

IBM WEBSPHERE 6.1 – LAB EXERCISE

WebSphere Enterprise Service Bus V6.1 mediation programming model

Lab Four – Retry with alternate endpoints

What this exercise is about	1
Lab requirements	1
What you should be able to do	2
Introduction	2
Exercise Instructions	3
Understanding how to read the instructions	4
Part 1: Setting up the environment for the lab	5
Part 2: Authoring the mediation flow to retry using alternate endpoints	9
Part 3: Test the retry with alternate endpoints mediation flow	20
Part 4: Clean up the environment	28
What you did in this exercise	30
Solution instructions	31
Task: Adding remote server to the WebSphere Integration Developer test environment	32

What this exercise is about

The objective of this lab is to provide you with an understanding of how automated service call retry capabilities can use alternate endpoints when retrying a call.

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

Lab requirements

- WebSphere Integration Developer V6.1 installed on Windows or Linux
- WebSphere Enterprise Service Bus V6.1 or WebSphere Process Server V6.1. The server can either be the test server installed by WebSphere Integration Developer or a remote server from a separate WebSphere Enterprise Service Bus V6.1 or WebSphere Process Server V6.1 installation.

What you should be able to do

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

- Understand the service message object structure for dynamic endpoints and alternate endpoints and how they can be initialized
- Configure a service invoke mediation primitive to use alternate endpoints when performing service call retry.

Introduction

This lab exercise is the fourth of a series of four tutorials intended to illustrate the new programming model for mediations introduced by WebSphere Enterprise Service Bus V6.1. The new programming model includes message augmentation using service invoke, splitting and aggregating to handle repeating elements within a message and service call retry capabilities including the use of alternate service endpoints.

The four tutorials are described in the presentation entitled [Augmentation, aggregation and retry tutorials](#). You should familiarize yourself with the tutorials as described in the presentation before attempting this lab.

Exercise Instructions

Some instructions in this lab might be Windows operating system specific. If you plan on running WebSphere Integration Developer on a Linux operating system you will need to issue the appropriate commands and use appropriate files for Linux. The directory locations are specified in the lab instructions using symbolic references, as follows:

Reference variable	Example Windows location	Example Linux location
<WID_HOME>	C:\Program Files\IBM\WID61	/opt/IBM/WID61
<LAB_FILES>	C:\Labfiles61\WESB\61ProgModel	/tmp/Labfiles61/WESB/61ProgModel

Instructions if using a remote server for testing

Note that the previous table is relative to where you are running WebSphere Integration Developer. The following table is related to where you are running the remote test environment:

Reference variable	Example: Remote Windows test server location	Example: Remote z/OS® test server location	Input your values for the remote location of the test server
<SERVER_NAME>	server1	sssr011	
<WAS_HOME>	C:\Program Files\IBM\WebSphere\AppServer	/etc/sscell/AppServer	
<HOSTNAME>	localhost	mvsxxx.rtp.raleigh.ibm.com	
<SOAP_PORT>	8880	8880	
<TELNET_PORT>	N/A	1023	
<PROFILE_NAME>	AppSrv01	default	
<USERID>	N/A	ssadmin	
<PASSWORD>	N/A	fr1day	

Instructions for using a remote testing environment, such as z/OS, AIX® or Solaris, can be found at the end of this document, in the section "[Task: Adding remote server to the WebSphere Integration Developer test environment](#)".

Understanding how to read the instructions

In this lab, the instructions are written to allow an experienced user to complete the steps easily while at the same time providing very explicit instructions needed by the new user. The format of the instructions follows this pattern:

- ___ 1. This is a sentence or short paragraph that describes a particular task to be completed. In some cases this might be sufficient for an experienced user, but in other cases the experienced user might require some additional specific information to complete the task. In that case, there is a bulleted list that helps the experience user know specifics.
 - **Additional information for experienced user**
 - **This information, along with the above paragraph, should allow the experienced user to complete the task**
 - -----
 - ___ a. First step needed by the new user
 - ___ b. Second step needed by the new user
 - 1) Additional details for completing this step
 - 2) More details for completing this step
 - ___ c. Third step needed by new user
- ___ 2. Next task to be completed
 - **Info for experience user**
 - -----
 - ___ a. First step needed by the new user
 - ___ b. Second step needed by the new user.

Part 1: Setting up the environment for the lab

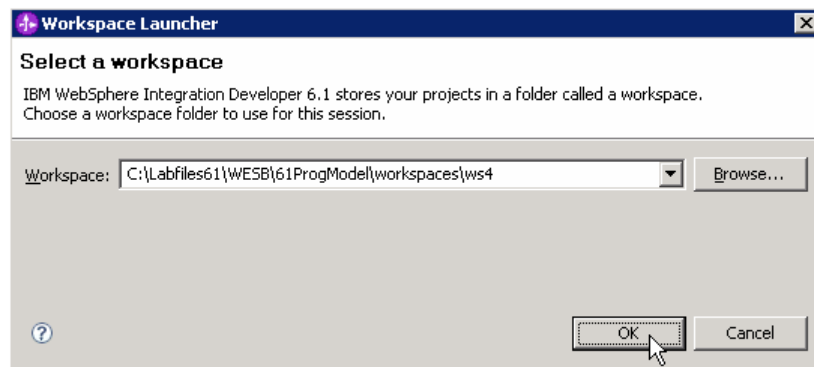
What you will do in this part: In this part you are getting the environment set up to do the lab. There are three different ways you might be approaching this which will dictate what you need to do.

(1) Proceeding from Lab Three – You are directly continuing from Lab Three and you did not shut down the server and development environment. In this case, there is nothing that needs to be done in this part.

(2) Restarting from Lab Three – You are continuing from Lab Three which you did previously and therefore you did shut down the server and development environment. In this case, you will need to restart the development environment and server but will not need to import a project interchange to initialize the workspace.

(3) Directly starting Lab Four – You are starting this lab from scratch, irrespective of whether you had previously completed Lab Three.

- ___ 1. If you are **proceeding from Lab Three** there is nothing to do, skip directly to **Part 2 Authoring the mediation flow to retry using alternate endpoints**
 - -----
- ___ 2. Start WebSphere Integration Developer.
 - **Restarting from Lab Three** : Use the same workspace used in Lab 3
 - **Directly starting Lab 4**: Suggested location: <LAB_FILES>/workspaces/ws4
 - -----
 - ___ a. Select **Start** → **All Programs** → **IBM WebSphere Integration Developer** → **IBM WebSphere Integration Developer V6.1** → **WebSphere Integration Developer V6.1**
 - ___ b. From the Workspace Launcher window, enter the name of the workspace in the Workspace field
 - 1) Restarting from Lab 3: Use the same workspace used in Lab 3
 - 2) Directly starting Lab 4: <LAB_FILES>/workspaces/ws4



- ___ c. If the Welcome panel is displayed, click on **Go to the Business Integration perspective** or the arrow next to it in the upper right corner of the panel.



___ 3. If you are **restarting from Lab Three** there is nothing additional to do, skip directly to **Part 2 Authoring the mediation flow to recover from a modeled fault**

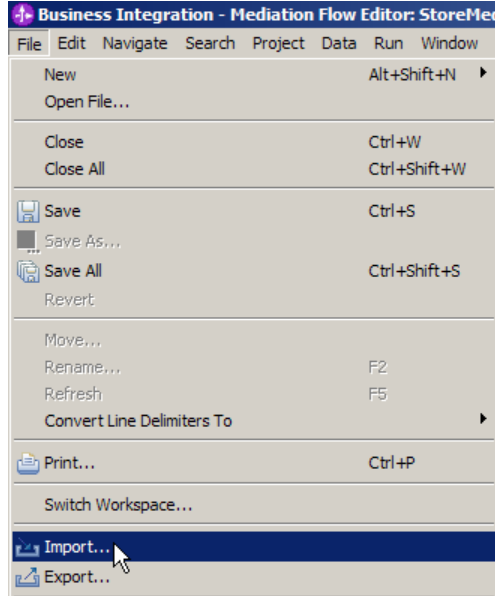
- -----

___ 4. If you are **directly starting Lab Four** import the project interchange file containing the starting point for the lab

