



# **SupportPac LA71: IBM WebSphere Operational Decision Management Integration for WebSphere Process Server**

## **Getting started with IBM Business Process Manager**

### **Task 6 – Integration Designer assembles a BPEL Process containing an SCA Decision Component**

# Copyright

## Copyright notice

© Copyright IBM Corp. 1987, 2012

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

---

## Trademarks

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

Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at *Copyright and trademark information*.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

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

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

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

---

## Overview

With the business rules created, the next step is to utilize this business logic in a business processes. In this task you will create the business process that calls the business rules using the SCA Managed Decision Components created in task 5.

Readers who want to skip the exercise should delete the existing **BPELProcessDecisionModule** projects and import the task 6 solution into IID from **[SupportPac LA71 Path]\BPMTutorial\task6\BPELProcessDecisionModule.zip**. You can continue at Step 8 to test the Process in the Process Server.

### Step 1. Define the Process Interface

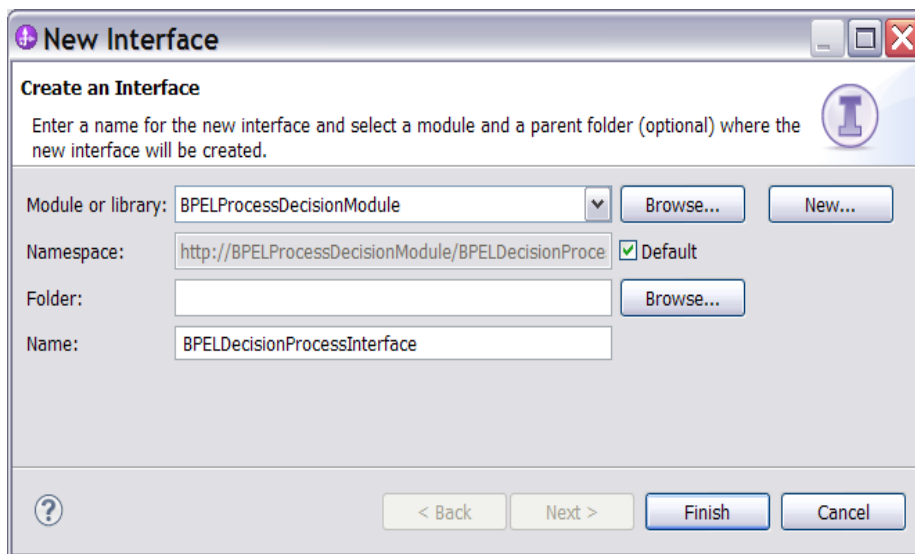
The business process requires an interface which can be called to start the BPEL process. In this step you will create a new interface using the existing Business Objects.

Select the **Interfaces** folder and right click.

Select **New > Interface**.

Make sure the Module is defined as **BPELProcessDecisionModule**

Define the interface name as **BPELDecisionProcessInterface**



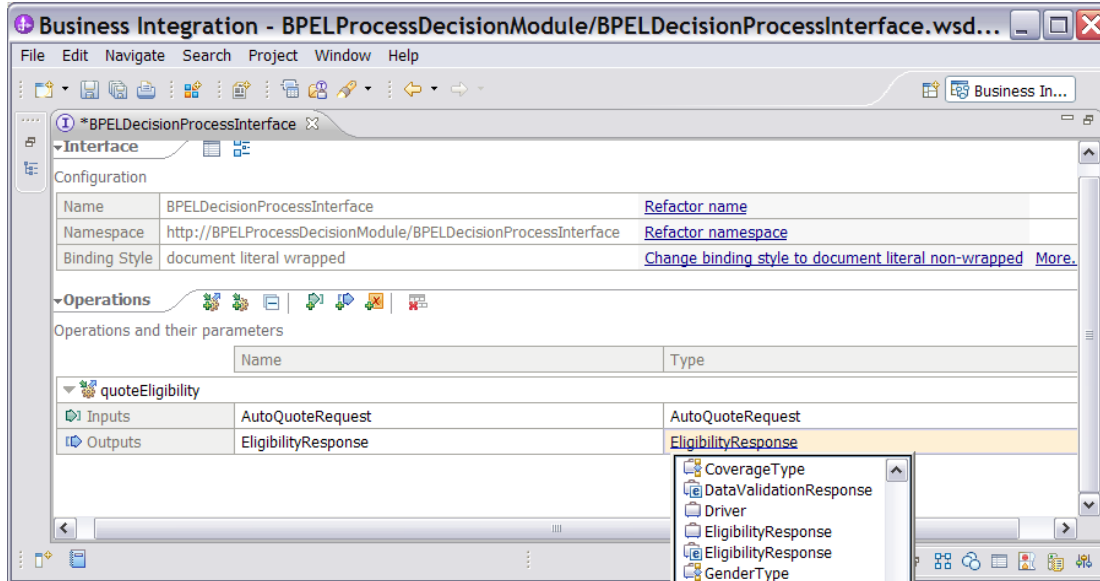
Click **Finish**.

In the **BPELDecisionProcessInterface** tab that opens :

Click the **Add Request Response operation** icon and rename the resulting operation to **quoteEligibility**.

Rename the operation input to **AutoQuoteRequest** and select the existing **AutoQuoteRequest** type.

Rename the operation output to **EligibilityResponse** and select the existing **EligibilityResponse** type.



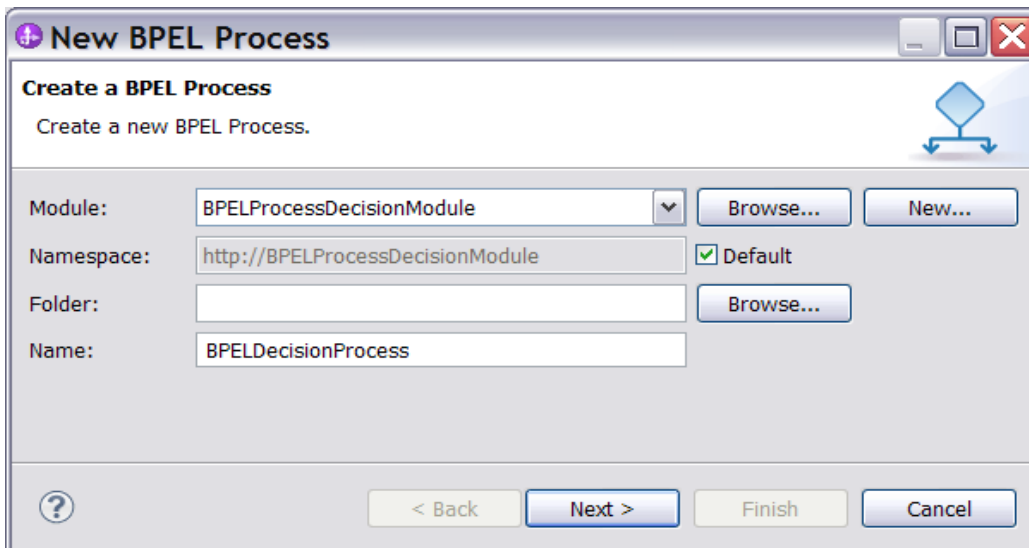
Save the updated **BPELDecisionProcessInterface** interface.

