_ d. Click **Close** on the Progress Information window.
- \_\_\_ 2. View the integration server resource statistics for **server1** in the IBM Integration web interface.
  - \_\_\_ a. Click the **server1** integration server to display its properties and tabs.
  - \_\_\_ b. Click the **Resource Statistics** tab to display the **Resource analysis** view.



- \_\_\_ c. In the table at the bottom of the **Resource analysis** view, click **Heap Memory** to add it to the graph view. You should see a new line in the graph view that matches the color that is shown for **Heap Memory** in the resource table.
- \_\_\_ d. In the table at the bottom of the **Resource analysis** view, click **Non-Heap Memory** to add it to the graph view. You should see a new line in the graph view that matches the color that is shown for **Non-Heap Memory** in the resource table.
- \_\_\_ e. In the **Used Memory** graph, hover your cursor over any of the graph lines to display the data points.

If you are doing these exercises on a course image, it might take a few seconds for the interface to recognize that your cursor is stopped over a line in the graph.

- \_\_\_ 3. View the message flow statistics for **Transformation\_Map** message flow.
- \_\_\_ a. Click **IIBNODE\_WITHQM** to display its properties and tabs.
- \_\_\_ b. Click the **Statistics** tab to display the **Flow comparison** view.



The **Flow comparison** view displays statistical information for all message flows in the selected resource. You can use this information to see where you might enhance performance in terms of resource usage, message flow throughput, and response times.

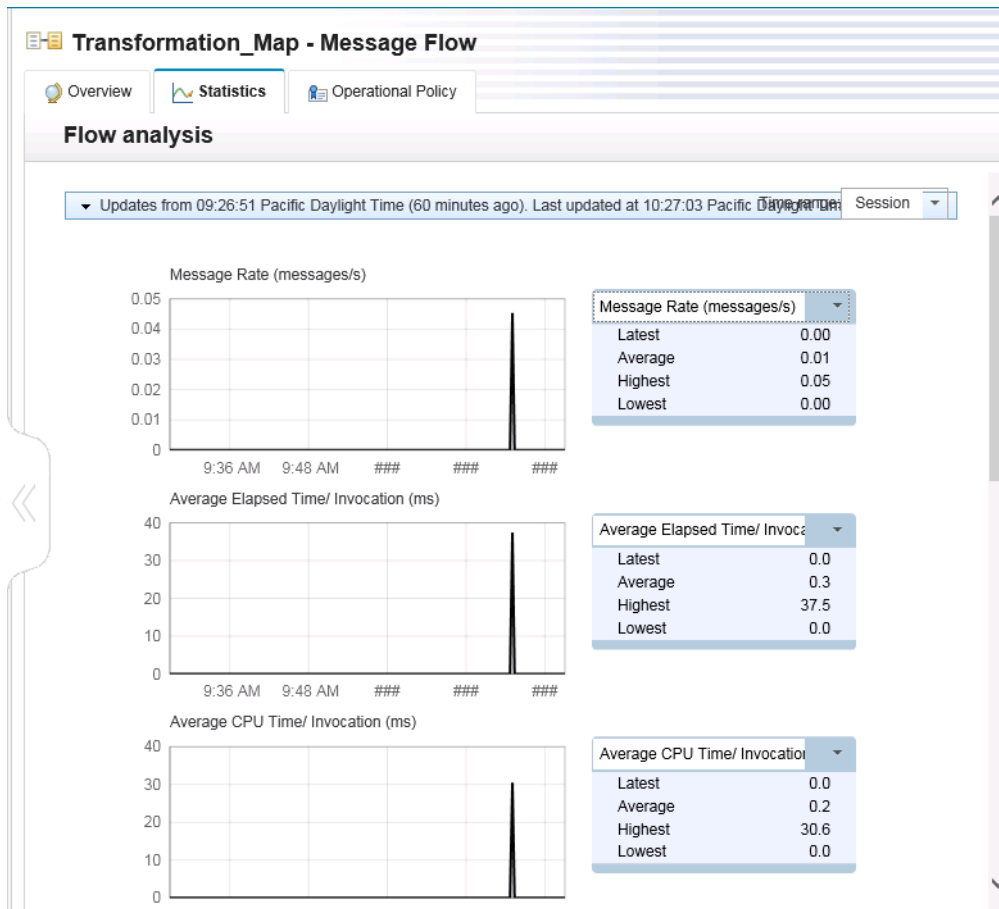
- \_\_\_ c. When you identify a message flow that requires attention, you can use the **Flow analysis** view to see more detailed statistical information about the nodes in the flow.

Click the magnifying glass icon to the right of the message flow name (**Transformation\_Map**) to go to the **Flow analysis** view for the message flow.

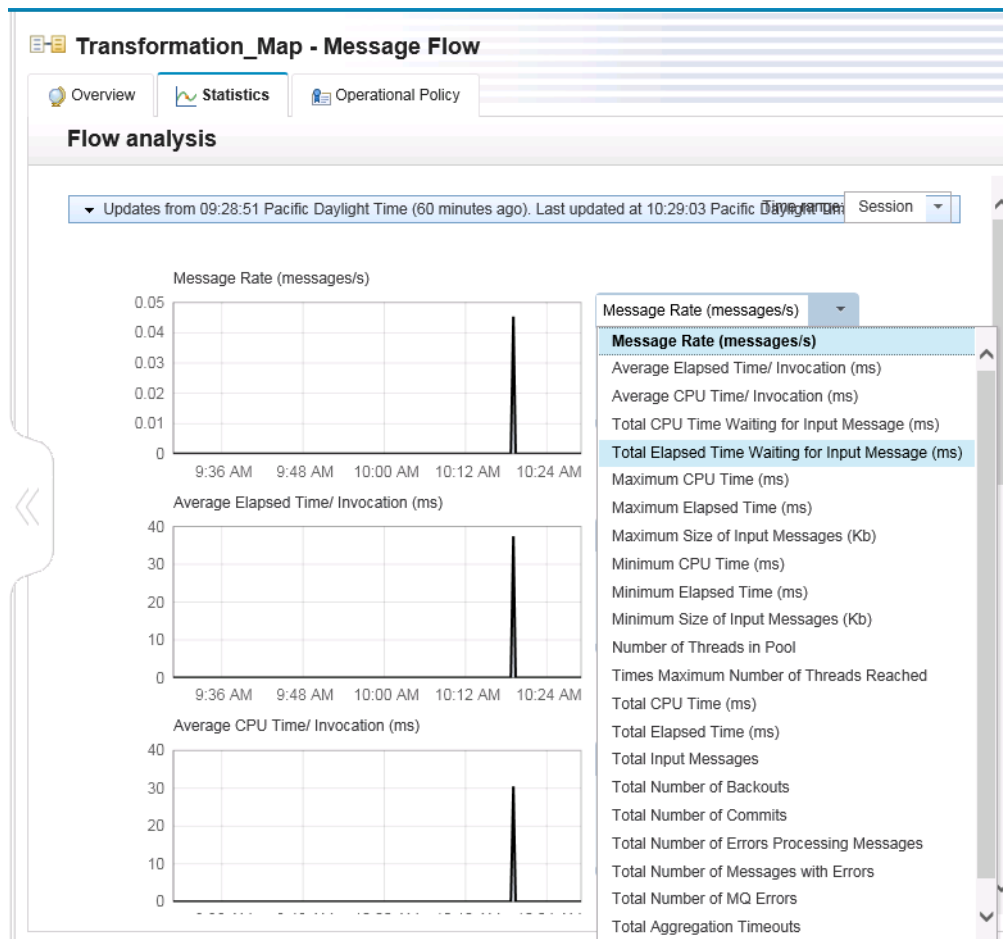
- \_\_\_ d. The **Flow analysis** view contains statistical information about the selected message flow. The view includes data such as the message rate, CPU time, and elapsed time for messages that are processed by each node in the message flow.



The charts in this view show data for three statistics at a time. By default the charts show the message rate, average elapsed time, and average CPU time for the selected message flow.



- \_\_\_ e. Change the statistics to display in the charts by clicking the arrow in the table next to each graph and then selecting the required statistic from the list.



- \_\_\_ f. Scroll down in this view to also see the average elapsed time and average CPU time for each message processing node and the map profile.

To see real-time updates in the **Statistics** view, go back to the Integration Toolkit and send another test message by using the Flow Exerciser.

- \_\_\_ 4. Review the statistics.

### Part 3: Set up subscriptions for resource statistics

If you enable resource statistics collection for one or more integration servers on an integration node, you can subscribe to the messages that the integration node publishes.