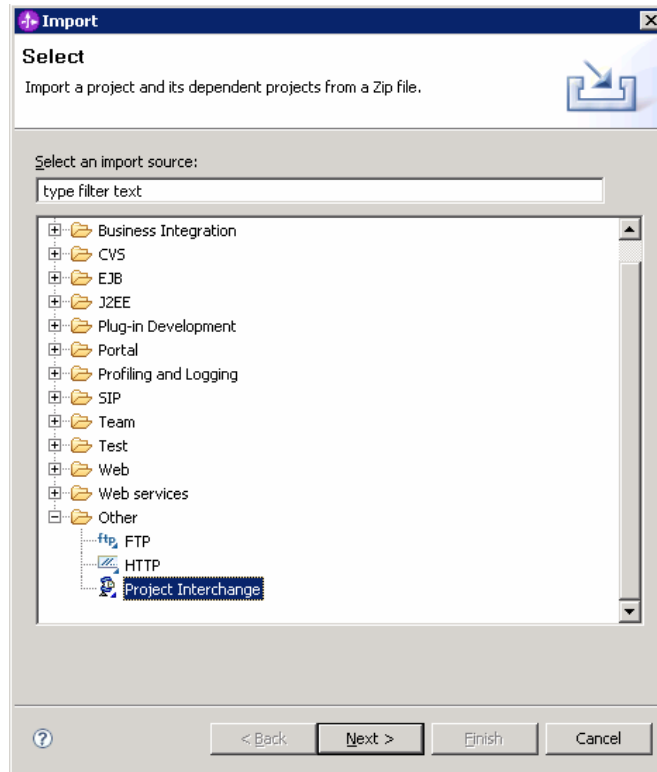
- <LAB_FILES>/PI4-RetrySolution-AlternateEndpointsStart.zip

- -----

___ a. From the menu, select **File → Import...**

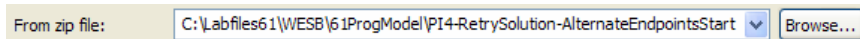


__ b. In the **Import** dialog, select **Other → Project Interchange**

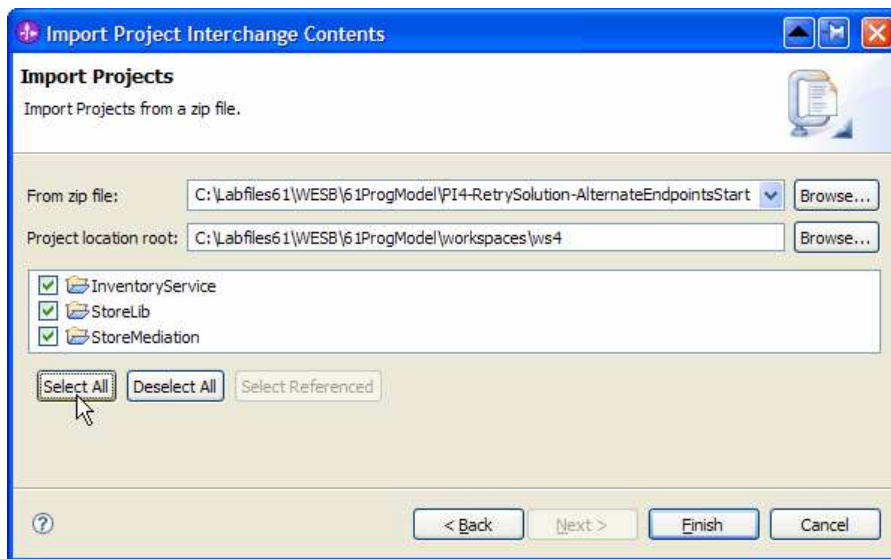


__ c. Click **Next**

__ d. In the **Import Project Interchange Contents** dialog set the **From zip file:** value to **<LAB_FILES>/PI4-RetrySolution-AlternateEndpointsStart.zip**

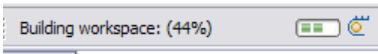


__ e. Click **Select All** to selected all of the projects listed



___ f. Click **Finish**

___ g. Wait for the build process to complete for the imported projects, which is seen at the lower right corner of WebSphere Integration Developer.

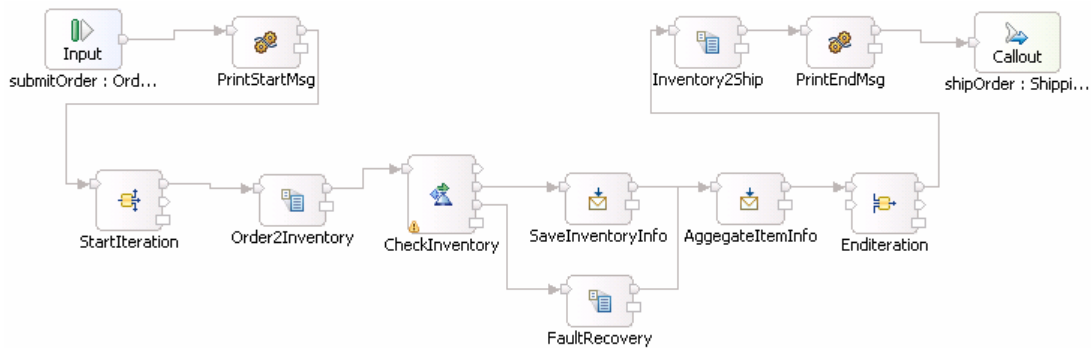


Part 2: Authoring the mediation flow to retry using alternate endpoints

What you will do in this part: In this part you will modify the flow so that it initializes the SMO header fields for dynamic target address and alternate endpoints. Then you will modify the CheckInventory service invoke primitive to use the dynamic target address (rather than what is wired on the assembly diagram) and to use the alternate target addresses during service invoke retry processing.

You should see the presentation entitled [Augmentation, aggregation and retry tutorials](#) to better understand what this part is doing.

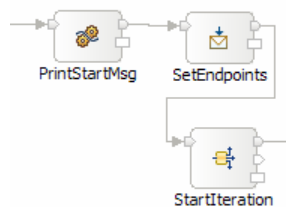
1. Open the StoreMediation flow found in the StoreMediation module.



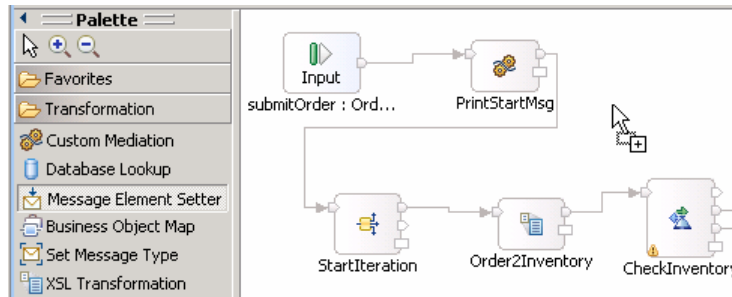
- a. In the Business Integration view, expand **StoreMediation** → **Mediation Logic** → **Flows** and then double-click on **StoreMediation** to open it in the mediation flow editor
- b. The line connecting **Ordering** → **submitOrder** and **ShippingPartner** → **shipOrder** should be selected for you to view the flow

2. Add a message element setter primitive and rewire the flow so that it is in between the StartPrintMsg primitive and the StartIteration primitive. This is used to initialize the target endpoint and alternate endpoints. In a typical production scenario, this is most likely done with an endpoint lookup primitive interfacing with WebSphere Service Registry and Repository.

- **Display Name:** SetEndpoints
- **Wire** : PrintStartMsg to SetEndpoints
- **Wire** : SetEndpoints to StartIteration

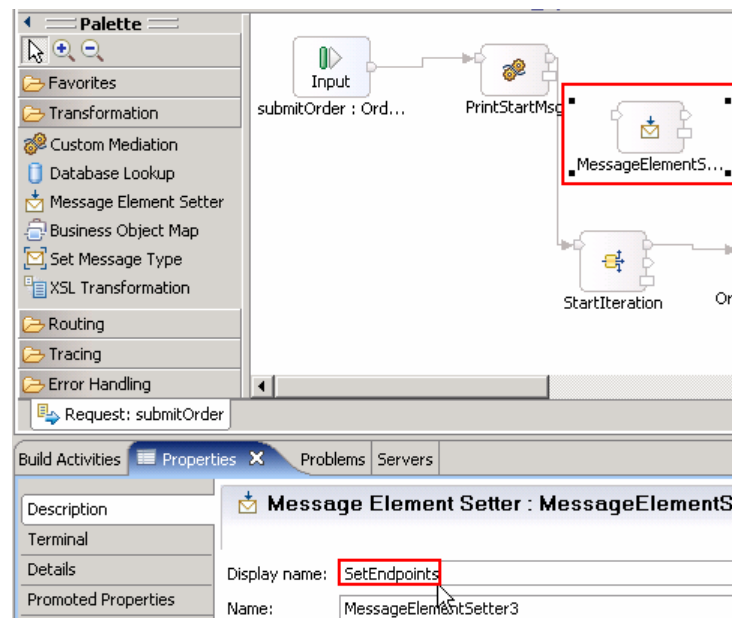


- ___ a. From the **Palette**, select **Transformation → Message Element Setter** and then click on the canvas as shown below:



- ___ b. You will now see a new message element setter primitive added to the flow. Ensure that this primitive is highlighted and select **Properties → Description**

- ___ c. Change the **Display name** to **SetEndpoints**

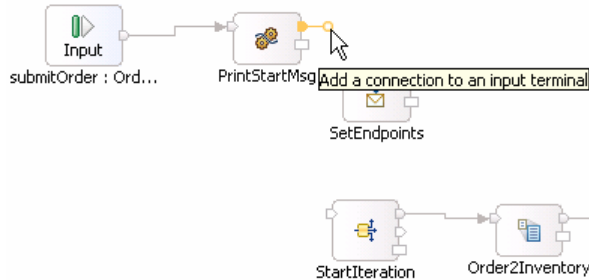


- ___ d. Remove connection between **PrintStartMsg** and **StartIteration** primitives