## Step 2. Create the Business Process Implementation for Quote Eligibility

In this step you will create a new BPEL process to that will be started from this interface.

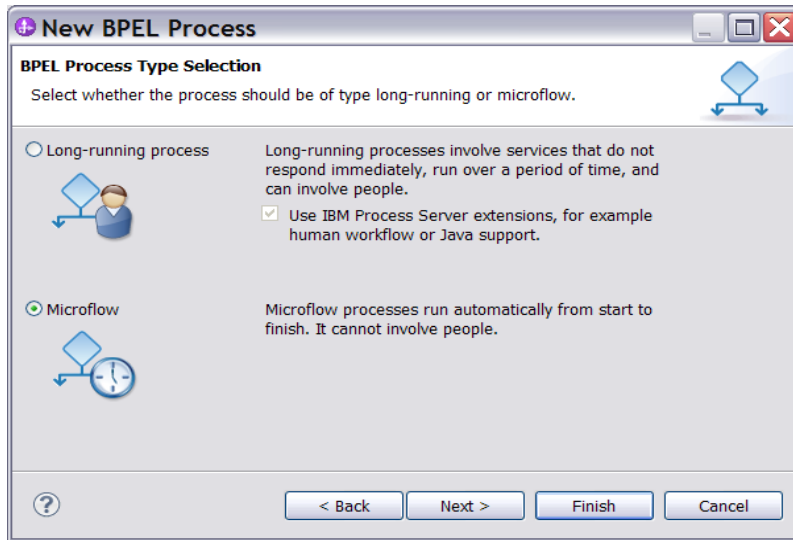
Select **File > New > BPEL Process**.

In the Create a BPEL Process screen, make sure the **BPELProcessDecisionModule** is selected. For the name, enter **BPELDecisionProcess**.

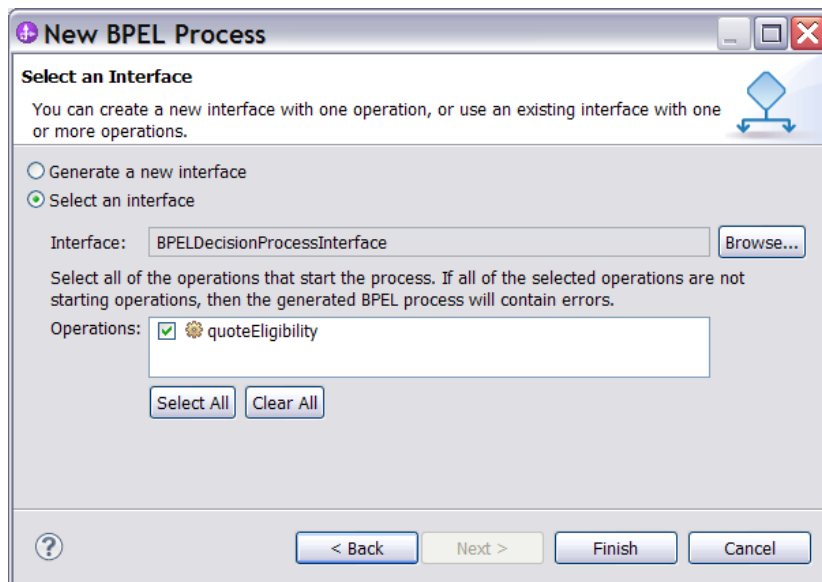


Click **Next**.

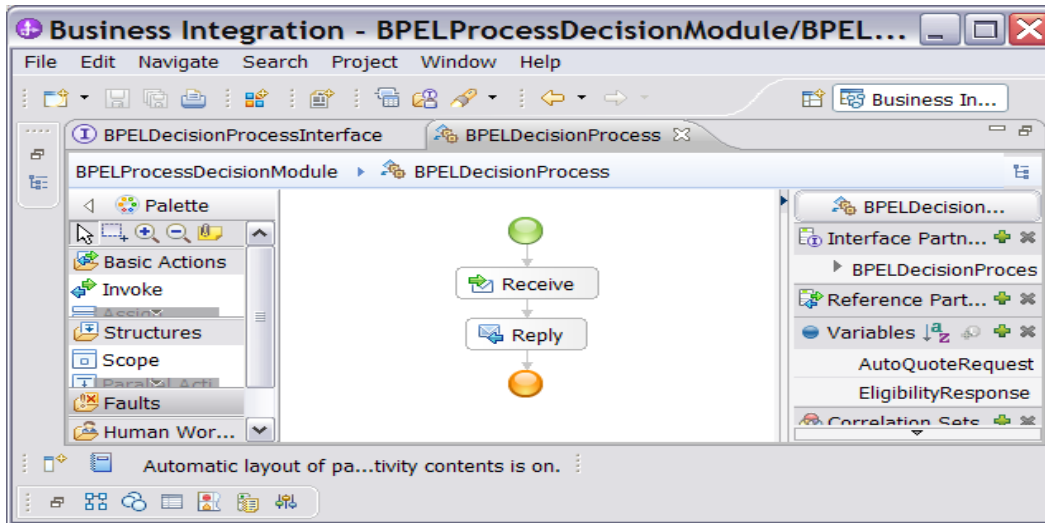
On the **Business Process Type Selection** screen, select **Microflow**.



Click **Next**. In the **Select an Interface** screen,  
 Select the radiobutton: **Select an interface**.  
 Click **Browse...** and select the **BPELDecisionProcessInterface** you have just created.  
 Ensure the **quoteEligibility** operation is selected and click **Next**.



The BPEL process is created and displayed in the process editor.