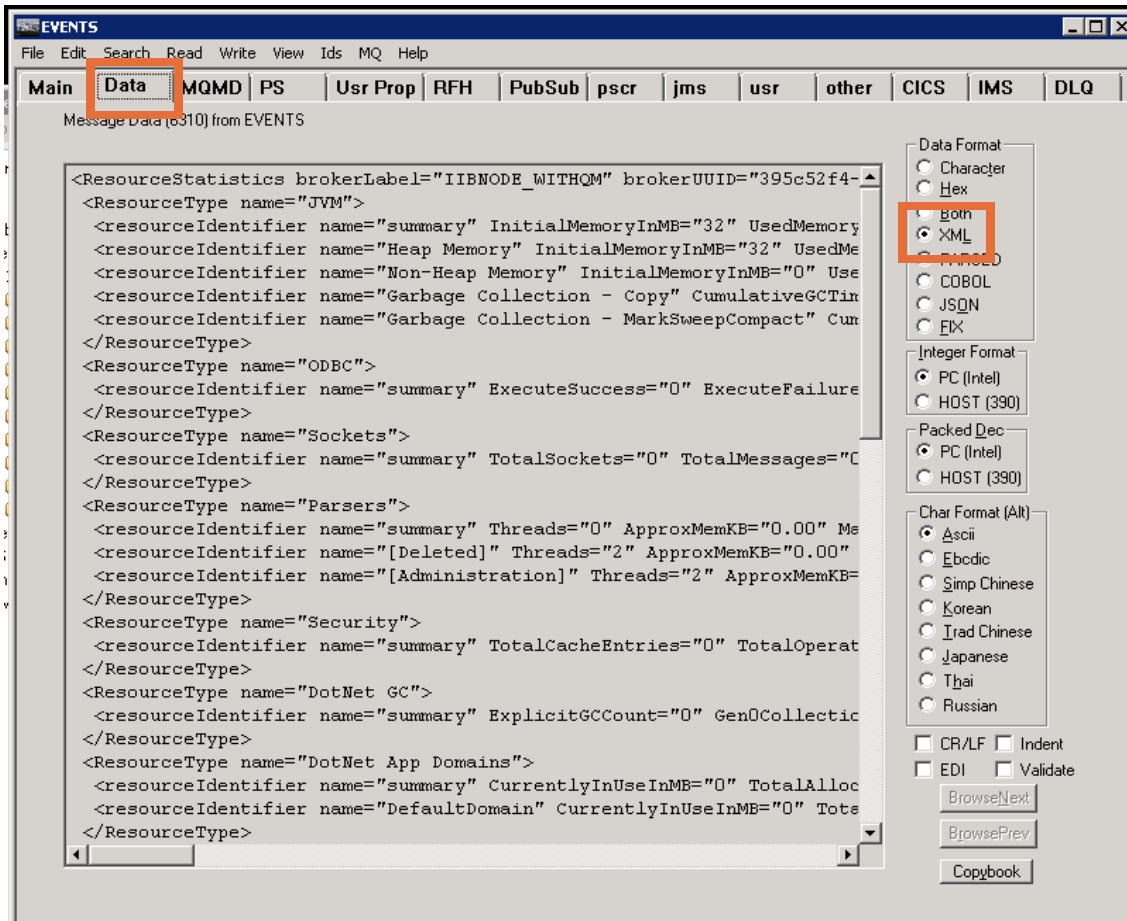
In this part of the exercise, you subscribe to the resource statistics topic from IBM MQ Explorer.

- \_\_\_ 1. In IBM MQ Explorer, create the local queue that is named EVENTS.

- \_\_\_ 2. In IBM MQ Explorer, create a subscription for the resource statistics.
  - \_\_\_ a. Right-click **Subscriptions** under IIBQM and then click **New > Subscription**.
  - \_\_\_ b. For the subscription name, type **IIBResourceStatistics** and then click **Next**.
  - \_\_\_ c. Configure the subscription properties on the Change properties window:
    - For **Topic name**, click **Select**, select **SYSTEM.BROKER.MB.TOPIC**, and then click **OK**.
    - For **Topic string**, type: **+/ResourceStatistics/#**
    - For **Destination queue manager**, type: **IIBQM**
    - For **Destination name**, type: **EVENTS**

- \_\_\_ d. Click **Finish**, and then click **OK** on the confirmation window.
- \_\_\_ e. The new subscription **IIBResourceEvents** with a topic string of **\$/SYS/Broker/+/ResourceStatistics/#** should now appear in the Subscriptions content view.
- \_\_\_ 3. Use RfhUtil to view the resource statistics from the subscription destination message queue and create an XML file that contains the resource statistics.
  - \_\_\_ a. Start RfhUtil.
  - \_\_\_ b. Select **IIBQM** for the **Queue Manager Name**.
  - \_\_\_ c. Select **EVENTS** for the **Queue Name**.

- \_\_\_ d. Click **Browse Q**.
- \_\_\_ e. Click the **Data** tab. You should see the resource statistics.
- \_\_\_ f. Under **Data Format**, select **XML** so the data can be viewed as XML data.



- \_\_\_ 4. To make it easier to read the statistics data, copy the data into Windows Notepad, save it as an XML file, and open it in a web browser.
  - \_\_\_ a. Select the message data in RFHUtil, right-click the selected data, and then click **Copy**.
  - \_\_\_ b. Open Windows Notepad.
  - \_\_\_ c. Right-click in the Notepad session and then click **Paste**.
  - \_\_\_ d. Save the file in **C:\labfiles\Lab10-Stats** directory as **Statistics.xml** and then close Windows Notepad.
  - \_\_\_ e. In Windows Explorer, locate the **Statistics.xml** file and then double-click it to start a web browser window.
- \_\_\_ 5. Review the statistics. When you are finished, close the web browser window.

## Exercise clean up

- \_\_\_ 1. In the IBM Integration web interface, stop the collection of resource statistics.  
Click the down pointing arrow on the right side of the **server1** integration server and then click **Resource statistics off**.
- \_\_\_ 2. In the IBM Integration web interface, stop message flow accounting and statistics.  
Click the down pointing arrow on the right side of the **Transformation\_Map** message flow and then click **Statistics off**.
- \_\_\_ 3. Close the RFUtil session.
- \_\_\_ 4. Delete the **Transformation\_map** message flow components from the **server1** integration server on IIBNODE\_WITHQM.
- \_\_\_ 5. In the IBM Integration Toolkit, stop the Flow Exerciser.
- \_\_\_ 6. In the IBM Integration Toolkit **Integration Nodes** view, right-click the **server1** integration server and then click **Stop recording**.

## End of exercise

## Exercise review and wrap-up

In this exercise, you used the IBM Integration web interface to enable integration node and message flow statistics collection. You then used the IBM Integration web interface to view the resource and message flow statistics.

Finally, you used IBM MQ Explorer to create a subscription for resource statistics.

Having completed this exercise, you should be able to:

- Enable integration node resource and message flow statistics
- View integration node statistics in the IBM Integration web user interface
- Subscribe to integration node statistics from IBM MQ

# Exercise 11.Administering workload management policies

## What this exercise is about

In this exercise, you use the IBM Integration web interface and IBM Integration commands to create workload management policies. You also attach workload management policies to a message flow.

## What you should be able to do

After completing this exercise, you should be able to:

- Create workload management policies
- Attach a workload management policy to a message flow

## Introduction

Administrators can use a workload management policy to control the workload management attributes without the need to redeploy the message flow or integration services resources.

Workload management policies are set at the message flow level. Using a workload management policy, the system administrators can complete the following tasks:

- Restrict the maximum rate at which a message flow can run
- Cause a notification message to be sent if the number of messages that arrive in the flow exceeds a specified threshold

A workload management policy can be set up and administered within the Integration Registry by using the IBM Integration web interface, IBM Integration commands, and the IBM Integration API.

In the first part of this exercise, you create a workload management policy by using the IBM Integration web interface.

In the second part of the exercise, you attach a workload management policy to a message flow. You also use message flow statistics to observe the effects of the attached workload policy on the message flow.

## Requirements

- IBM Integration Bus V10
- The IIBNODE\_NOQM integration node that was created in Exercise 1

- Lab files in the C:\labfiles\Lab11-WLM directory

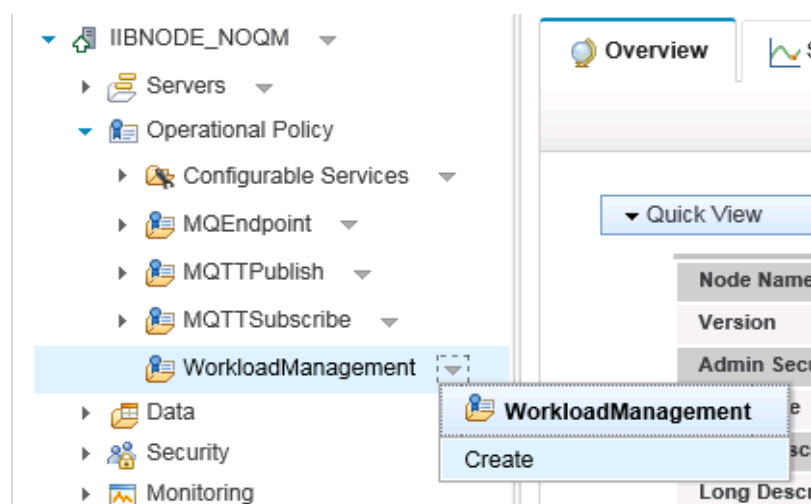


## Exercise instructions

### Part 1: Create a workload management policy

In this part of the exercise, you use the IBM Integration web interface to create a workload management policy that is defined within a configurable service.

- \_\_\_ 1. Open the IBM Integration web interface for the IIBNODE\_NOQM integration node.
- \_\_\_ 2. In the navigator, expand **Operational Policy**.
- \_\_\_ 3. Click the down arrow on the right side of **Workload Management** and then click **Create**.



- \_\_\_ 4. For the policy name, type: **DefaultWlmPolicy**
- \_\_\_ 5. The **Additional Instances** property specifies the additional threads that the integration node can use to service the message flow.

For this exercise, set the **Additional Instance** property to 3.

Also, set the **Start additional instances when flow starts** property to **Yes**.

- \_\_\_ 6. The **Processing Timeout** property is the maximum time a message flow can process a message before taking a specified action. The time is measured in seconds. The time is taken from the point that the message is received on an input node.

For this exercise, set the **Processing Timeout** to 60.



#### Information

Automatically stopping a flow by using the workload management policy, stops the flow after the specified amount of processing time.

There is no way to know whether the message flow was in a loop, only that it used too many cycles, and should be stopped. When restarted, it might automatically restart again or continue processing until finished. If problems continue with the message flow, you can manually force it to stop by using the `mqsisstopmsgflow -f` command.

- \_\_\_ 7. The **Processing timeout action** property specifies the action to take when the **Processing timeout** is exceeded.