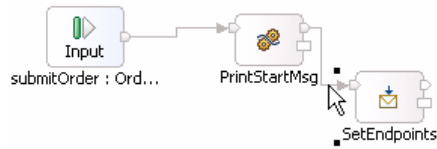
- 1) Right-click on the line connecting **PrintStartMsg** and **StartIteration** primitives and select **Delete** from the pop-up menu

___ e. Add a connection from **PrintStartMsg** to **SetEndpoints** primitive

- 1) Hover your mouse over **out** terminal of **PrintStartMsg** to get the 'Add a connection to an input terminal'

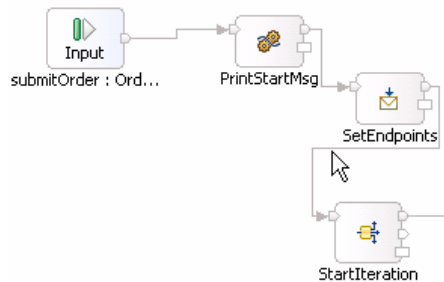


- 2) Click on the orange bubble and then click on **SetEndpoints** primitive
- 3) You should now see a connection from **PrintStartMsg** to **SetEndpoints** primitive



___ f. Similarly add a connection from **SetEndpoints** to **StartIteration** primitive

- 1) Hover your mouse over **out** terminal of **SetEndpoints** to get the 'Add a connection to an input terminal'
- 2) Click on the orange bubble and then click on **StartIteration** primitive
- 3) You should now see a connection from **SetEndpoints** to **StartIteration** primitive



- ___ 3. Configure the **SetEndpoints** message element setter primitive to set up the target address and alternate target addresses. To illustrate retry with alternate endpoints, the target address is set to the endpoint that always fails and thus a retry will always occur. Then the alternate addresses are set to the endpoints that randomly fail and always work, in that order. So in some cases, the first alternate endpoint will work and in other cases it will fail, but the second alternate endpoint will always work.
 - **Set properties of message element setter to assign string values as follows**

Target location in SMO	String value
/headers/SMOHeader/Target/address	sca://InventoryService/InvFails
/headers/SMOHeader/AlternateTarget[1]/address	sca://InventoryService/InvRandom
/headers/SMOHeader/AlternateTarget[2]/address	sca://InventoryService/InvWorks

Message Elements:

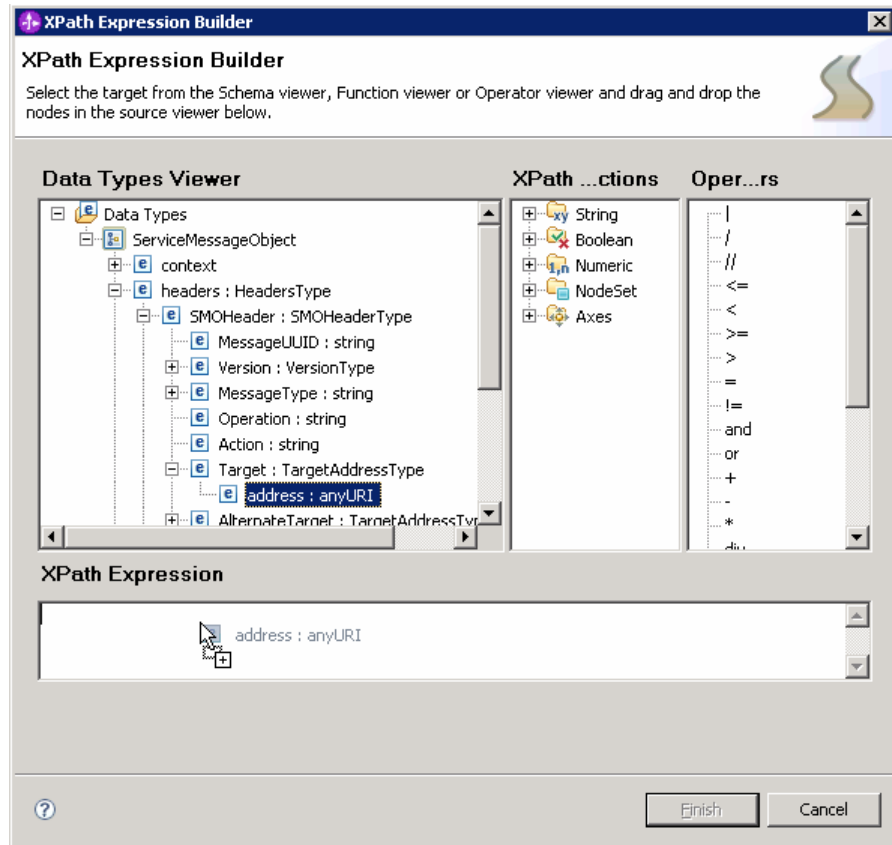
Target	Type	Value
/headers/SMOHeader/Target/address	String	sca://InventoryService/InvFails
/headers/SMOHeader/AlternateTarget[1]/address	String	sca://InventoryService/InvRandom
/headers/SMOHeader/AlternateTarget[2]/address	String	sca://InventoryService/InvWorks

Note: Ensure that the setting of alternate target address[1] comes before the setting of alternate target address[2] in the table. If not, use the up or down icons to rearrange the table rows so that they match the screen capture.

• -----

___ a. Set /headers/SMOHeader/Target/address to the string URL value sca://InventoryService/InvFails

- 1) Ensure that the **SetEndpoints** primitive is highlighted and then select **Properties → Details**
- 2) Under Message Elements table, click the **Add...** button. The Add/Edit window is opened
- 3) Define Target:
 - a) Click **Edit...** next to **Target**. The XPath Expression Builder wizard is opened
 - b) Expand **ServiceMessageObject → headers → SMOHeader → Target**
 - c) Drag **address** and drop it under **XPath Expression** (or double-click on address)



d) You should now see this expression under **XPath Expression**

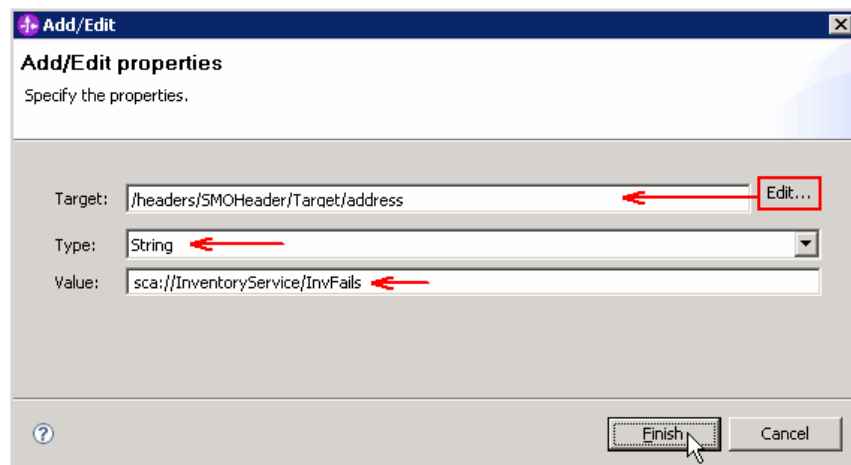


e) Click **Finish** button. You are now back to Add/Edit window

4) For **Type**, select **String** (default) from the drop down list

5) For **Value**, enter **sca://InventoryService/InvFails**

6) Your Add/Edit window should look like this:



7) Click **Finish**

8) You will see this entry in the Message Elements table:

Message Elements:

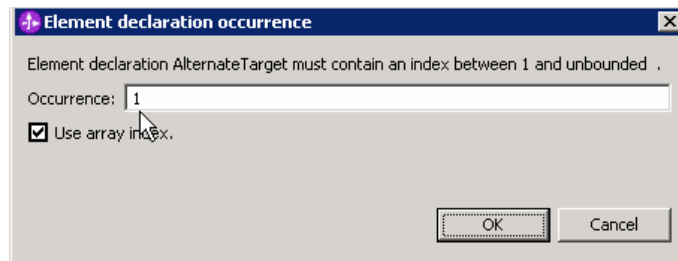
Target	Type	Value	
/headers/SMOHeader/Target/address	String	sca://InventoryService/InvFails	Add...
			Edit...