### Step 3. Add logic for checking eligibility before confirming the quote.

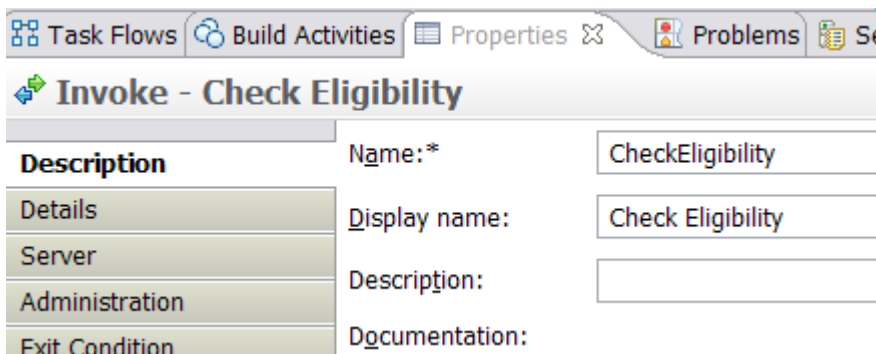
In the process editor, from the Palette under Basic Actions, select **Invoke**. Click beneath the **Receive** activity to insert the **Invoke** activity.

Select the **Invoke** activity and select the **Properties** view.

Change the Name to **CheckEligibility**.

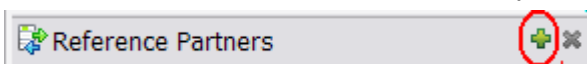
Change the Display name to **Check Eligibility**.

This node will make the call to the ManagedDecision CheckEligibility decision to determine if the individual is eligible for a insurance.

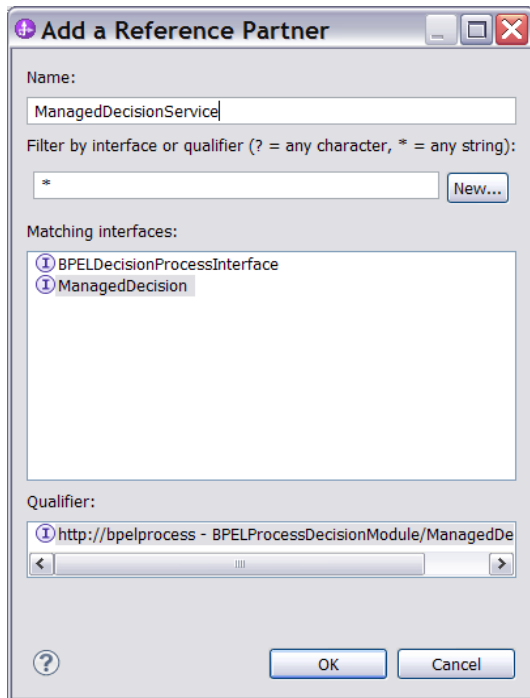


The Decision Service is called through a Reference Partner that must now be created.

In the Process Editor, select the add symbol for Reference Partners.



In the **Add a Reference Partner** screen  
 For the name type **ManagedDecisionService**  
 In the **Matching interface** field select **ManagedDecision**

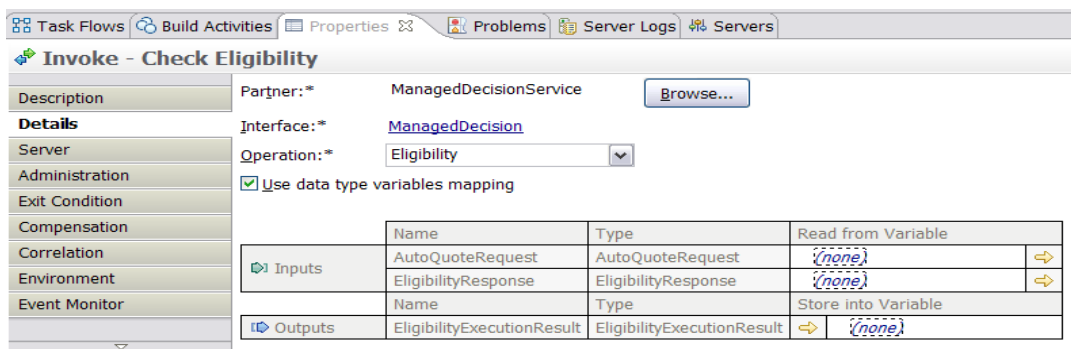


Click **OK**.

Select the Check Eligibility node again and in the Properties view, select the **Details** tab.

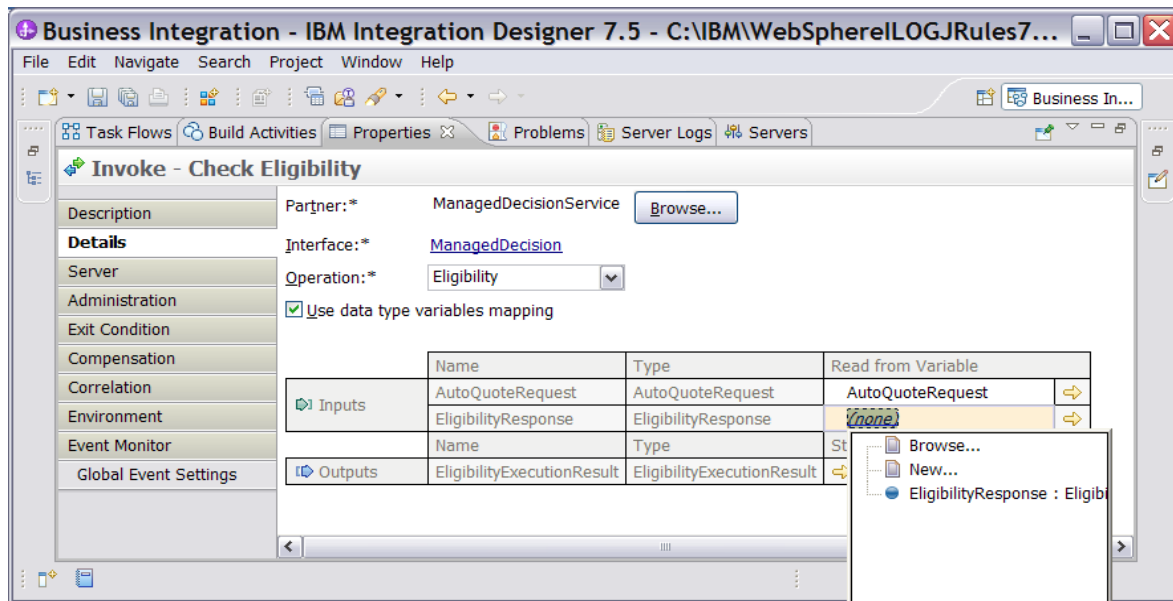
For the Partner, select **Browse...** and select **ManagedDecisionService** (the partner you have just created)  
 Click **OK**.

Change the Operation to be **Eligibility**.