For this exercise, set **Processing timeout action** to **restartExecutionGroup**, which restarts the integration server. Set the **Processing timeout** to 60.

**Operational Policy - WorkloadManagement : DefaultWlmPolicy**

Overview

Save Save As Revert

Use a policy to control the operational behavior of a message flow at run time. [ More... ] Oct 21, 2015, 7:27:29 AM x

Policy URL /apiv1/policy/WorkloadManagement/DefaultWlmPolicy

Targets and Limits

Additional Instances

Additional instances 3

Start additional instances when flow starts Yes

Start Mode

Start mode Maintained

Transactionality

Unresponsive Message Flows

Processing timeout action Restart execution group

Processing timeout 60 seconds

- \_\_\_ 8. Click **Save**.
- \_\_\_ 9. Create another workload management policy that is named **TenAdditionalInstances**. Use the same values that you used for the **DefaultWlmPolicy** policy but set the **Additional instances** property to 10.

## Part 2: Attach a workload management policy to a message flow

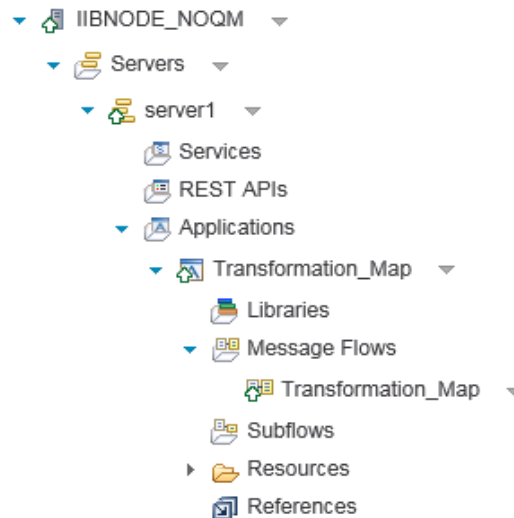
In this part of the exercise, you deploy a message flow and then attach a workload management policy. You also use message flow statistics to observe the effects of the attached workload policy on the message flow.

- \_\_\_ 1. Using the IBM Integration web interface, deploy the BAR file `SimpleFlow.bar` (in the `C:\labfiles\Lab11-WLM\resources` directory) to the **server1** integration server.

- \_\_\_ 2. The SimpleFlow.bar file contains the **Transformation\_Map** application, which contains the **Transformation\_Map** message flow.

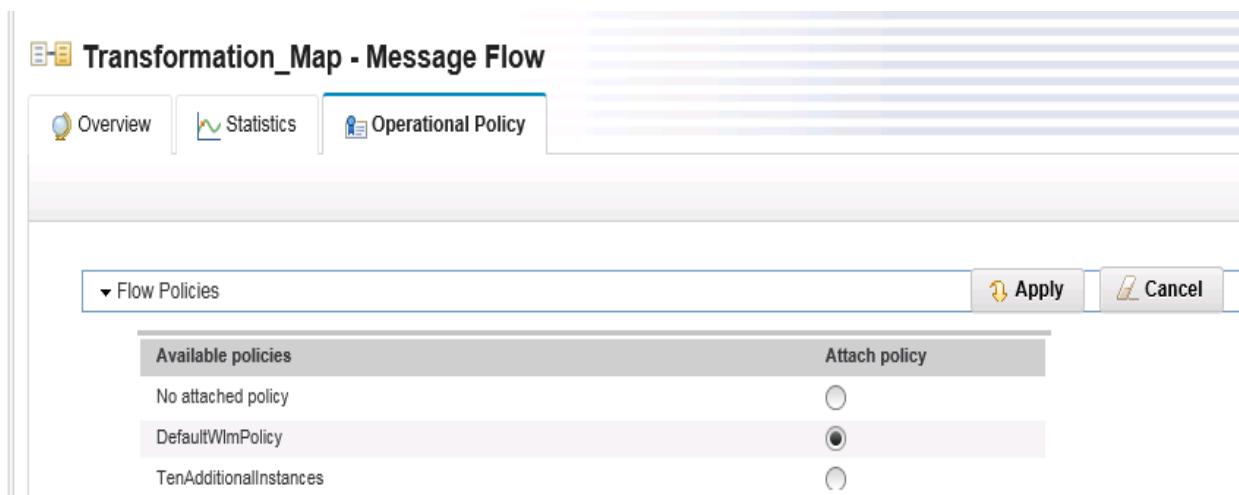
Expand the **Transformation\_Map** application until you see the **Transformation\_Map** message flow.

You should see that the **Transformation\_Map** message flow is running.



- \_\_\_ 3. Click the down arrow on right side of **Transformation\_Map** message flow and then click **Attach Policy**. The **Operational Policy** view opens.
- \_\_\_ 4. The **Operational Policy** view shows the list of available workload management policies for the integration node.

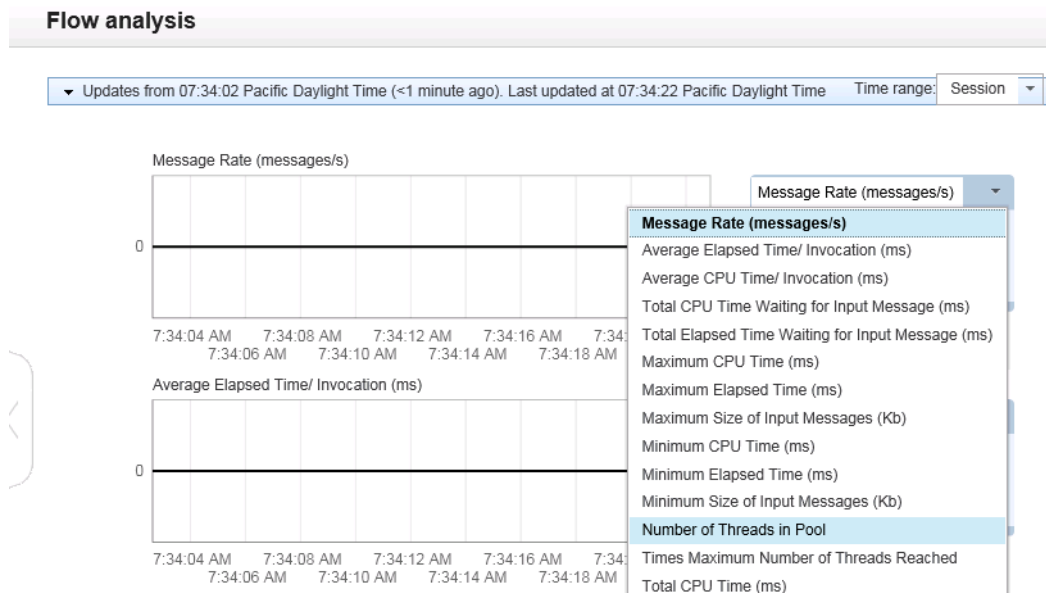
Click **Attach policy** for the **DefaultWlmPolicy** workload management policy and then click **Apply**.



You should see a message that the policy was successfully attached.

- \_\_\_ 5. Enable message flow statistics for the message flow.
- \_\_\_ 6. Click the **Statistics** tab to view the message flow statistics for the **Transformation\_Map** message flow.

- \_\_\_ 7. Change the information that the first graph displays to **Number of threads in pool**.

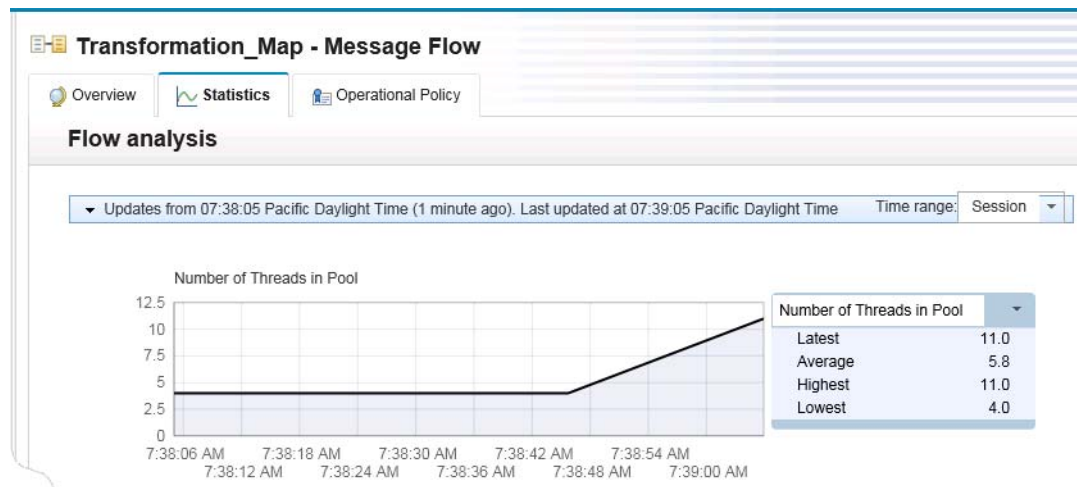


You should see that one thread is running.

- \_\_\_ 8. Stop and then start the message flow.
- \_\_\_ 9. In the **Statistics** view, you should now see that four instances are available in the thread pool (the original instance plus three more instances).



- \_\_\_ 10. Detach the **DefaultWlmPolicy** policy.
- Click down arrow to the right of message flow and then click **Detach policy**.
- \_\_\_ 11. Attach the **TenAdditionalInstances** workload management policy to the message flow.
- \_\_\_ 12. Stop and then start the message flow.
- \_\_\_ 13. View the **Number of threads in pool** in the **Statistics** view. You should now see that 11 instances are available in the thread pool.