___ b. Similarly, set /headers/SMOHeader/AlternateTarget[1]/address to the string URL value sca://InventoryService/InvRandom

1) Under Message Elements table, click the **Add...** button. The Add/Edit window is opened

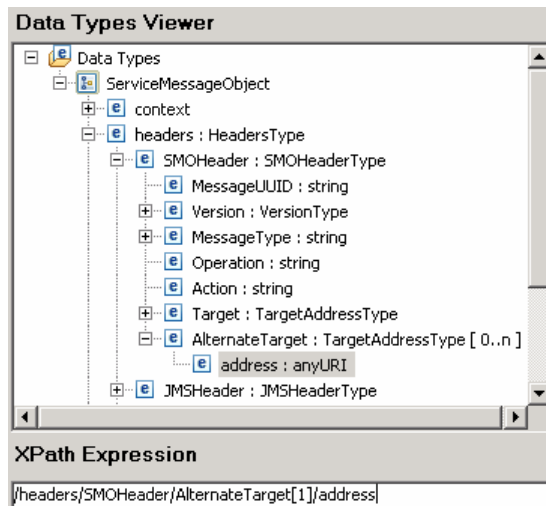
2) Define Target:

- a) Click **Edit...** next to **Target**. The XPath Expression Builder wizard is opened
- b) Expand **ServiceMessageObject** → **headers** → **SMOHeader** → **AlternateTarget**
- c) Drag **address** and drop it under **XPath Expression** (or double-click on address)
- d) An Element declaration occurrence window is opened. Enter **'1'** for **Occurrence**



e) Click **OK**

f) You should now see this expression under **XPath Expression**



g) Click **Finish** button. You are now back to Add/Edit window

3) For **Type**, select **String** (default) from the drop down list

4) For **Value**, enter `sca://InventoryService/InvRandom`

5) Your Add/Edit window should look like this:

6) Click **Finish**

7) You will see the second entry in the Message Elements table:

Message Elements:

Target	Type	Value	
/headers/SMOHeader/Target/address	String	sca://InventoryService/InvFails	Add...
/headers/SMOHeader/AlternateTarget[1]/address	String	sca://InventoryService/InvRandom	Edit...
			Remove

___ c. Finally, `/headers/SMOHeader/AlternateTarget[2]/address` to the string URL value `sca://InventoryService/InvWorks`

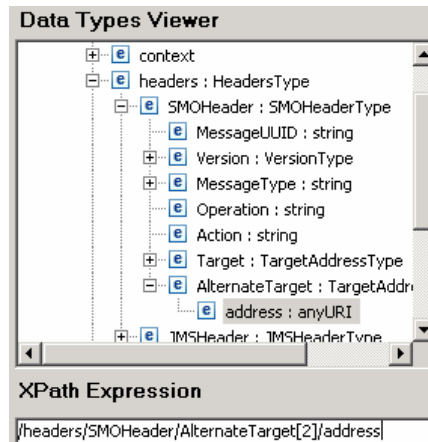
1) Under Message Elements table, click the **Add...** button. The Add/Edit window is opened

2) Define Target:

- a) Click **Edit...** next to **Target**. The XPath Expression Builder wizard is opened
- b) Expand **ServiceMessageObject** → **headers** → **SMOHeader** → **AlternateTarget**
- c) Drag **address** and drop it under **XPath Expression** (or double-click on address)
- d) An Element declaration occurrence window is opened. Enter **'2'** for **Occurrence**

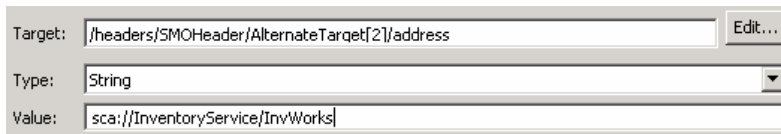
e) Click **OK**

f) You should now see this expression under **XPath Expression**



g) Click **Finish** button. You are now back to Add/Edit window

- 3) For **Type**, select **String** (default) from the drop down list
- 4) For **Value**, enter `sca://InventoryService/InvWorks`
- 5) Your Add/Edit window should look like this:



- 6) Click **Finish**
- 7) You will now see the third entry in the Message Elements table:

Message Elements:

Target	Type	Value	
/headers/SMOHeader/Target/address	String	sca://InventoryService/InvFails	Add...
/headers/SMOHeader/AlternateTarget[1]/address	String	sca://InventoryService/InvRandom	Edit...
/headers/SMOHeader/AlternateTarget[2]/address	String	sca://InventoryService/InvWorks	Remove

___ d. Sometimes the Message Elements table does not enter the element settings in the same order they were created. For example, it might look like this with `AlternateTarget[2]` listing before `AlternateTarget[1]`. This will fail at runtime as the array elements need to be created in order. To fix this, perform these steps:

Message Elements:

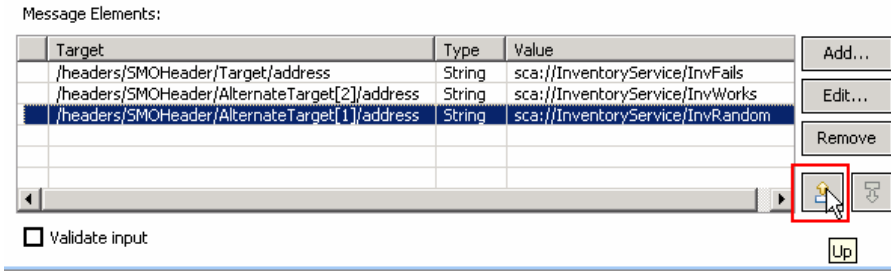
Target	Type	Value	
/headers/SMOHeader/Target/address	String	sca://InventoryService/InvFails	Add...
/headers/SMOHeader/AlternateTarget[2]/address	String	sca://InventoryService/InvWorks	Edit...
/headers/SMOHeader/AlternateTarget[1]/address	String	sca://InventoryService/InvRandom	Remove

1) Select the row containing `AlternateTarget[1]` so that it is highlighted in dark blue.

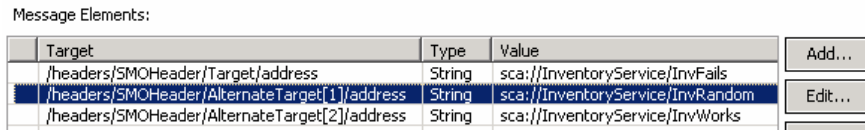
Message Elements:

Target	Type	Value	
/headers/SMOHeader/Target/address	String	sca://InventoryService/InvFails	Add...
/headers/SMOHeader/AlternateTarget[2]/address	String	sca://InventoryService/InvWorks	Edit...
/headers/SMOHeader/AlternateTarget[1]/address	String	sca://InventoryService/InvRandom	Remove

2) Use the up arrow to move the selected row up the table.

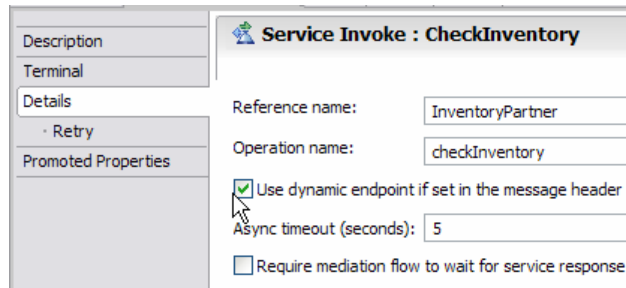


3) Check that your resulting table looks like this.

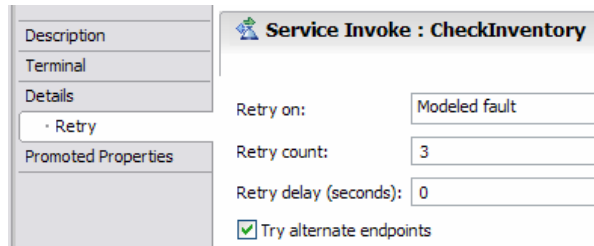


4. Modify the CheckInventory service invoke primitive to take the endpoint URL from the target address field and to perform retries using the endpoint URLs from the alternate target address array.