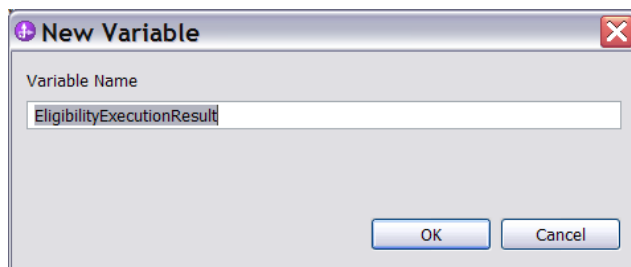


You now need to map the decision service parameters to process variables.

For the AutoQuoteRequest Input, under the Read from Variable column click **(none)**.  
 Select **AutoQuoteRequest** – the parameter input to the process when starting it.  
 For the EligibilityResponse Input, under the Read from Variable column click **(none)**.  
 Select **EligibilityResponse** – the parameter returned from the process when starting it. This parameter will need to be initialized before passing to the decision service.



For the **EligibilityExecutionResult** Output, under the Read from Variable column, click **(none)**.  
 Click **New...** and for the Variable Name accept the value **EligibilityExecutionResult**.



Click **OK**.  
 Save the Invoke activity and check the properties again.



**Invoke - Check Eligibility**

Description: Partner:\* ManagedDecisionService [Browse...](#)

Details: Interface:\* [ManagedDecision](#)

Operation:\* Eligibility

☒ Use data type variables mapping

	Name	Type	Read from Variable
Inputs	AutoQuoteRequest	AutoQuoteRequest	AutoQuoteRequest
	EligibilityResponse	EligibilityResponse	EligibilityResponse
	Name	Type	Store into Variable
Outputs	EligibilityExecutionResult	EligibilityExecutionResult	EligibilityExecutionResult

This should remove any error from the process and result in a flow including the invoke of the decision.

#### Step 4. Initialize the decision parameters.

In the process editor, from the Palette under Basic Actions, select **Assign** and click after the Request node.

Rename the node to **Initialize**. This will be used to ensure that the EligibilityResponse variable is initialized.

Select the **Initialize node** and select the Properties tab.

**Assign - Assign**

Details: Server, Exit Condition, Environment

Assign From: ab

Assign To: EligibilityResponse MainMessage

[Add](#) [Remove](#) [Up](#) [Down](#)

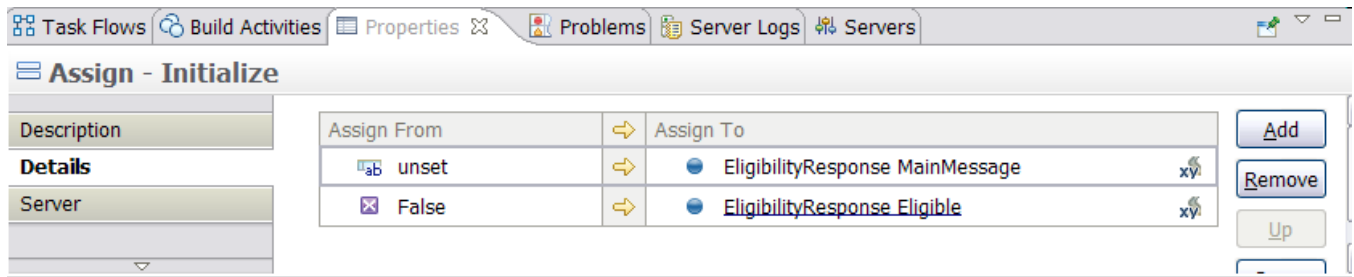
**ab Assign From - String**

In the **Select From** field select a **String** and enter the value **"unset"**.

In the **Select To** select **EligibilityResponse**, open the twistie and select **MainMessage**.

Click the **Add** Button.

Repeat this to set the **EligibilityResponse.Eligible** to **false**.

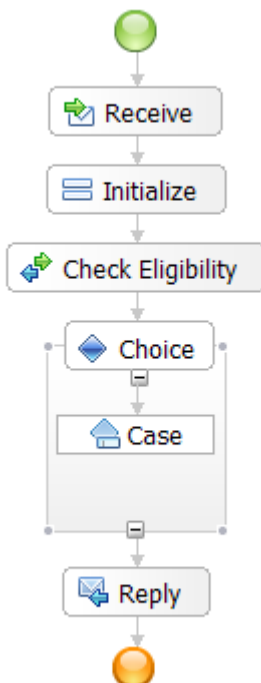


This has now initialized the EligibilityResponse variable so it can be used as an in/out parameter to the decision service.

### Step 5. Use the managed decision to influence the process flow.

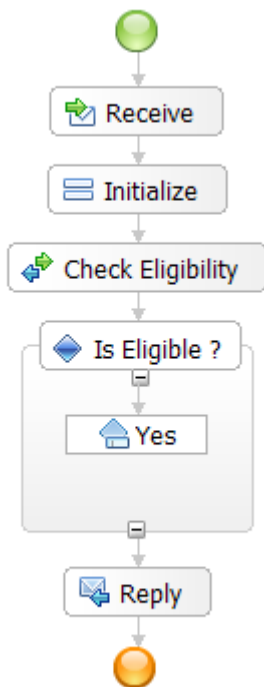
Combining a call to an Invoke (representing a decision) followed by a Choice node is a common design pattern which reduces the amount of hard-coded logic. The decision is reused from the BRMS but the "Choice" enforces the decision in the context of the process. The logic can be captured in business rules with the result checked by the Choice and different cases

In the process editor, from the Palette under **Structures**, select **Choice**. Click beneath the **Check Eligibility** node. A Choice is inserted into the flow.



Select the **Choice** node and in the Properties view change the Name to **Is Eligible?**. A Case is created by default.

Select the Case and in the Properties view change the Display name to **Yes**.

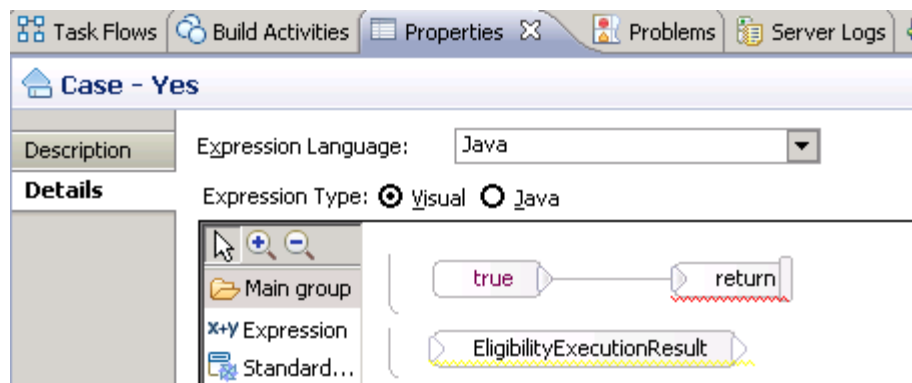


The “Yes” condition is executed if the result from the Check Eligibility invoke is true.