## Exercise clean up

- \_\_\_ 1. Stop the **Transformation\_Map** message flow application from server1 on the IIBNODE\_NOQM integration node.
- \_\_\_ 2. Delete all flows and resources from server1 on the IIBNODE\_NOQM integration node.
- \_\_\_ 3. Close the IBM Integration web interface browser window for IIBNODE\_NOQM.

## End of exercise

## Exercise review and wrap-up

In the first part of this exercise, you created a workload management policy by using the IBM Integration web interface.

In the second part of the exercise, you attached a workload management policy to a message flow. You also used message flow statistics to observe the effects of the attached workload policy on the message flow.

Having completed this exercise, you should be able to:

- Create workload management policies
- Attach a workload management policy to a message flow

# Exercise 12. Recording and replaying message flow data

## What this exercise is about

In this exercise, you define message flow events and then record them in a database. You also replay messages and capture and report failed events.

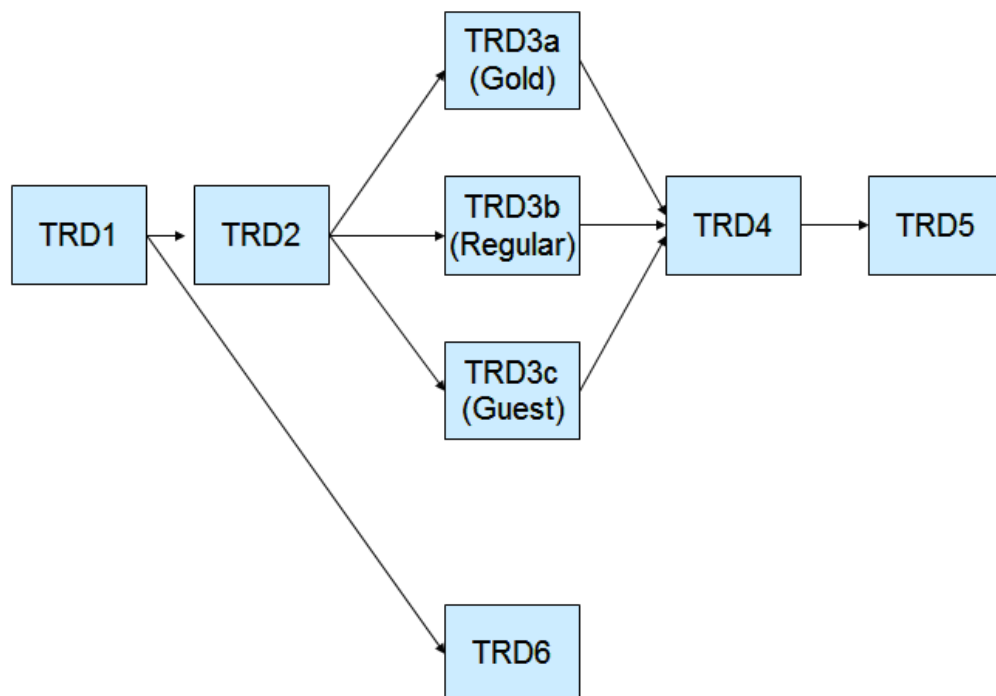
## What you should be able to do

After completing this exercise, you should be able to:

- Activate message flow monitoring and store events in a database
- Enable an integration node to connect to a database with ODBC
- View event messages in the IBM Integration web user interface
- Replay a message to an IBM MQ queue by using the IBM Integration web interface

## Introduction

This exercise uses an application that contains several message flows (summarized in the following figure).



The application processes stock trade requests, and each trade runs five message flows: TRD1, TRD2, TRD3\*, TRD4, and TRD5. The TRD3 message flow is selected based on the customer type:

- If the customer type is “Gold”, the TRD3a message flow runs.
- If the customer type is “Regular”, the TRD3b message flow runs.
- If the customer type is “Guest”, the TRD3c message flow runs.

The TRD6 flow runs if a validation failure occurs in the first message flow, TRD1.

Several of the message flow nodes have monitoring points that are defined on them by using the **Monitoring** node properties. These monitoring points publish certain items of the message payload data. The **Data viewer** in the IBM Integration web user interface uses this data to access processed messages, and to resubmit (replay) the message for reprocessing.



#### Note

Flow monitoring can also be achieved noninvasively by using monitoring templates, which are not covered in this lab.

In this second part of this exercise, you set up a DB2 database for record and replay database. You also configure the JDBC connection to the database, and the IBM MQ queues that are required for the Trades application. Command and batch files are provided in the `C:\labfiles\Lab12-RandR\install` directory to automate this process.

## Requirements

- IBM Integration Bus V10
- IBM MQ V8 and a queue manager that is named IIBQM
- IBM DB2
- An IBM Integration Bus integration node with an IBM MQ queue manager specified on the integration node. The exercise instructions refer to IIBNODE\_WITHQM that is created in Exercise 2
- Lab files in the `C:\labfiles\Lab12-RandR` directory
- The IBM MQ queues (created in the exercise): TRADE.VALIDATE.IN, TRADE.CUST.TYPE.IN, TRADE.GOLD.IN, TRADE.REGULAR.IN, TRADE.VALIDATION.FAILURE.IN, TRADE.GUEST.IN, TRADE.RECONCILIATION.IN, TRADE.COMPLETE.IN, TRADE.COMPLETE.OUT, TRADE.FIX.IN, TRADE.FIX.OUT, TRADE.REPLAY.INPUT



## Exercise instructions

### Part 1: Set up the environment for record and replay

In this part of the exercise, you set up the record and replay database, JDBC connections, and the queues that are required for the TRADES application.

Command and batch files are provided in the C:\labfiles\Lab12-RandR\install directory to automate the setup process. You can examine any of the files by using Notepad.

- \_\_\_ 1. Create the queues and subscriptions for the TRADES message flows on the queue manager that is specified on the integration node.
  - \_\_\_ a. Start an IBM Integration Console as the administrator (right-click the **IBM Integration Console** shortcut on the desktop and then click **Run as administrator**).
  - \_\_\_ b. In the IBM Integration Console, change directories to C:\labfiles\Lab12-RandR\install\MQsetup. Type:
 

```
cd C:\labfiles\Lab12-RandR\install\MQsetup
```
  - \_\_\_ c. Enter the command to create the queues on the integration node queue manager IIBQM:
 

```
runmqsc IIBQM < TradeQueues.mqsc
```
  - \_\_\_ d. In IBM MQ Explorer, verify that the following objects were created:
    - The local queues TRADE.VALIDATE.IN, TRADE.CUST.TYPE.IN, TRADE.GOLD.IN, TRADE.REGULAR.IN, TRADE.VALIDATION.FAILURE.IN, TRADE.GUEST.IN, TRADE.RECONCILIATION.IN, TRADE.COMPLETE.IN, TRADE.COMPLETE.OUT, TRADE.FIX.IN, TRADE.FIX.OUT, TRADE.REPLAY.INPUT, and RECORD.REPLAY.SUB
    - The subscription RECORD.REPLAY.SUB with the topic string of IIBNODE\_WITHQM/Monitoring/#
- \_\_\_ 2. Create the record and replay database.
  - \_\_\_ a. In the IBM Integration Console, type the following commands to create the DB2 database and tables:
 

```
cd ..\DBSetup
CreateTRADES_Tables.bat
```

This command opens a DB2 command window and creates the record and replay tables in the TRADES database. The tables are re-created so that this lab exercise starts with a clean display of monitor events, and so that new events are easily viewable in the IBM Integration web user interface.

It might take a few minutes for this batch file to complete.

Verify that the command ran successfully and then press Enter in the DB2 command window to close the window.
- \_\_\_ 3. Define the JDBC connections from the IIBNODE\_WITHQM integration node to the TRADES database tables.

In the IBM Integration Console, type the following command:

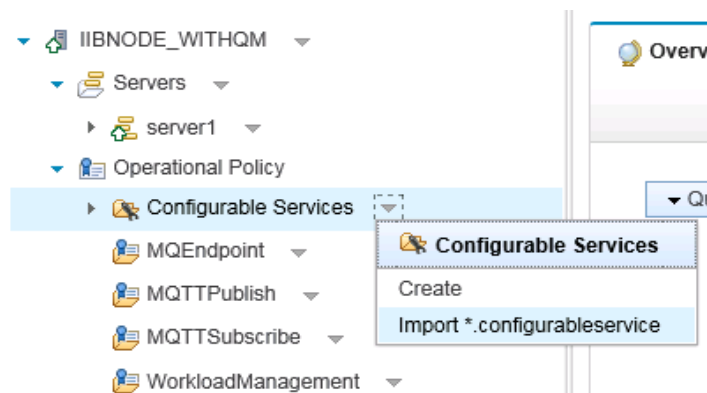
**ConfigureJDBC**

- \_\_\_ 4. Import the data store configurable service.

The record and replay function uses a data store, which represents the database that holds the captured monitoring events. The data store is defined to the integration node by using configurable services.

You can define the configurable services manually. For this lab, the configurable services are already created but they must be imported into the IIBNODE\_WITHQM integration node.

- \_\_\_ a. In the IBM Integration web interface for IIBNODE\_WITHQM, expand the **Operational Policy** folder.
- \_\_\_ b. Click the down arrow to the right of the **Configurable Services** folder and then click **Import \*.configurableservice**.



- \_\_\_ c. Browse to the C:\labfiles\Lab12-RandR\configurable\_services directory and then select the **Trades\_data\_capture\_store.configurableservice** file. Click OK.
- \_\_\_ d. The **Trades** configurable service should now be listed under the **Configurable Services** > **DataCaptureStore** folder.