- **Dynamic endpoint:** Check "Use dynamic endpoint if set in message header" option on Details panel

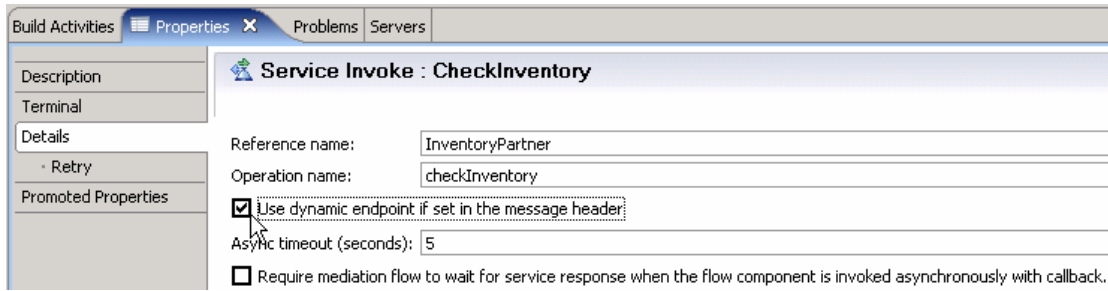


- **Alternate endpoints:** Check "Try alternate endpoints" option on Retry panel



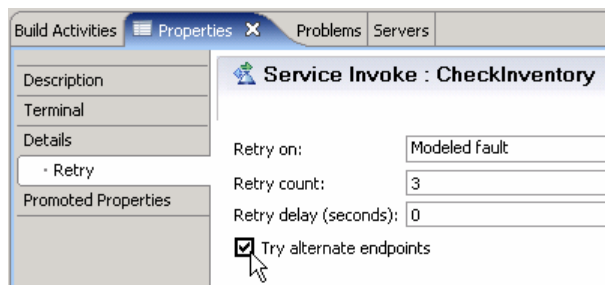
 a. From the mediation flow, select CheckInventory service invoke and then select **Properties** → **Details**

___ b. Check the box for 'Use dynamic endpoint if set in the message header'

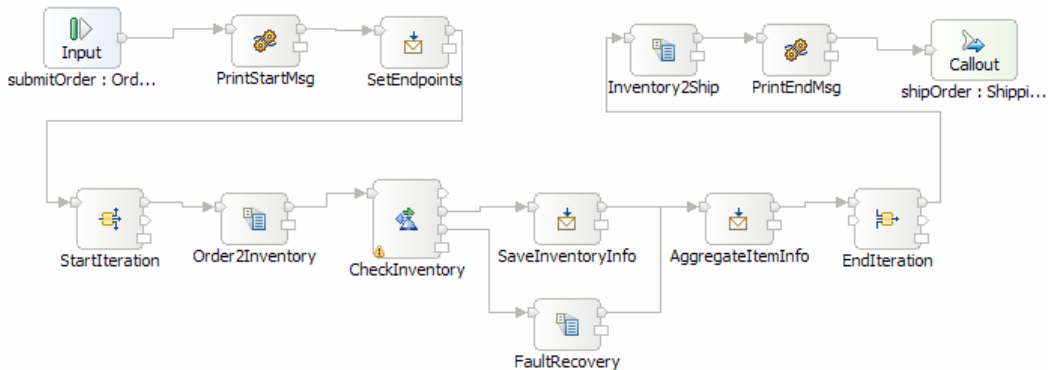


___ c. Select **Retry** under Details

___ d. Check the box for 'Try alternate endpoints'



___ 5. Ensure that the flow looks similar to this.

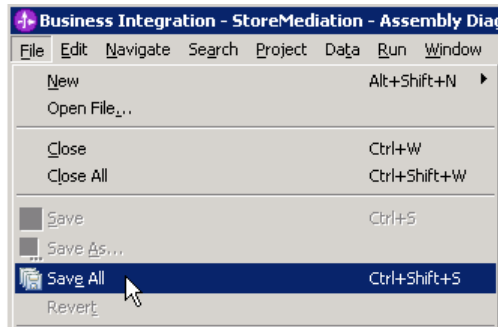


- -----

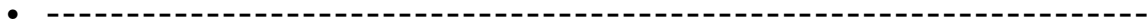
___ 6. Check that all artifacts have been saved.

- -----

___ a. From the menu select **File** → **Save All** to save your changes



___ 7. Check that there are no problems reported in the Problems view



___ a. Select **Problems** view at the bottom. You can ignore warnings, but there should not be any errors at this time:


Build Activities Properties Problems Servers			
0 errors, 5 warnings, 0 infos			
Description ^	Resource	Path	Location
Warnings (5 items)			
⚠ The InventoryPartner reference for the	StoreMediati...	StoreMediation	Unknown
⚠ The method getMyService() from the ty	InventoryFa...	InventoryService/inventory	line 23
⚠ The method getMyService() from the ty	InventoryR...	InventoryService/inventory	line 27
⚠ The method getMyService() from the ty	InventoryW...	InventoryService/inventory	line 23
⚠ The method getMyService() from the ty	ShippingJav...	StoreMediation/shipping	line 24

Part 3: Test the retry with alternate endpoints mediation flow

What you will do in this part: In this part you use the component test facilities of WebSphere Integration Developer to test the mediation. The resulting output is explained.

You should see the presentation entitled [Augmentation, aggregation and retry tutorials](#) to better understand what this part is doing.

- ___ 1. If not already running, start the WebSphere Enterprise Service Bus (or WebSphere Process Server) test server.

- -----
- ___ a. From the **Servers** view of WebSphere Integration Developer, select **WebSphere ESB Server v6.1** and hit 'Start the server' icon () from the toolbar



- ___ b. Wait until the server Status shows as **Started**

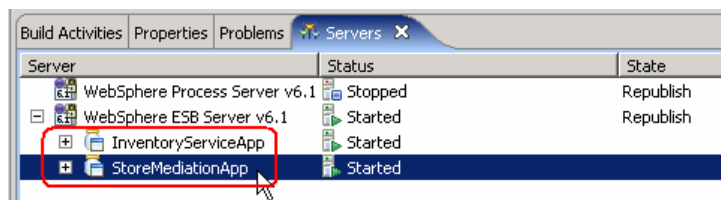


NOTE: Depending upon the preferences you have specified for the **Console** view, the **Console** view might grab the focus from the **Servers** view. If this is the case, the status in the lower right should indicate when the server startup is complete, at which time you can switch back to the **Servers** view

- ___ 2. Check to see if the InventoryServiceApp and StoreMediationApp are deployed on the server.

- **No, not deployed yet:** **Add both projects to the server**
- **Yes, already deployed:** **Restart the StoreMediationApp**
- -----

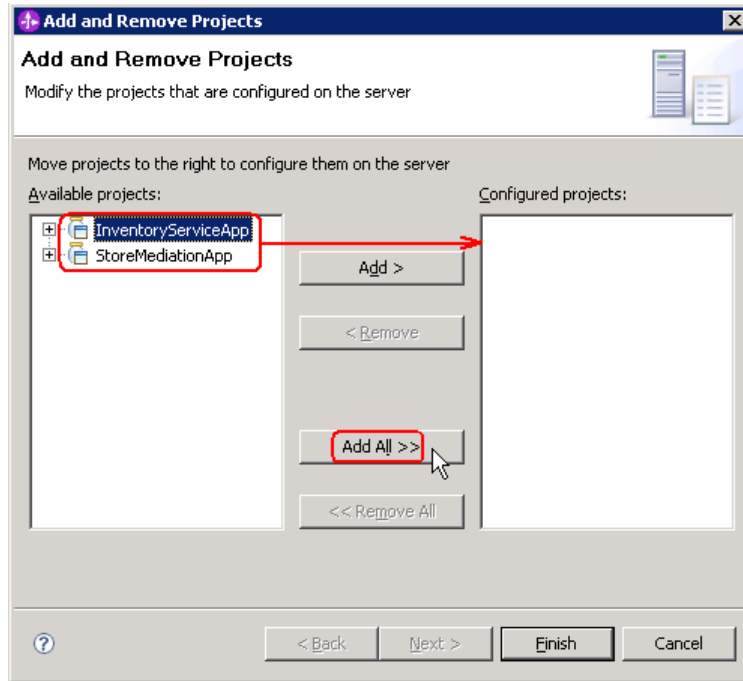
- ___ a. Check the Servers view to see if the InventoryServiceApp and StoreMediationApp are already deployed. If they are, they will appear below the server as shown here. **Follow the appropriate instructions in step b or step c**



- ___ b. If the applications are **not deployed yet**, add both projects to the server following these steps

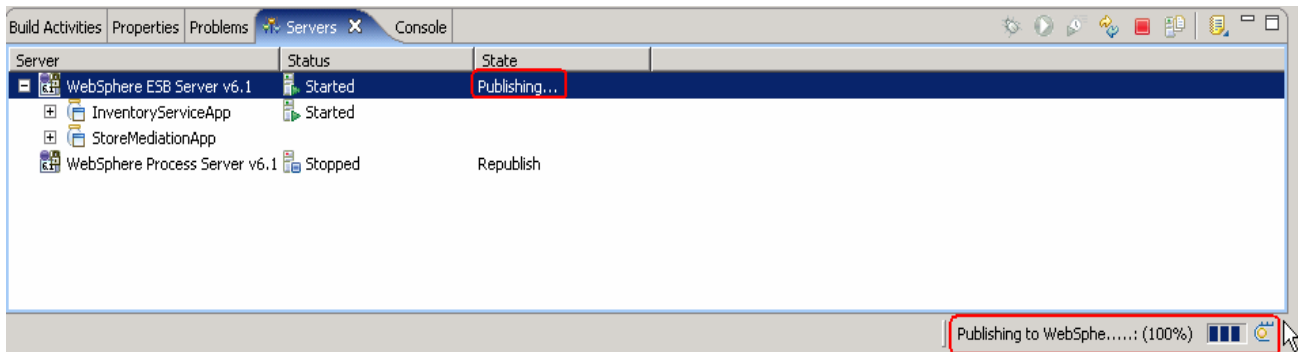
- 1) Right-click on **WebSphere ESB Server v6.1** under the Servers view and select **Add and remove projects...** from the context menu