To specify the condition, select the Details tab.

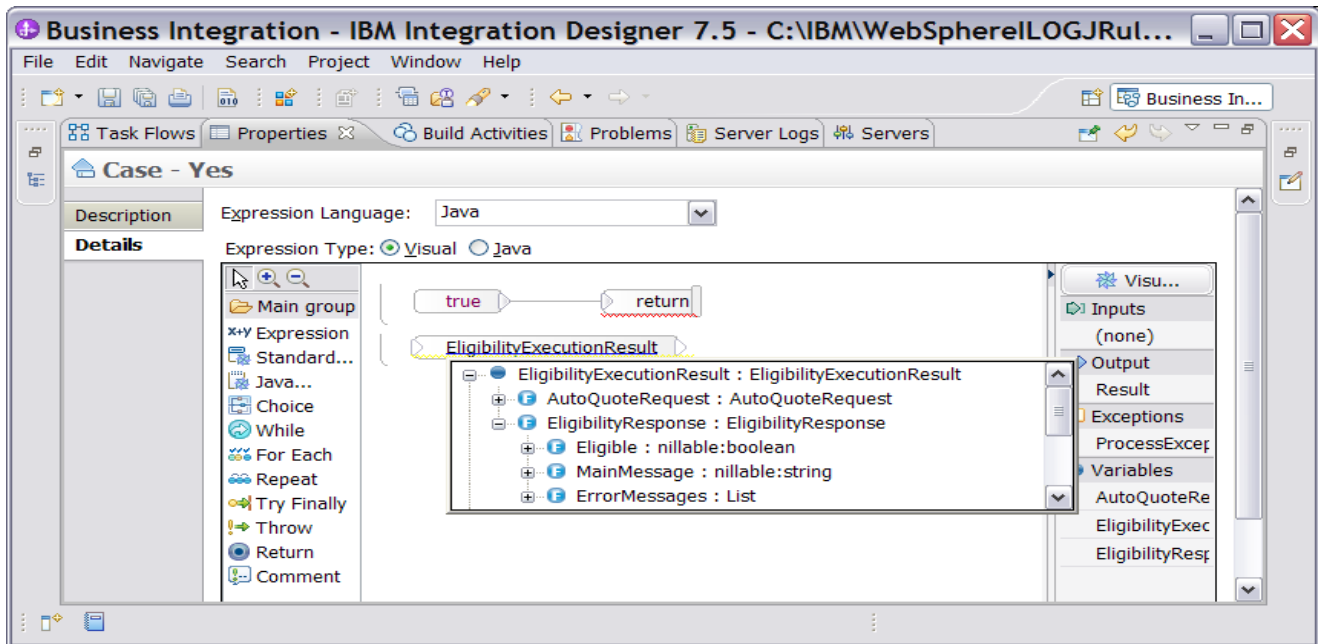
For the Expression Language select **Java**.

From the list of Variables, drag and drop **EligibilityExecutionResult** onto the case editor.

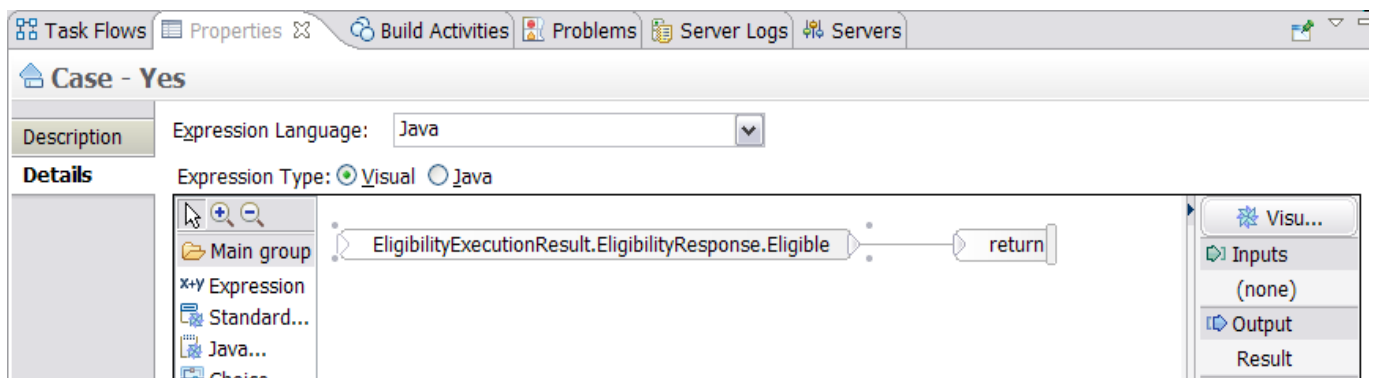


Click **EligibilityExecutionResult** to see the attributes for the variable.

Expand and select **EligibilityExecutionResult > EligibilityResponse > Eligible**.



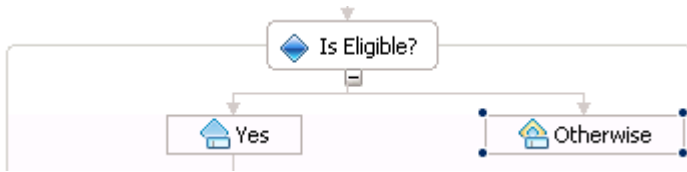
Delete the **true** value and wire the **EligibilityExecutionResult** variable to the **return**. Note that the wire will not operate until the fully qualified value is used.



Besides the Yes case, a case is needed to handle a customer that is not eligible. Click the **Is Eligible?** choice node.



Select the **Otherwise** icon. An otherwise case is added.



You have now created two cases that the process will follow dependent on the decision made by the BRMS as to whether the customer is eligible for insurance or not.

## Step 6. Define Process Response Messages based on decision cases.

In this step you will provide the process logic for each of the decision cases.

A confirmation needs to be returned to the customer according to whether they were eligible or not.

For this process, a snippet will be used to specify a confirmation message.

From the Palette, select **Snippet** from the list of Basic Actions.