The screenshot shows the IIB console with the 'Trades - DataCaptureStore Configurable Service' selected. The left pane shows the project hierarchy: IIBNODE\_WITHQM > Servers > server1 > Operational Policy > Configurable Services > DataCaptureStore > Trades. The right pane shows the 'Overview' tab with a 'Properties' table.

| Properties                |                          |
|---------------------------|--------------------------|
| commitCount               | 10                       |
| useCoordinatedTransaction | false                    |
| threadPoolSize            | 10                       |
| queueName                 | SYSTEM.BROKER.DC.RECORD  |
| dataSourceName            | TRADES                   |
| schema                    | IIBADMIN                 |
| egForView                 | server1                  |
| backoutQueue              | SYSTEM.BROKER.DC.BACKOUT |
| egForRecord               | server1                  |
| commitIntervalSecs        | 5                        |

- \_\_\_ 5. Following the same process as in the step 4, import `Trades_source.configurableservice` and `Trades_BPM_Data_Destination.configurableservice` from the `C:\labfiles\Lab12-RandR\configurable_services` directory.

You should now have three new configurable services for the Trades application:

- **DataCaptureSource > Trades\_Source**
- **DataCaptureStore > Trades**
- **DataDestination > Trades\_Redirect\_to\_BPM**

The screenshot shows the IIB console with the updated project hierarchy. The left pane shows: IIBNODE\_WITHQM > Servers > server1 > Operational Policy > Configurable Services. Under 'DataCaptureSource', 'Trades\_Source' is listed. Under 'DataCaptureStore', 'Trades' is listed. Under 'DataDestination', 'Trades\_Redirect\_to\_BPM' is listed. The 'Trades\_Redirect\_to\_BPM' service is currently selected.

**Note**

The **Trades\_Source** service subscribes to the topic `$SYS/Broker/IIBNODE_WITHQM/Monitoring/server1/#`. So this data source collects all monitoring events that the applications in the **server1** integration server generate. This data source does not collect events that are emitted in other nodes or integration servers.

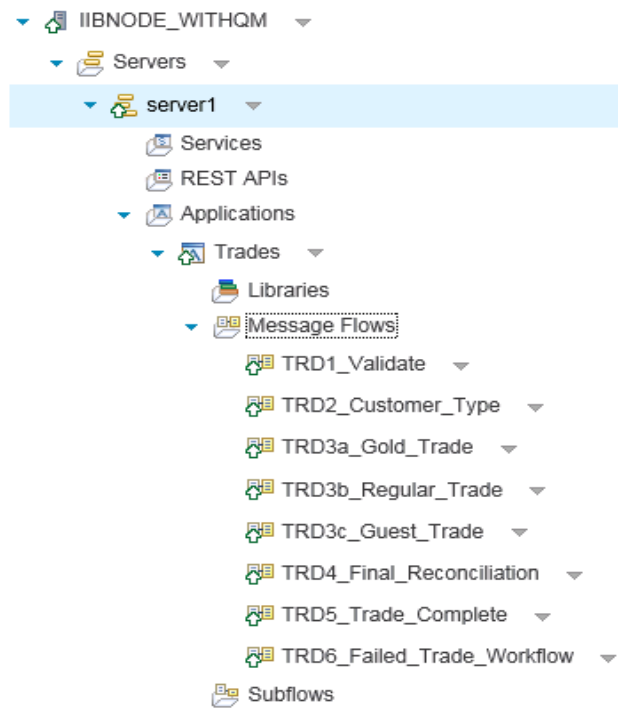
## Part 2: View messages in the IBM Integration web interface

1. In the IBM Integration web interface, expand **Data > Data Capture Stores** in the navigator and then select the **Trades** data capture store.

The events list for the **Trades** database is empty.

2. In the IBM Integration web interface, deploy the BAR File that is named **Trades.bar** from the `C:\labfiles\Lab12-RandR\application` directory to the **server1** integration server.
3. Validate that the **Trades** application is deployed.

Expand the **Trades** application in the IBM Integration web user interface. Verify that the application contains eight message flows. The message flows are run in sequence, with just one of the TRD3\* messages flows used, depending on the type of customer.



- \_\_\_ 4. Enable flow monitoring on the message flows in the **Trades** application.  
Type the following command to enable monitoring on each of the message flows:

```
cd ../../monitoring
enableMonitoringTrades
```



### Information

This command runs the integration node commands:

```
mqsichangeflowmonitoring IIBNODE_WITHQM -e server1 -k Trades
-f TRD1_Validate -c active
```

It also runs equivalent commands for the other message flows in the **Trades** application.

If you redeploy the **Trades** application, the flow monitoring status is reset. You must enter the `enableMonitoringTrades` command again to reactivate flow monitoring.

- \_\_\_ 5. Use RfhUtil to send some new events to the **Trades** application queue TRADE.VALIDATE.IN on the IIBQM queue manager.
- \_\_\_ a. In RfhUtil, open the file for a “Gold” customer:  
C:\labfiles\Lab12\_RandR\data\TradeMessageGold\_BNY347290.xml
  - \_\_\_ b. Send one instance of the data to the queue TRADE.VALIDATE.IN on IIBQM.

Queue Manager Name (to connect to): IIBQM

Queue Name: TRADE.VALIDATE.IN

Remote Queue Manager Name (remote queues only):

Selector:

File Code Page: 437

File Name: C:\labfiles\Lab12-RandR\data\TradeMessageGold\_BNY347290.xml

Data Size: 433

User Props:

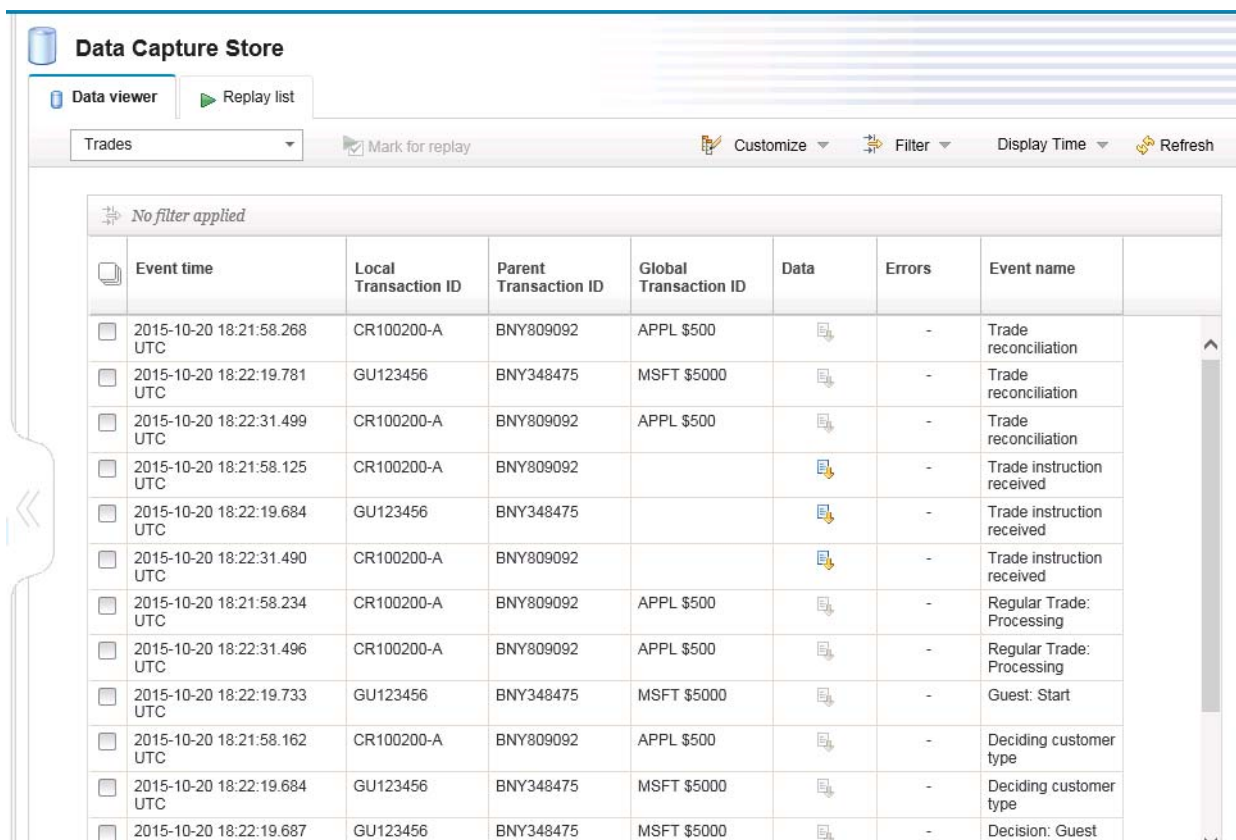
- ☒ As Queue
- ☐ None
- ☐ Yes
- ☐ RFH2
- ☐ Compat

- \_\_\_ c. Send a message to TRADE.VALIDATE.IN for a “Regular” customer:  
C:\labfiles\Lab12-RandR\data\TradeMessageRegular\_BNY809092.xml.
- \_\_\_ d. Send a message to TRADE.VALIDATE.IN for a “Guest” customer:  
C:\labfiles\Lab12-RandR\data\TradeMessageGuest\_BNY348475.xml.
- \_\_\_ 6. Click **Refresh** on the **Data viewer** in the IBM Integration web interface to show the new events.



### Note

If you receive an error when trying to access the Data Capture Store, restart the **server1** integration server and try to access the **Data viewer** again.