- 2) In the Add and Remove Projects window, click **Add All >>** to add InventoryServiceApp and StoreMediationApp to the Configured projects panel

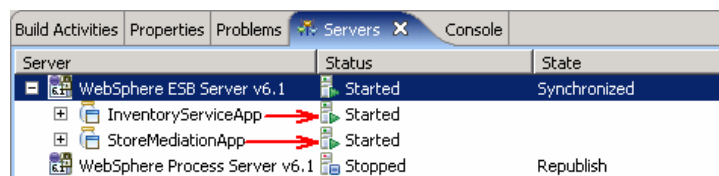


- 3) The projects will now be moved to Configured projects. Click **Finish**

- 4) Wait while the projects are being published to the server

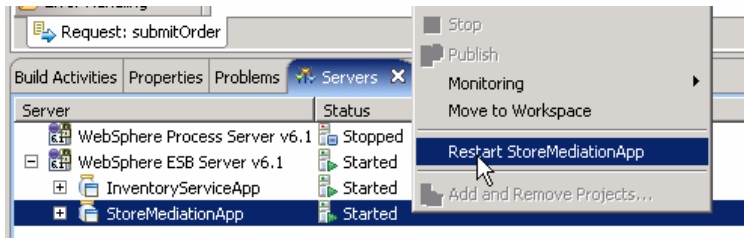


- 5) Once the publishing is done, from the Servers view, expand **WebSphere ESB Server v6.1** and you should see the 2 applications started as following:



___ c. If the applications are **already deployed**, restart the StoreMediationApp following these steps:

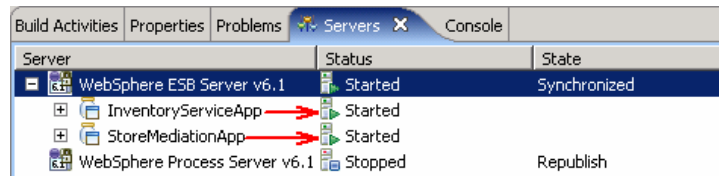
- 1) Right click on StoreMediationApp to get the pop-up menu and select **Restart StoreMediationApp**




- 2) Wait while the application stops and restarts.



- 3) Once it has restarted, you should see the two applications started as shown here:



___ 3. Check if the StoreMediation_Test panel is still present from a previous lab.

- **No:** From the assembly diagram for StoreMediation, start Test Component for the StoreMediation component.
- **Yes:** Hit the Invoke icon () in the Events panel
- -----

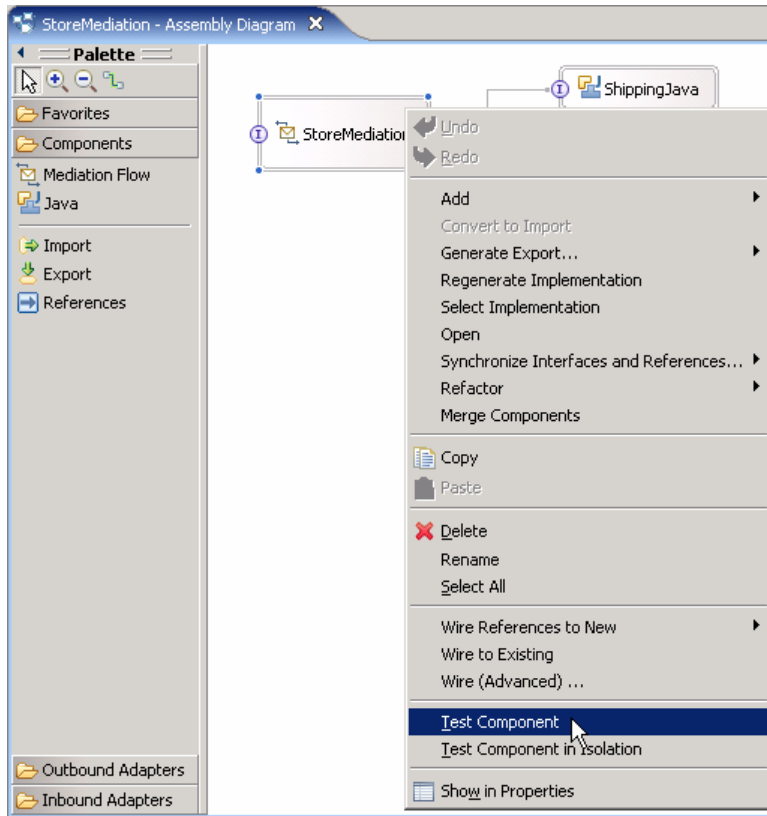
___ a. Look at the tabs to see if StoreMediation_Test is still open as shown in this screen capture. **Follow the appropriate instructions in step b or step c**




___ b. **No, it is not open.** From the assembly diagram for StoreMediation, start Test Component for the StoreMediation component

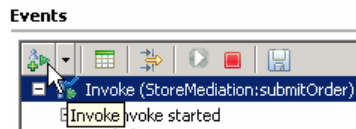
- 1) In the Business Integration window, expand **StoreMediation** and double-click on **Assembly Diagram** to open it in Assembly editor

- 2) From the StoreMediation-Assembly Diagram, right-click on **StoreMediation** component and select **Test Component** from the pop-up menu



- 3) The **StoreMediation_Test** window is opened where you will enter your test data

___ c. **Yes, it is open.** Hit the Invoke icon () in the Events panel.

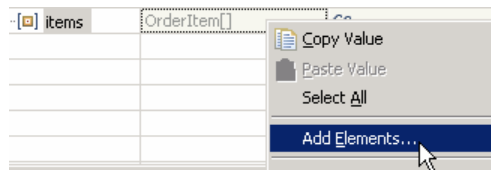


- ___ 4. Initialize the test data
- Set `customerID` to `cust123`
 - Set `items/items[0]/itemID` to `item001`
 - Set `items/items[0]/quantity` to `3`
 - Set `items/items[1]/itemID` to `item009`
 - Set `items/items[1]/quantity` to `5`
 - Set `items/items[2]/itemID` to `item002`
 - Set `items/items[2]/quantity` to `15`

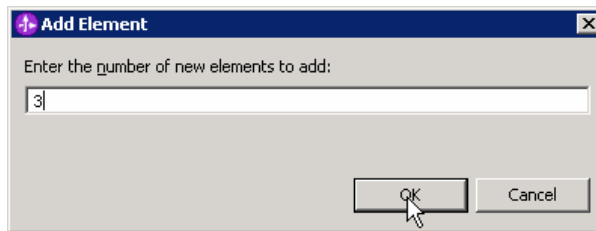
Name	Type	Value
order	Order	✓
customerID	string	✓ cust123
items	OrderItem[]	6
items[0]	OrderItem	✓
itemID	string	✓ item001
quantity	int	✓ 3
items[1]	OrderItem	✓
itemID	string	✓ item009
quantity	int	✓ 5
items[2]	OrderItem	✓
itemID	string	✓ item002
quantity	int	✓ 15

___ a. Enter this under Initial request parameters table:

- 1) For **customerID**, click under Value and enter **cust123**
- 2) Right-click any where on the row containing **items** and select **Add Elements...** from the pop-up menu




3) Enter '3' in the Add Element window:

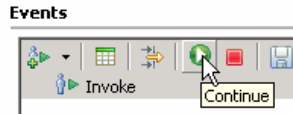


- 4) Click **OK**
- 5) Enter values for items[0]:
 - a) For **itemID**, click under Value and enter **item001**
 - b) For **quantity**, click under Value and enter **3**
- 6) Enter values for items[1]:
 - a) For **itemID**, click under Value and enter **item009**
 - b) For **quantity**, click under Value and enter **5**
- 7) Enter values for items[2]:
 - a) For **itemID**, click under Value and enter **item002**
 - b) For **quantity**, click under Value and enter **15**

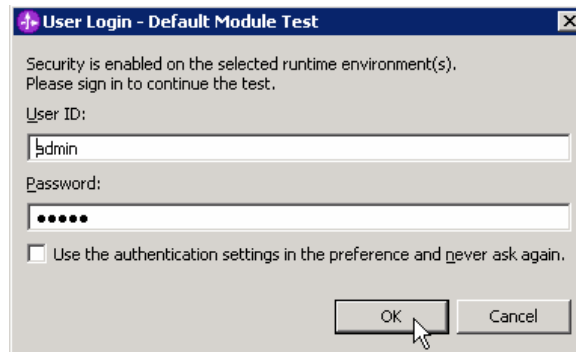
___ 5. Run the test by hitting the Continue icon ()..