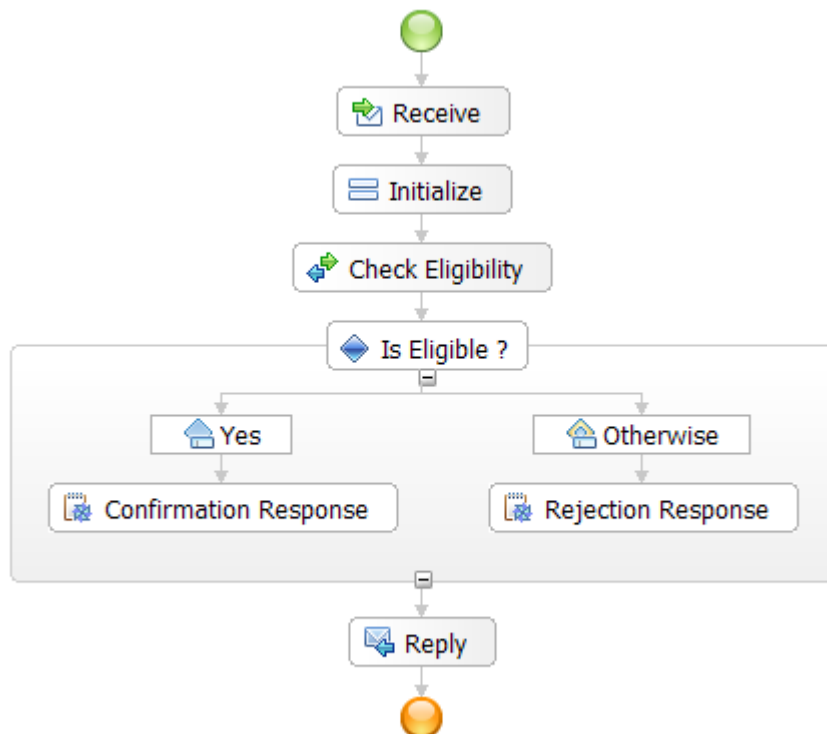
Click beneath the **Yes** case node.

Change the Display name to **Confirmation Response**.

From the Palette, select **Snippet** from the list of Basic Actions.

Click beneath the Otherwise case node.

Change the Display name to **Rejection Response**.



Each of these responses will be generated by a small amount of visual Java code.

Select the **Confirmation Response** node and in the Properties view select the Details tab. Leave the **Visual** radio button selected.

From the Variables panel drag the EligibilityExecutionResult onto the canvas.

Select the **EligibilityExecutionResult > EligibilityResponse** which contains the response back from the decision service.

Drag the **EligibilityResponse** onto the canvas – this is the response passed back from the decision service.

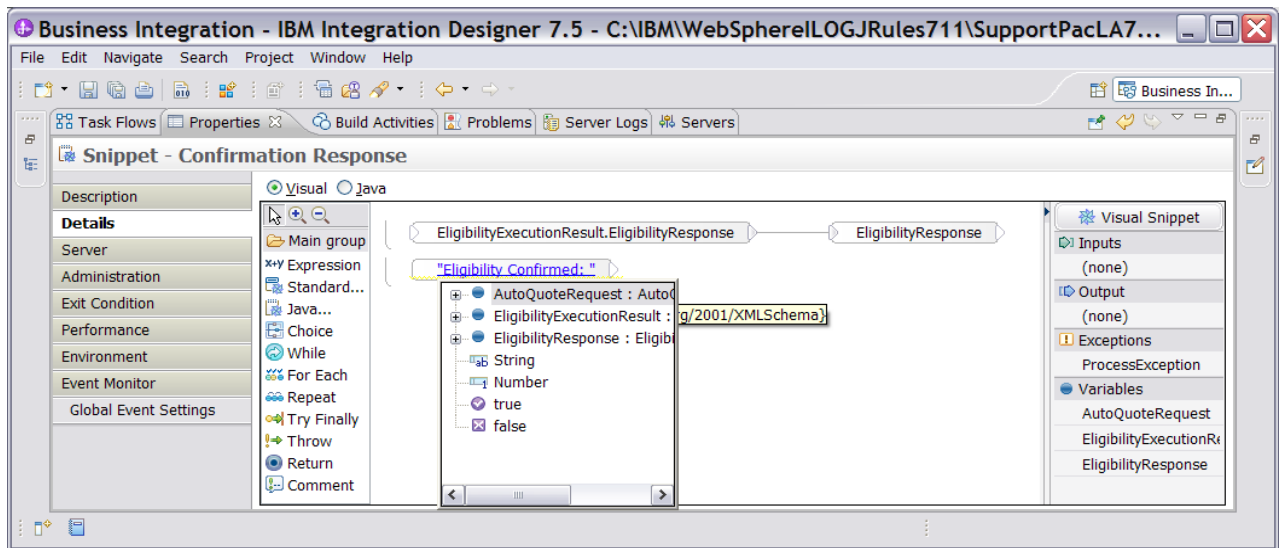
Wire the **EligibilityExecutionResult** to the **EligibilityResponse**.

This will copy the response from the decision service into the response from the process.

From **Main Group** of icons select Expression and click on the canvas under EligibilityExecutionResult.

Click in the middle of the icon and select String from the popup.

Enter **"Eligibility Confirmed: "** as the string contents.



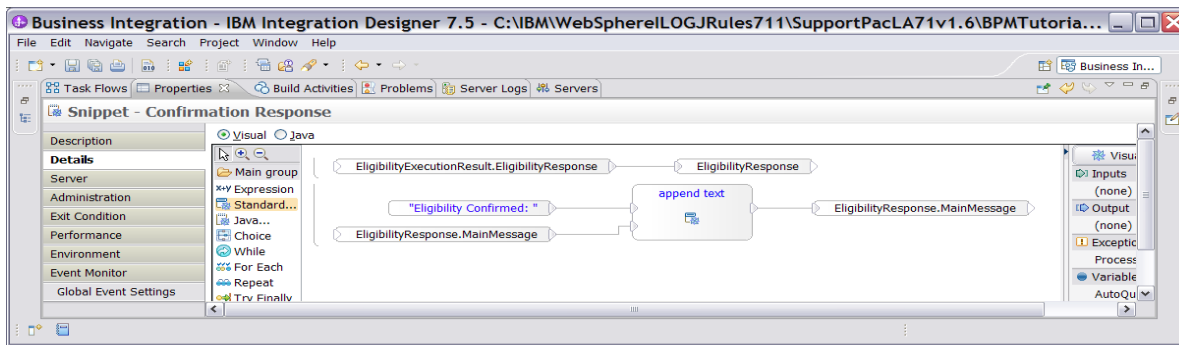
Drag the EligibilityResponse Variable onto the canvas beneath this string and select EligibilityResponse.MainMessage.

In the Main Group of icons select **Standard Visual Snippet > text > append text** and click **OK**. The visual snippet is added to the diagram.

Wire the **"Eligibility Confirmed"** string to the first input of **append text** and the **EligibilityResponse.MainMessage** to the second. This will concatenate the two strings.