**Data Capture Store**

Data viewer | Replay list

Trades | Mark for replay | Customize | Filter | Display Time | Refresh

No filter applied

|                          | Event time                  | Local Transaction ID | Parent Transaction ID | Global Transaction ID | Data | Errors | Event name                 |
|--------------------------|-----------------------------|----------------------|-----------------------|-----------------------|------|--------|----------------------------|
| <input type="checkbox"/> | 2015-10-20 18:21:58.268 UTC | CR100200-A           | BNY809092             | APPL \$500            |      | -      | Trade reconciliation       |
| <input type="checkbox"/> | 2015-10-20 18:22:19.781 UTC | GU123456             | BNY348475             | MSFT \$5000           |      | -      | Trade reconciliation       |
| <input type="checkbox"/> | 2015-10-20 18:22:31.499 UTC | CR100200-A           | BNY809092             | APPL \$500            |      | -      | Trade reconciliation       |
| <input type="checkbox"/> | 2015-10-20 18:21:58.125 UTC | CR100200-A           | BNY809092             |                       |      | -      | Trade instruction received |
| <input type="checkbox"/> | 2015-10-20 18:22:19.684 UTC | GU123456             | BNY348475             |                       |      | -      | Trade instruction received |
| <input type="checkbox"/> | 2015-10-20 18:22:31.490 UTC | CR100200-A           | BNY809092             |                       |      | -      | Trade instruction received |
| <input type="checkbox"/> | 2015-10-20 18:21:58.234 UTC | CR100200-A           | BNY809092             | APPL \$500            |      | -      | Regular Trade: Processing  |
| <input type="checkbox"/> | 2015-10-20 18:22:31.496 UTC | CR100200-A           | BNY809092             | APPL \$500            |      | -      | Regular Trade: Processing  |
| <input type="checkbox"/> | 2015-10-20 18:22:19.733 UTC | GU123456             | BNY348475             | MSFT \$5000           |      | -      | Guest: Start               |
| <input type="checkbox"/> | 2015-10-20 18:21:58.162 UTC | CR100200-A           | BNY809092             | APPL \$500            |      | -      | Deciding customer type     |
| <input type="checkbox"/> | 2015-10-20 18:22:19.684 UTC | GU123456             | BNY348475             | MSFT \$5000           |      | -      | Deciding customer type     |
| <input type="checkbox"/> | 2015-10-20 18:22:19.687 UTC | GU123456             | BNY348475             | MSFT \$5000           |      | -      | Decision: Guest            |

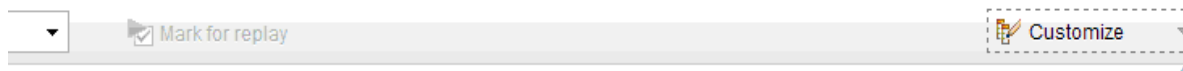
- \_\_\_ 7. The **Data viewer** uses the standard column heading names. You can customize the headings so that they are more descriptive.

There are several options for customizing the column headings:

- You can change the display name of each column by double-clicking the required display name, and entering another name. These changes are stored in the Integration Registry, and are retained uniquely for each data capture store. All users who display data from the same data capture store see the changes that this user makes. If you want to record and view data with different headings, record the events in a separate data capture store.
- You can select or clear any of the recorded fields for display.
- You can override the width of the displayed column. The widths can also be overridden by using the divider bars.

Click **Customize** and change the column heading name as follows:

- \_\_\_ a. Change **Local Transaction ID** to **Customer number**.
- \_\_\_ b. Change **Parent Transaction ID** to **Trade number**.
- \_\_\_ c. Change **Global Transaction ID** to **Stock/Trade amount**.
- \_\_\_ d. Change **Event name** to **Trade processing stage** and change the field width to 160.
- \_\_\_ e. Change **hasBitstream (Data)** so that it is not selected for display.

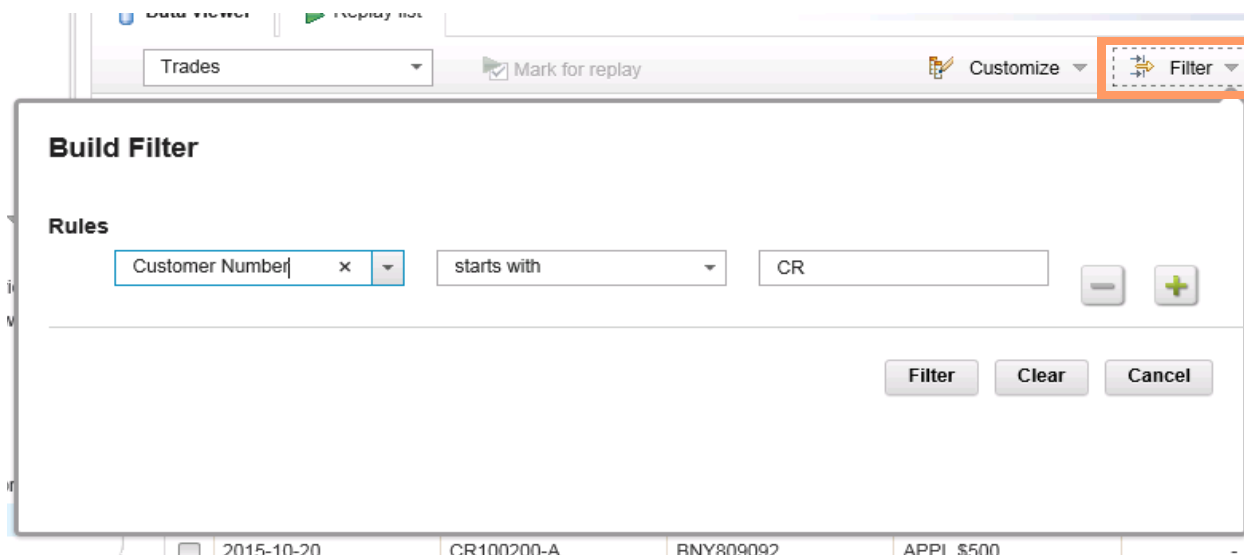


## Customize Columns

Select the columns to display in the Data viewer. Double-click a name or width that you want to edit. You can sort the order by clicking the header. You can also reorder the columns and change their widths by using the header in the main Data viewer and saving your changes here. The saved changes apply only to the current data capture store; other data capture stores retain their current settings.

| Field ID            |                                     | Display Name         | Width (px) |
|---------------------|-------------------------------------|----------------------|------------|
| eventTimestamp      | <input checked="" type="checkbox"/> | Event time           | 160        |
| localTransactionId  | <input checked="" type="checkbox"/> | Customer number      | 100        |
| parentTransactionId | <input checked="" type="checkbox"/> | Trade number         | 100        |
| globalTransactionId | <input checked="" type="checkbox"/> | Stock / Trade amount | 100        |
| hasBitstream        | <input type="checkbox"/>            | Data                 | 70         |

- \_\_\_ f. Click **Apply**. The column names are updated with the new values.
- \_\_\_ 8. You can click the column headers to change the order of the events in the **Data viewer**.  
For example, click the **Event time** column to display the oldest events first.
- \_\_\_ 9. You can limit the data that is displayed in the **Data viewer** by using the **Filter** function.
  - \_\_\_ a. Click **Filter**.
  - \_\_\_ b. Specify the filter criteria of: Customer number starts with CR
  - \_\_\_ c. Click **Filter** to activate the defined filters.



- \_\_\_ 10. You can add more filters to the display.
  - \_\_\_ a. Click **Filter** again, and click the green plus sign to add a second filter.
  - \_\_\_ b. Specify the filter criteria of: Trade processing stage contains Complete.



- \_\_ c. Click **Filter** to activate the new filter.



### Note

The values for each filter are case-sensitive.

- \_\_ 11. Clear the filters by selecting **Filter** from the **Data viewer** tab and then clicking **Clear**.

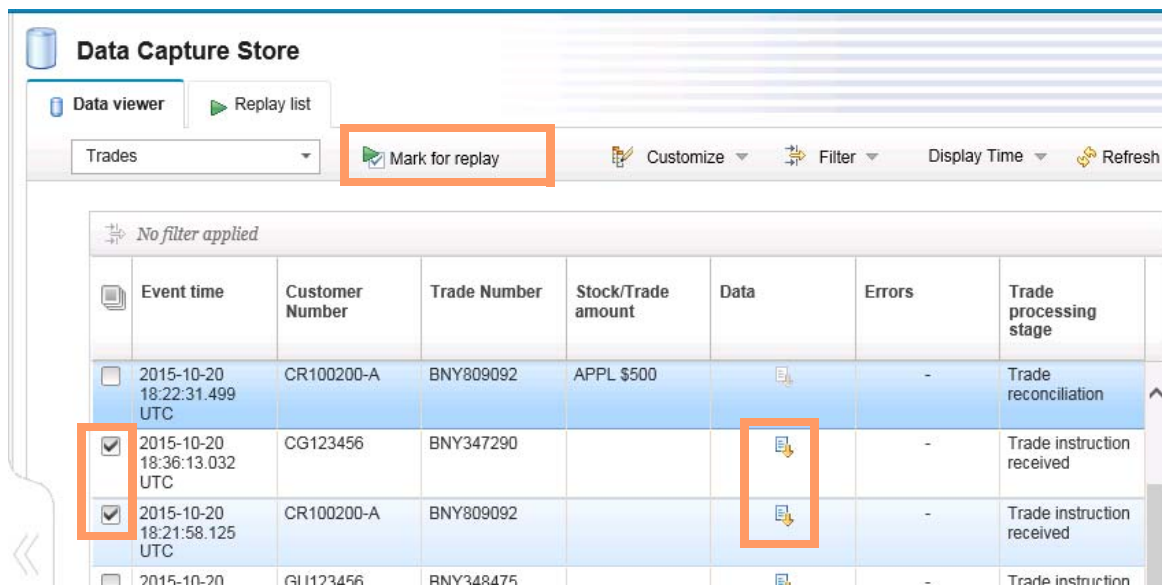
## Part 3: Replay messages

In this part of the exercise, you configure the IIBNODE\_WITHQM integration node to replay messages. Replay allows messages that are listed on the IBM Integration web interface to be selected and sent to the same, or a different, message flow for further processing.