• -----

___ a. Click **Continue** icon () under Events panel

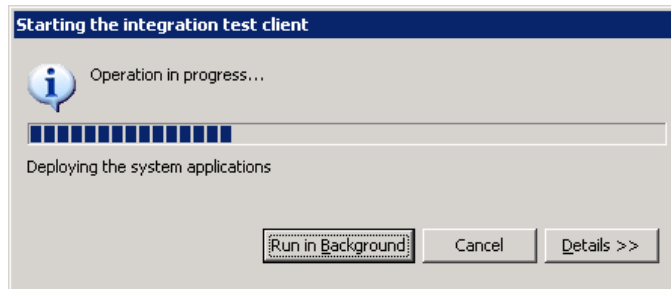


___ b. If presented with the **User Login** dialog, enter the **User ID** and **Password**, which by default is normally set to **admin** and **admin**. Optionally, you can select the 'Use the authentication settings in the preference and never ask again' check box to prevent this dialog from being displayed in the future

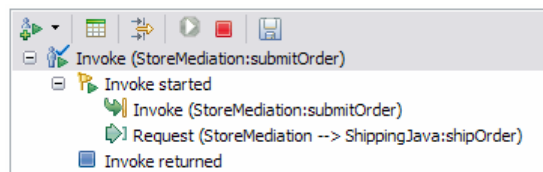


___ c. Click **OK**

___ d. Wait until the integration test client starts



___ 6. The result of running should look like this. Notice that, unlike in the previous labs, there are no calls shown to the inventory service. That is because test component only shows request/response flows that are explicitly wired in the SCA assembly. Since the calls to the inventory service in this case used dynamic addresses, the calls do not show



• -----

7. Switch to the Console view and examine the output, which should look similar to this screen capture. You will see several calls to the inventory service. Notice that there is a mixture of calls to the different inventory service implementations (InvFails, InvRandom and InvWorks) because of the use of alternate endpoints. Your screen capture might be slightly different because of the use of the InvRandom implementation of the inventory service. The target address was set to InvFails so the first attempt to access the inventory service for each item should show an exception. The first alternate address was set to InvRandom implementation, so the second call for each item will vary, with some working and others not. In the cases where the second call failed, the third call for the item is OK because the second alternate address was set to the InvWorks implementation. The final ship object sent to the shipping service should contain the three items, each initialized with its appropriate inventory information as the call to the inventory service eventually was successful for each item.

```

O *****
O ***** START mediation flow *****
O ***
O ***** InvFails - EXCEPTION: InventoryFault for itemID = item001
O ***** InvRandom - returning InventoryItem for itemID = item001
O ***** InvFails - EXCEPTION: InventoryFault for itemID = item009
O ***** InvRandom - returning InventoryItem for itemID = item009
O ***** InvFails - EXCEPTION: InventoryFault for itemID = item002
O ***** InvRandom - EXCEPTION: InventoryFault for itemID = item002
O ***** InvWorks - returning InventoryItem for itemID = item002
O ***
O ***** END mediation flow *****
O *****
O -----
O -- Ship object dump begins -----
O
O EObject: com.ibm.ws.bo.bomodel.impl.DynamicBusinessObjectImpl@544c544c
O Value:
O   customerID = cust123
O   items = ShipItem[3]
O     items[0] = <ShipItem@549a549a>
O       itemID = item001
O       orderQuantity = 3
O       inventoryQuantity = 5
O       inventoryStatus = OK - but stock is running low
O     items[1] = <ShipItem@54b454b4>
O       itemID = item009
O       orderQuantity = 5
O       inventoryQuantity = 45
O       inventoryStatus = OK - sufficient stock levels
O     items[2] = <ShipItem@54ce54ce>
O       itemID = item002
O       orderQuantity = 15
O       inventoryQuantity = 10
O       inventoryStatus = Backorder - insufficient stock to fill order
O
O -- Ship object dump ends -----
O -----

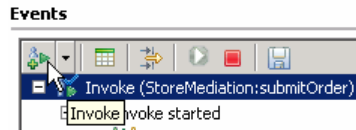
```

- a. Double click on **Console** view next to Servers view to see the above message (double clicking will maximize the Console view).



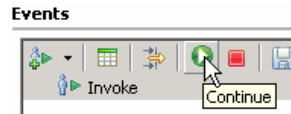
- ___ 8. If you like, you can run additional tests using different input data to see how the output varies. Key things to note about the data you use:
- `customerID` can be any string and should not have any particular affect on the results
 - The `items` array can have any number of elements
 - `itemID` values of "item001" through "item010" are recognized by the inventory service, all other values are not recognized
 - Inventory status will change according to the relationship between the order and inventory quantities
 - -----

___ a. Click **Invoke** icon () under Events panel



___ b. Enter values for **customerID**, **itemID**, and **quantity** as per the above instructions

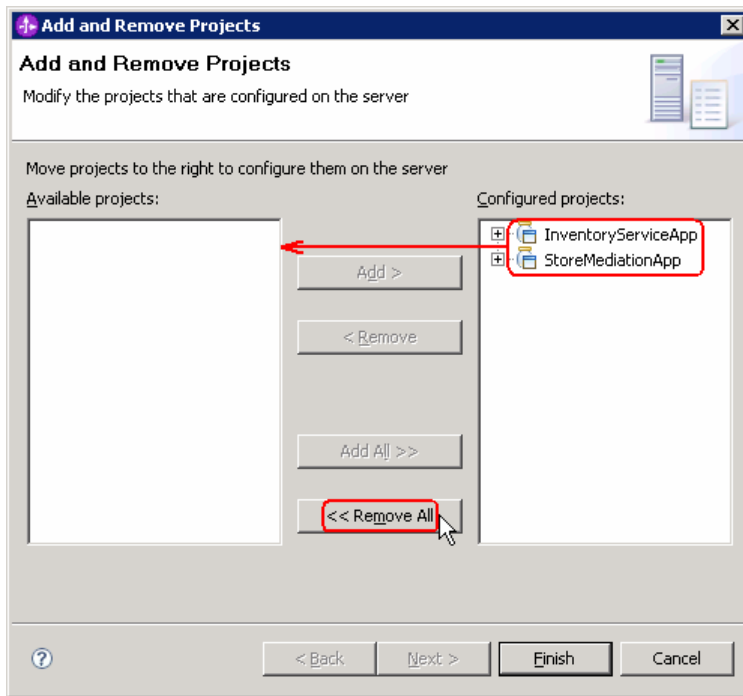
___ c. Click **Continue** icon () under Events panel



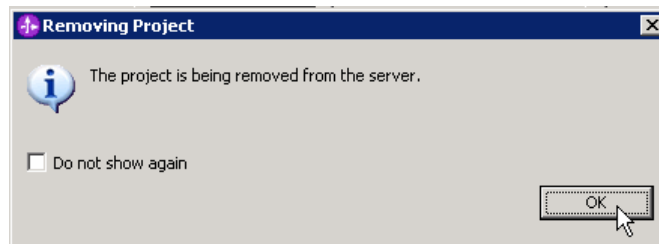
Part 4: Clean up the environment

What you will do in this part: Since this is the last lab in this sequence of labs, you should perform this part to clean up your development environment. This will leave your test server in a known state without any of the projects deployed on it.

- ___ 1. Remove the InventoryServiceApp and StoreMediationApp from the test server.
 - -----
 - ___ a. Right-click on **WebSphere ESB Server v6.1** under the Servers view and select **Add and remove projects...** from the context menu
 - ___ b. From the Add and Remove Projects window, click **<< Remove All**



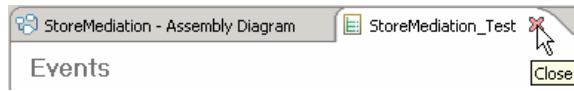
- ___ c. Click **Finish** after you see the applications moved to Available projects.
- ___ d. If displayed, click **OK** in 'Removing Project' window. Optionally, you can check the box for 'Do not show again' not to be asked again when you remove projects later



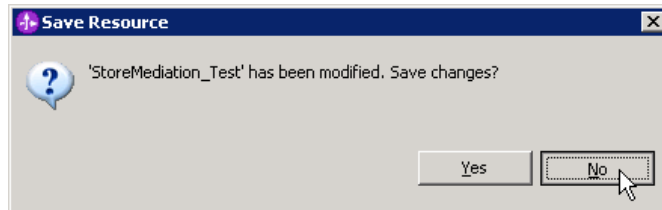
- ___ e. Wait until the application is removed from the server
- ___ 2. Close the StoreMediation_test panel without saving

• -----

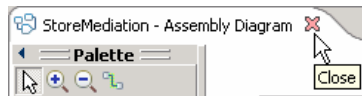
___ a. Click **X** on the StoreMediation_Test tab



___ b. Click **No** from Save Resource window




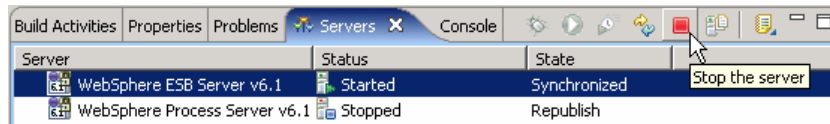
___ c. Click **X** on the StoreMediation – Assembly Diagram tab



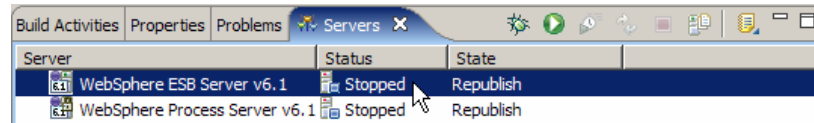
___ 3. Stop the test server

• -----

___ a. From the **Servers** view of WebSphere Integration Developer, select **WebSphere ESB Server v6.1** and hit '**Stop the server**' icon () from the toolbar



___ b. Wait until the server Status shows as **Stopped**



___ 4. Exit from WebSphere Integration Developer.

• -----

___ a. From menu, select **File → Exit** or click '**X**' at the right top corner of your WebSphere Integration Developer window

What you did in this exercise

In this exercise, you modified an augmentation and aggregation message flow so that the 'service invoke' primitive used alternate endpoints when performing a service call retry. This involved setting up the endpoints in the SMO and appropriately configuring the service invoke.