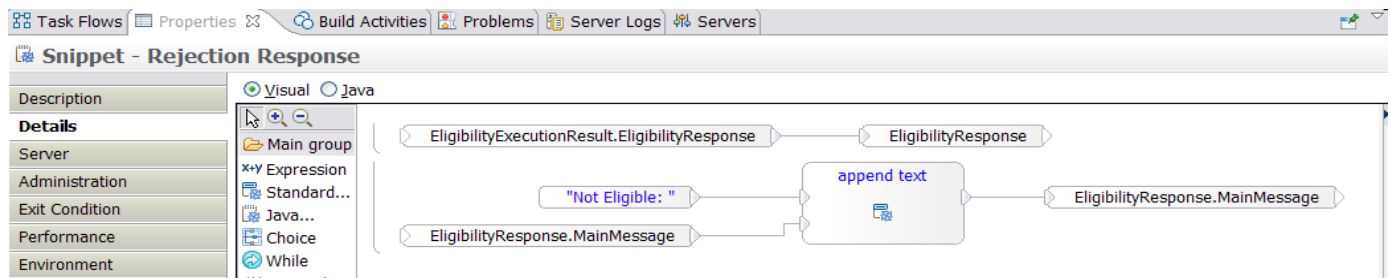
Drag the **EligibilityResponse** Variable onto the canvas again and select **EligibilityResponse.MainMessage**.

Wire the output of the **append text** snippet to this new **EligibilityResponse.MainMessage**. This will set the result to the concatenated string.



Now you need to repeat this for the Rejection Response.  
Before you leave the Confirmation Response select and copy all the visual snippets.

In the Process Editor select the Rejection Response node and open the Properties view Details tab.  
Paste the copy buffer into the window.  
Select the **"Eligibility Confirmed: "** string and change the text to **"Not Eligible: "**

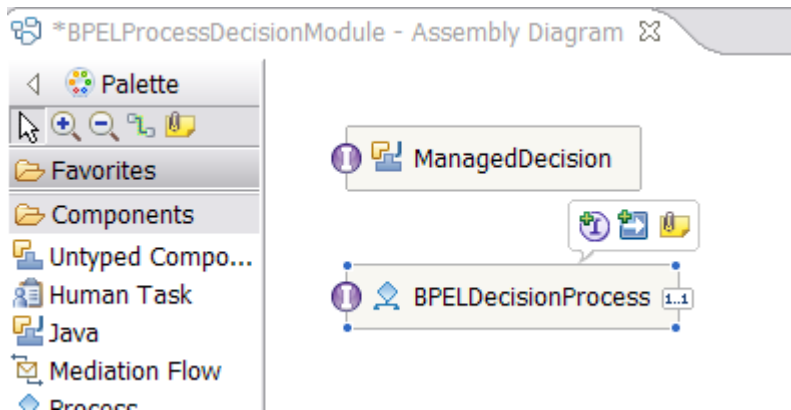


Save your work. You have now completed the implementation of the process.

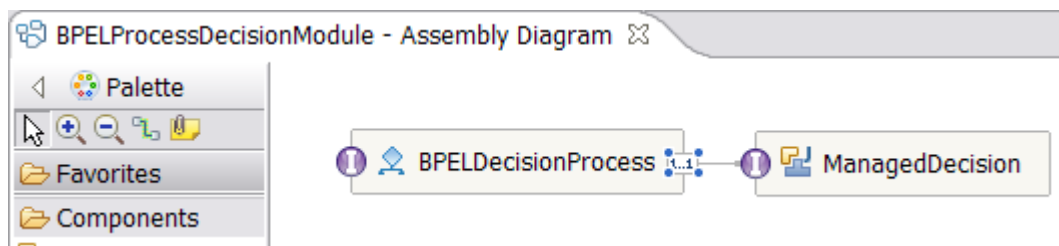
You have now created two cases that the process will follow dependent on the decision made by the BRMS as to whether the customer is eligible for insurance or not.

## Step 7. Assembling the BPM Application.

In this step you will assemble the full process solution before deploying and testing it.  
Open the **Assembly Diagram** to open the assembly editor if it is not already opened  
In the Business Integration view, under **BPELProcessDecisionModule > Integration Logic > BPEL Processes**, select the **BPELDecisionProcess** process.  
Drag and drop the **BPELDecisionProcess** process onto the assembly editor.



Connect the reference on the **BPELDecisionProcess** to the **ManagedDecision** component by dragging the connector from the reference to the interface of the component.



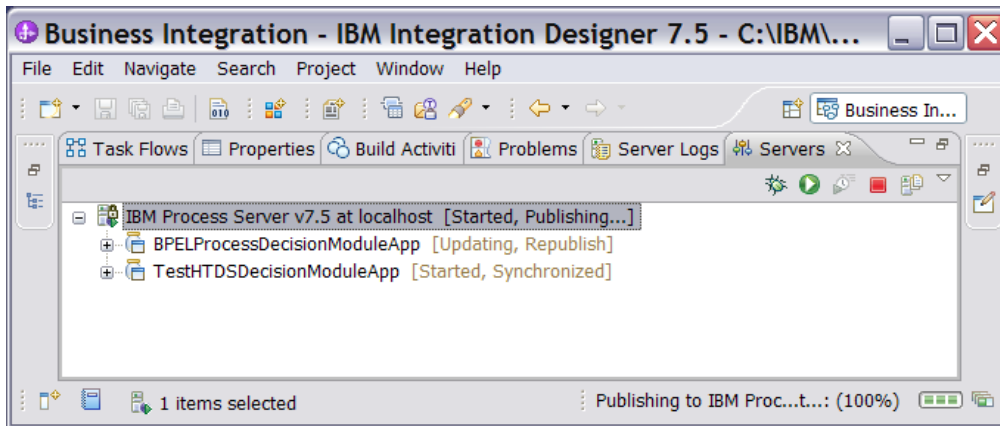
Save the assembly editor. Check for any errors in the Problems view and resolve. Your application is complete and ready for a full test of the process and rules.