In this lab, you replay the message by sending it to a separate IBM MQ queue; another application does not process it.

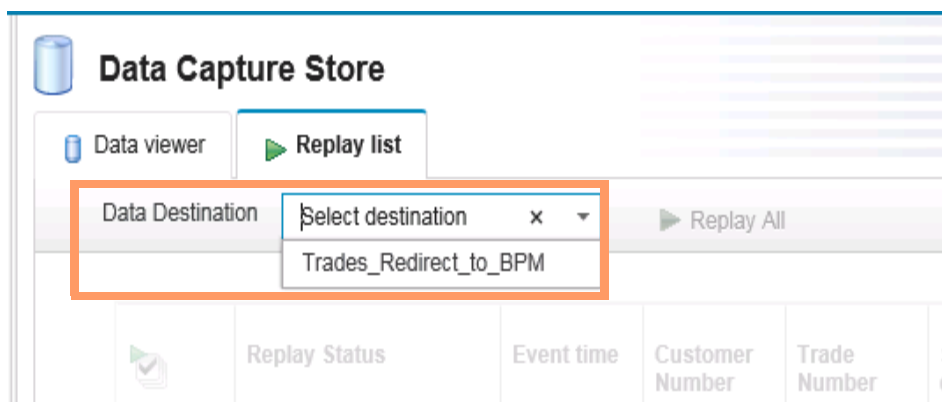
Depending on the message flow and the types of events that you want to replay, the replay queue might be the same input queue that the original message flow uses or a different queue. The replay might also be a separate (and different) message flow.

- \_\_ 1. Review the configurable service that was defined to enable the replay.
- \_\_ a. In the IBM Integration web interface, expand the **Configurable Services** folder.
  - \_\_ b. Select the **DataDestination > Trades\_Redirect\_to\_BPM** configurable service.  
 This configurable service enables messages to be routed to the TRADE.FIX.IN queue on the IIBQM.  
 In this lab, a simple message flow in the **Trades** application processes this queue. In another scenario, a business process management application might process it and amend the message before sending it to the **Trades** application again.
- \_\_ 2. In the IBM Integration web interface **Data viewer**, enable the **Data** column.
- \_\_ a. Click **Customize**,
  - \_\_ b. Enable the **hasBitstream (Data)** column.
  - \_\_ c. Click **Apply**.
- \_\_ 3. Select some of the messages for replay.
- \_\_ a. Select the check box for the message to enable it.
  - \_\_ b. Make sure that at least one of the selected messages shows the colored bitstream icon in the **Data** column.
  - \_\_ c. Click **Mark for replay**, which is now active.



- \_\_\_ 4. Clicking **Mark for replay** displays the **Replay list**. However, you still cannot start the **Replay** function. You must first select the **Data Destination**.

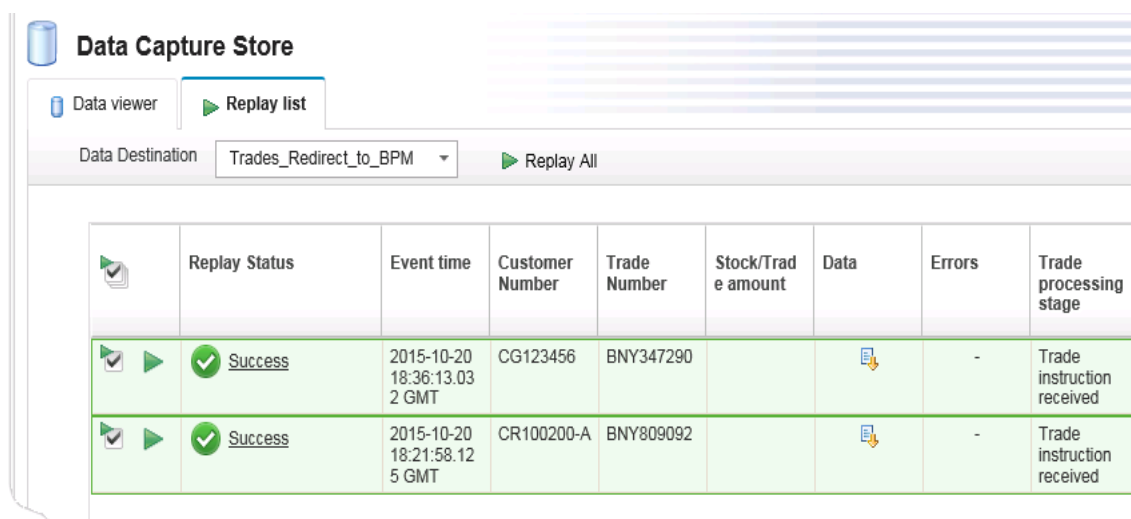
On the **Data Destination** menu, select the destination **Trades\_Redirect\_to\_BPM**.



- \_\_\_ 5. On the **Replay list** view, click **Replay All** (or you can replay each item individually by clicking the green arrow next to each message).

You should see that the item that contained the data bitstream was successfully sent to the replay destination. It does not mean that the application successfully processed data; it was successfully sent to the receiving destination.

You cannot replay items that do not contain a data bitstream.



The screenshot shows the 'Data Capture Store' interface. It has tabs for 'Data viewer' and 'Replay list'. Below the tabs, there is a 'Data Destination' dropdown set to 'Trades\_Redirect\_to\_BPM' and a 'Replay All' button. The main area displays a table with the following columns: a checkbox, 'Replay Status', 'Event time', 'Customer Number', 'Trade Number', 'Stock/Trade amount', 'Data', 'Errors', and 'Trade processing stage'.

|                                     | Replay Status | Event time                  | Customer Number | Trade Number | Stock/Trade amount | Data | Errors | Trade processing stage     |
|-------------------------------------|---------------|-----------------------------|-----------------|--------------|--------------------|------|--------|----------------------------|
| <input checked="" type="checkbox"/> | Success       | 2015-10-20 18:36:13.032 GMT | CG123456        | BNY347290    |                    |      | -      | Trade instruction received |
| <input checked="" type="checkbox"/> | Success       | 2015-10-20 18:21:58.125 GMT | CR100200-A      | BNY809092    |                    |      | -      | Trade instruction received |

- \_\_\_ 6. Confirm that the messages were sent to the replay queue.
- \_\_\_ a. Open IBM MQ Explorer.
  - \_\_\_ b. Select **Queues** under **IIBQM**.
  - \_\_\_ c. The **Current queue depth** of TRADE.FIX.OUT should increase by 2 (or the number of messages that you sent for replay).

#### Part 4: Handle failed messages

If a message flow encounters an error during processing, it can be captured and reported by using the IBM Integration web interface.

To enable this feature, the monitoring point on the message flow node must include the **\$ExceptionList** string in the monitoring event message.

- \_\_\_ 1. Using RFHUtil, send a bad message to the **Trades** application.
- \_\_\_ a. Open the file C:\labfiles\Lab12-RandR\data\TradeMessage\_BadMessage.xml
  - \_\_\_ b. Send the message to TRADE.VALIDATE.IN queue on IIBQM.

Although this message is a valid XML message, it is missing a required XML element. It fails validation on the **ReceiveTrade** node because validation is set to **Content and Value**.

- \_\_\_ 2. In the IBM Integration web interface, you should see three new entries.
- The **Trade processing stage** for the first event is *Gold customer Processing trade*.
- The **Trade processing stage** for the second event *Trade instruction received*.
- The **Trade processing stage** for the third event is *Data validation failure*.

|                          | Event time                  | Customer Number | Trade Number | Stock/Trade amount | Data | Errors | Trade processing stage          |
|--------------------------|-----------------------------|-----------------|--------------|--------------------|------|--------|---------------------------------|
| <input type="checkbox"/> | 2015-10-20 18:36:13.135 UTC | CG123456        | BNY347290    | IBM \$1000         |      | -      | Gold customer: Processing trade |
| <input type="checkbox"/> | 2015-10-20 19:19:36.936 UTC | CG123456        | BNY590012    |                    |      | -      | Trade instruction received      |
| <input type="checkbox"/> | 2015-10-20 19:19:36.937 UTC | CG123456        | BNY590012    |                    |      |        | Data validation failure         |

\_\_\_ 3. Failed events can be highlighted in the IBM Integration web interface **Data viewer** by customizing the displayed columns. Click **Customize**.

\_\_\_ a. If it is not already selected, enable the **hasException (Errors)** column.

\_\_\_ b. Click **Apply**.

The **Errors (hasException)** column shows a red cross for all monitoring events that contain the `$ExceptionList` data.

## Exercise clean up

\_\_\_ 1. In the IBM Integration web interface, stop the running application and then delete it from the **server1** integration server.

\_\_\_ 2. Close RFHUtil.

\_\_\_ 3. Close the IBM Integration web interface.

## End of exercise

## Exercise review and wrap-up

In Part 1 of this exercise, set up the environment for recording and replaying messages. Set up included building the database tables and importing the configurable services.

In Part 2 of this exercise, you viewed event messages in the IBM Integration web interface, customized the Data viewer, and defined filters to limit the data that is displayed.

In Part 3 of this exercise, you replayed messages by sending them to an alternative queue.

In Part 4 of this exercise, you configured the message flow to generate the `$ExceptionList` string and identified failed messages in the IBM Integration web interface.