Reviewing the presentation entitled [Augmentation, aggregation and retry tutorials](#) will help you better understand what was done in the lab.

Solution instructions

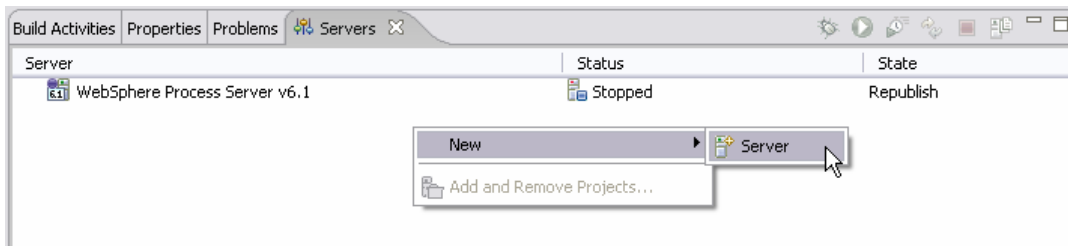
If you want to run this lab with a completed solution rather than authoring the flow yourself, follow these instructions:

- ____ 1. Follow **Part 1: Setting up the environment for the lab**, however use the project interchange file that contains the solution.
 - `<LAB_FILES>/PI5-AlternateEndpointsSolution.zip`
- ____ 2. Skip to **Part 3: Test the retry with alternate endpoints mediation flow** and proceed through the rest of the lab.

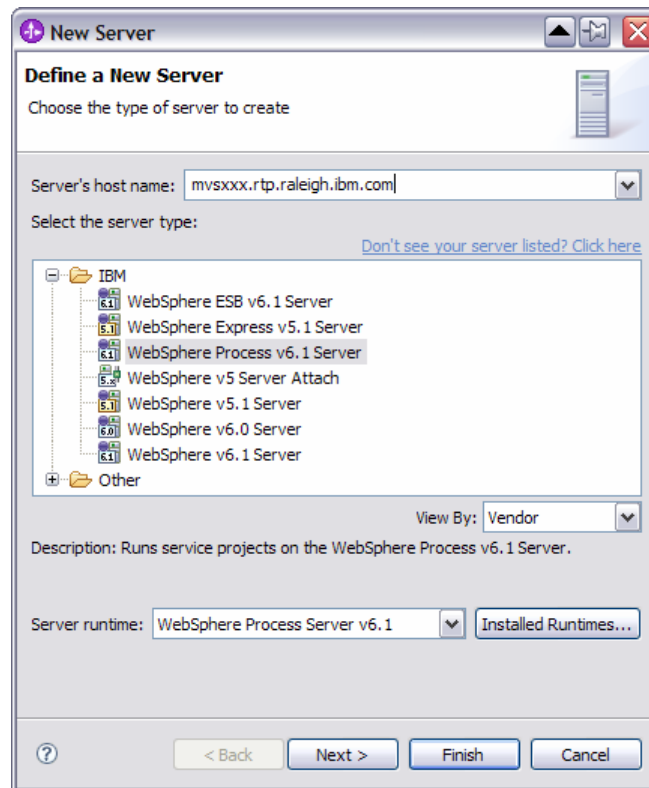
Task: Adding remote server to the WebSphere Integration Developer test environment

This task describes how to add a remote server to the WebSphere Integration Developer Test environment. This example uses a z/OS machine.

- ___ 1. Define a new remote server to WebSphere Integration Developer.
 - ___ a. Right click on the background of the Servers view to access the pop-up menu.
 - ___ b. Select New → Server.

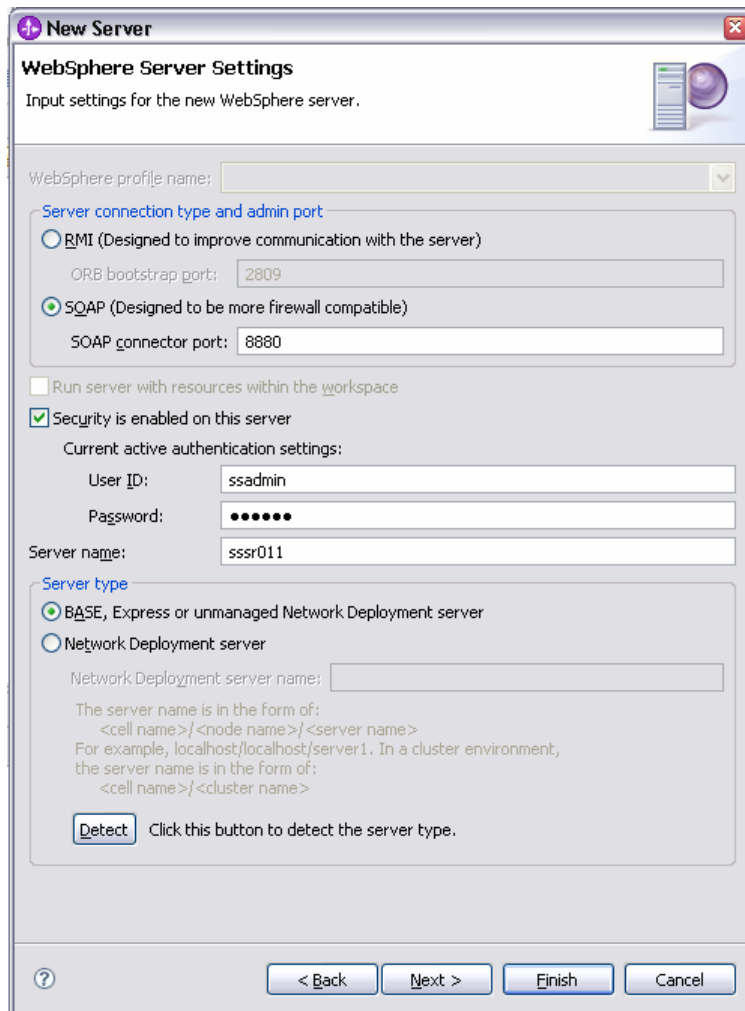


- ___ c. In the New Server dialog, specify the remote server's host name, <HOSTNAME>.
- ___ d. Ensure that the appropriate server type, 'WebSphere Process v6.1 Server' or 'WebSphere ESB v6.1 Server', is highlighted in the server type list

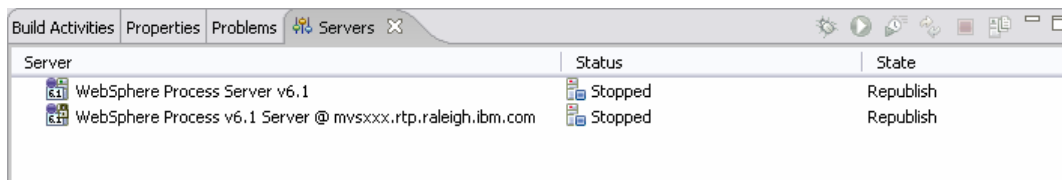


- ___ e. Click **Next**.

- ___ f. On the WebSphere Server Settings page, leave the radio button for **SOAP** selected, changing the **SOAP connector port** to the correct setting (<SOAP_PORT>). If security is on in your server, check the box for 'Security is enabled on this server' and input <USERID> for the user ID and <PASSWORD> for the password.



- ___ g. Click **Finish**.
- ___ h. The new server should be seen in the Server view.



- ___ 2. Start the remote server if it is not already started. WebSphere Integration Developer does not support starting remote servers from the Server View.
- ___ a. From a command prompt, telnet to the remote system if needed:

'telnet <HOSTNAME> <TELNET_PORT>'

User ID : **<USERID>**

Password : **<PASSWORD>**

__ b. Navigate to the bin directory for the profile being used:

cd <WAS_HOME>/profiles/<PROFILE_NAME>/bin

__ c. Run the command file to start the server: **./startServer.sh <SERVER_NAME>**

__ d. Wait for status message indicating server has started:

```
ADMU3200I: Server launched. Waiting for initialization status
```

```
ADMU3000I: Server sssr01 open for e-business; process id is 0000012000000002
```