Having completed this exercise, you should be able to:

- Activate message flow monitoring and store events in a database
- Enable an integration node to connect to a database with ODBC
- View event messages in the IBM Integration web user interface
- Replay a message to an IBM MQ queue by using the IBM Integration web interface

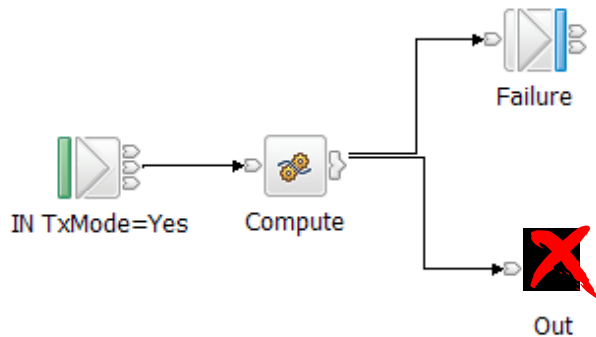


# Appendix A. Exercise solutions

## Exercise 9, Identifying runtime problems

### Part 1

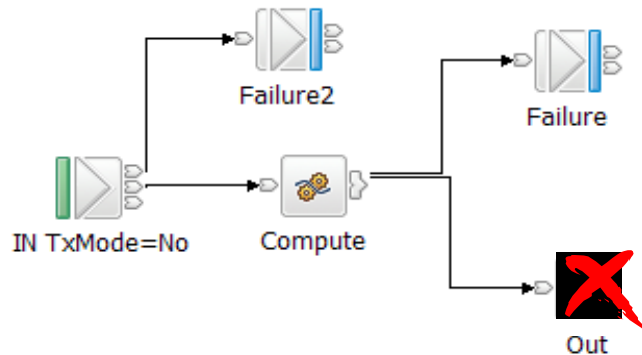
1. Queue manager dead-letter queue = DLQ; IN queue backout queue = IN\_BOQ



#### Answer:

The message is rolled back to the input queue. After rollbacks reach the backout threshold, the message is put to IN\_BOQ

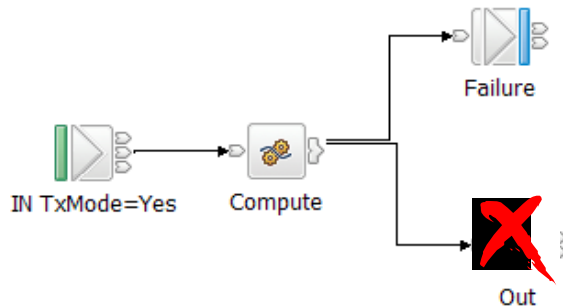
2. Queue manager dead-letter queue = DLQ; IN queue backout queue = IN\_BOQ



#### Answer:

The message is discarded because the flow is not transactional (TxMode=No) and cannot be rolled back. The MQInput node **Catch** terminal is not wired so, you cannot consider that option.

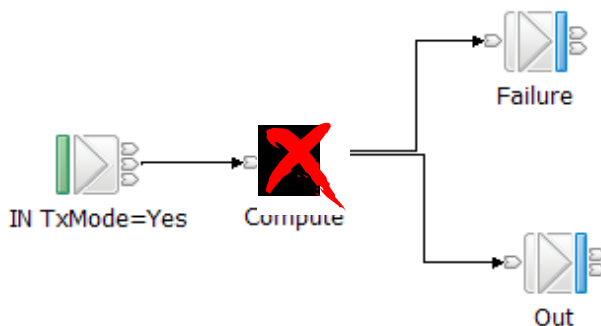
3. Queue manager dead-letter queue = DLQ; IN queue has no backout queue.



**Answer:**

The message is rolled back to the input queue. After the backout threshold is reached, the message is put to the DLQ. The MQInput node **Catch** terminal is not wired.

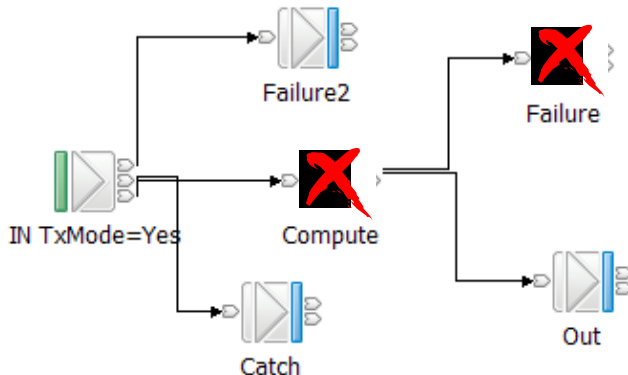
4. Queue manager dead-letter queue = DLQ; IN queue backout queue = IN\_BOQ



**Answer:**

The message is propagated through the Compute node Failure terminal. The message that is written to the Failure queue.

5. Queue manager dead-letter queue = DLQ; IN queue backout queue = IN\_BOQ

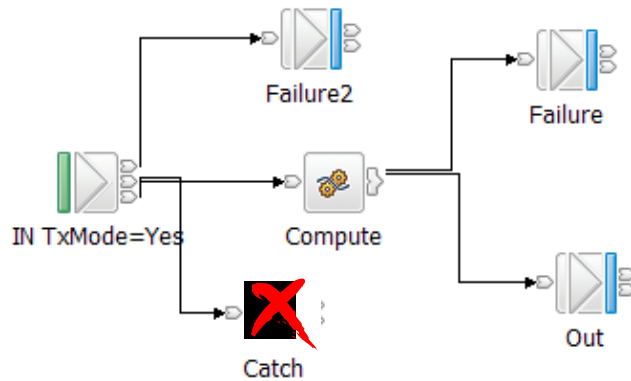


**Answer:**

The message is propagated through the Compute nodes Failure terminal; this action fails, so it is propagated to the Catch terminal and thus to the Catch queue, regardless of the transactional nature of the flow.



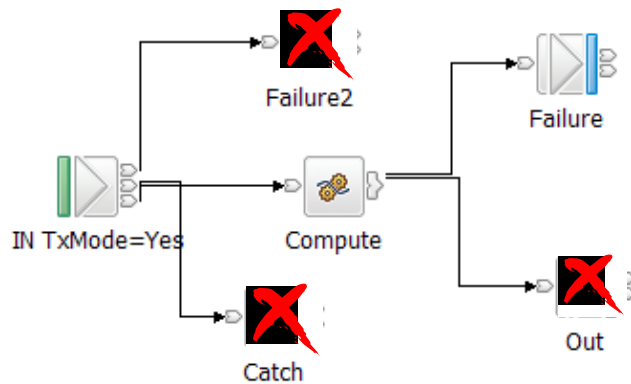
6. Queue manager dead-letter queue = DLQ; IN queue backout queue = IN\_BOQ



**Answer:**

The message is written to the Failure2 queue (regardless of transaction mode).

7. Queue manager dead-letter queue = DLQ; IN queue backout queue = IN\_BOQ



**Answer:**

The message is put onto the IN\_BOQ after 2\*BackoutThreshold retries that failed in FAILURE2. If the IN\_BOQ was full or PUT is disabled, the integration node tries to put the message onto the DLQ. If that fails as well, the message loops between the MQInput node and the Failure2 MQOutput node.

## Part 2

After submitting the test message, you should see that message did not go to the REPLY queue as expected. Instead, it went to the CATCH queue.

To determine why the message flow failed, enable Trace nodes on the integration server and then run the message flow again. The command to enable Trace nodes on **server1** is:

```
mqsichangetrace IIBNODE_WITHQM -n on -e server1
```

In the Windows Event Viewer Application log you see a Warning message:

```
( IIBNODE_WITHQM.server1 ) Operating system error ''The system cannot find
the path specified.
'' opening trace file ''C:\labfiles\TraceFiles\SimpleFlowtrace.txt'' for
trace node 'SimpleFlowWithTrace.Trace'.
```

The trace node 'SimpleFlowWithTrace.Trace' attempted to open the trace file 'C:\labfiles\TraceFiles\SimpleFlowtrace.txt' but the operating system reported the error 'The system cannot find the path specified.'.  
'. Message flow processing will continue, but trace output will not be written. The node will continually attempt to open the file until it succeeds.

If the name or location of the file was not specified correctly, then correct the message flow configuration and redeploy the integration node. If the file could not be opened due to an environment or system error, take appropriate action to correct this situation. This message indicates that the message flow contains a Trace node that is attempting to write its output to a directory that does not exist. Recall that if a Trace node is sending its output to a file, the directory path that is specified in the trace node properties must exist, or the file is not created. To correct this problem, you must either create the specified directory or modify the .bar file to change the path. If you correct this problem before correcting the next error in the log, the trace output that is written to the file provides further details on the error.

Create the folder C:\labfiles\TraceFiles and then rerun the message flow.

When you examine the properties for the Trace node that is connected to the CATCH terminal, you should see that it is writing to a User Trace, not a file.

Activate a User Trace on the message flow and then rerun the flow. The command is:

```
mqsichangetrace IIBNODE_WITHQM -e sever1 -u -f SimpleFlowWithTrace
-k SIMPLEFLOW_WITH_TRACE -l debug
```

Run the gettrace.cmd in the C:\labfiles\Tools directory to convert the trace file and open it in Notepad.

The error message in the log shows that the message flow fails with a reason code 2085 when it attempts to write to queue REPLYx, which does not exist. To correct this problem, you must define this queue to the queue manager or modify the **queueName** property for the REPLY node to a queue name that exists.

```
MessageException BIP2666E: An error occurred when opening queue ''REPLYx''
on destination queue manager ''IIBQM''. State = '-1' ''MQW101'' '2085' ''
An error occurred when opening a queue. The reason code from the MQOPEN is
displayed as the 3rd (native error) state.
```

Check the WebSphere MQ completion and reason codes in the WebSphere MQ Application Programming Reference manual to establish the cause of the error,

taking any appropriate action. It may be necessary to restart the integration node after you have performed this recovery action.

A complete copy of a sample user trace is provided in the C:\labfiles\Lab09-PD\solution directory.