## Step 8. Deploy and test the BPEL Process.

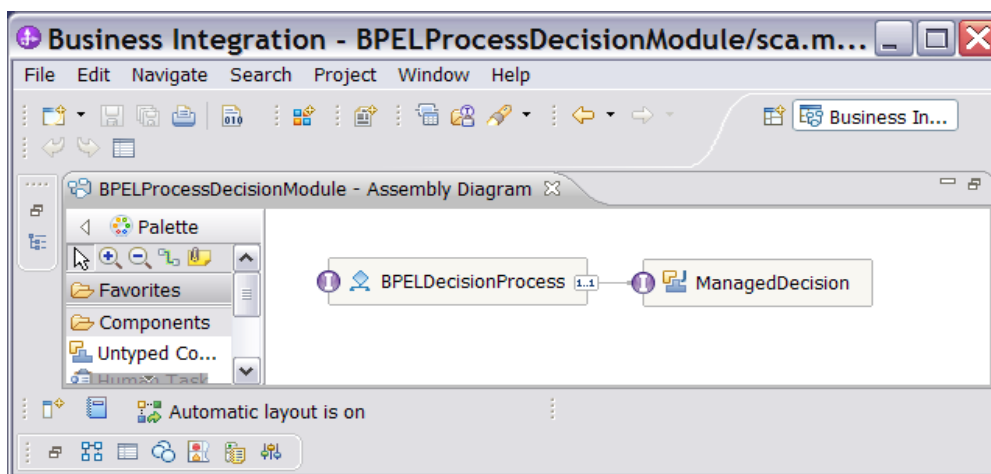
Just as the business rules, the business process can be deployed as an application to the Business Process Manager. Unlike the business rules, deployment is simply installing the application on the server. Once deployed, the process can be tested using the Test Component facility, Business Process Choreographer Explorer, or Business Space. You will test the process and the rules using the Test Component.

The process needs to be published to the server. In the Servers view right click your IBM Process Server v7.5 environment and select publish.





Open the Assembly Diagram.  
 Select the **BPELDecisionProcess**, right click and select **Test Component**.

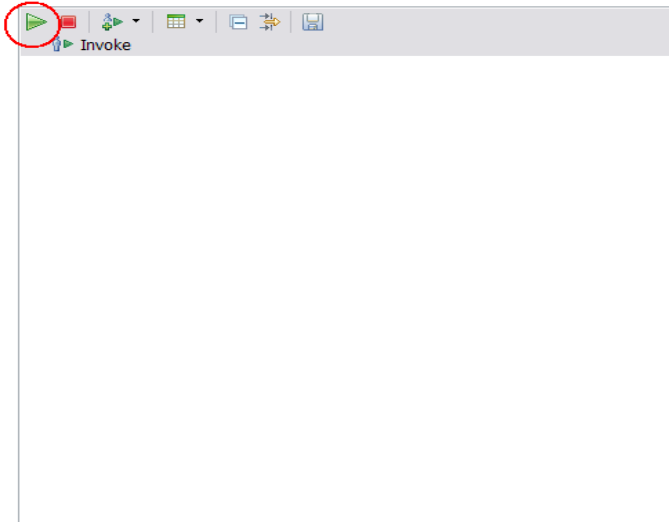


In the Integration Test Client Value Editor, select the **NumberOfAccidents** property and set it to 5.  
 The rules should indicate that the quote in Not eligible.  
 Select the continue icon to start the Test Client.

## Integration Test Client: BPELProcessDecisionModule\_Test

### Events

This area displays the events in a test trace. Select an event to display its properties in the General Properties and Detailed Properties sections. [More...](#)



### Detailed Properties

Specify the component, interface, operation, and input parameter values for the Invoke event, and then click the Continue icon in the Events area to run the test. [More...](#)

**Configuration:** Default Module Test

**Module:** BPELProcessDecisionModule

**Component:** BPELDecisionProcess

**Interface:** BPELDecisionProcessInterface

**Operation:** quoteEligibility

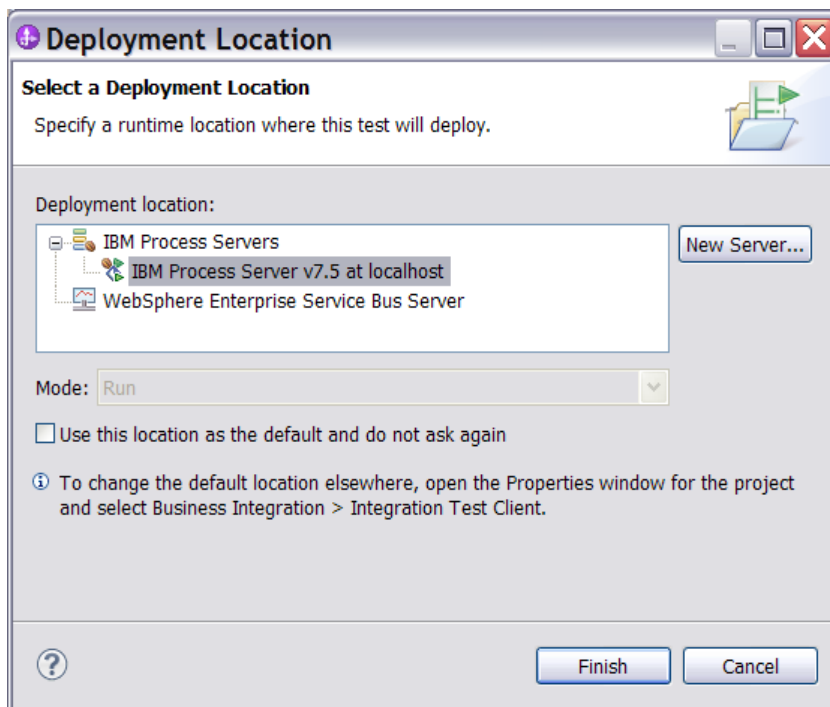
Initial request parameters:

☒ Value editor ☐ XML editor

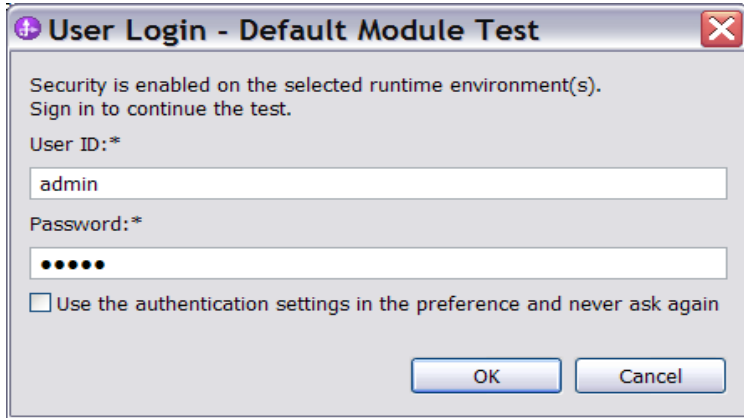
Name	T..	Value
VehicleVandalizedOrSt	b..	false
LicenseSuspendedOrR	b..	false
DUI	b..	false
NumberOfAccidents	i..	5
NumberOfTrafficTicket	i..	0
Vehicle	V..	
VehicleIdentificationNu	S..	

To edit values, start typing or press F2.

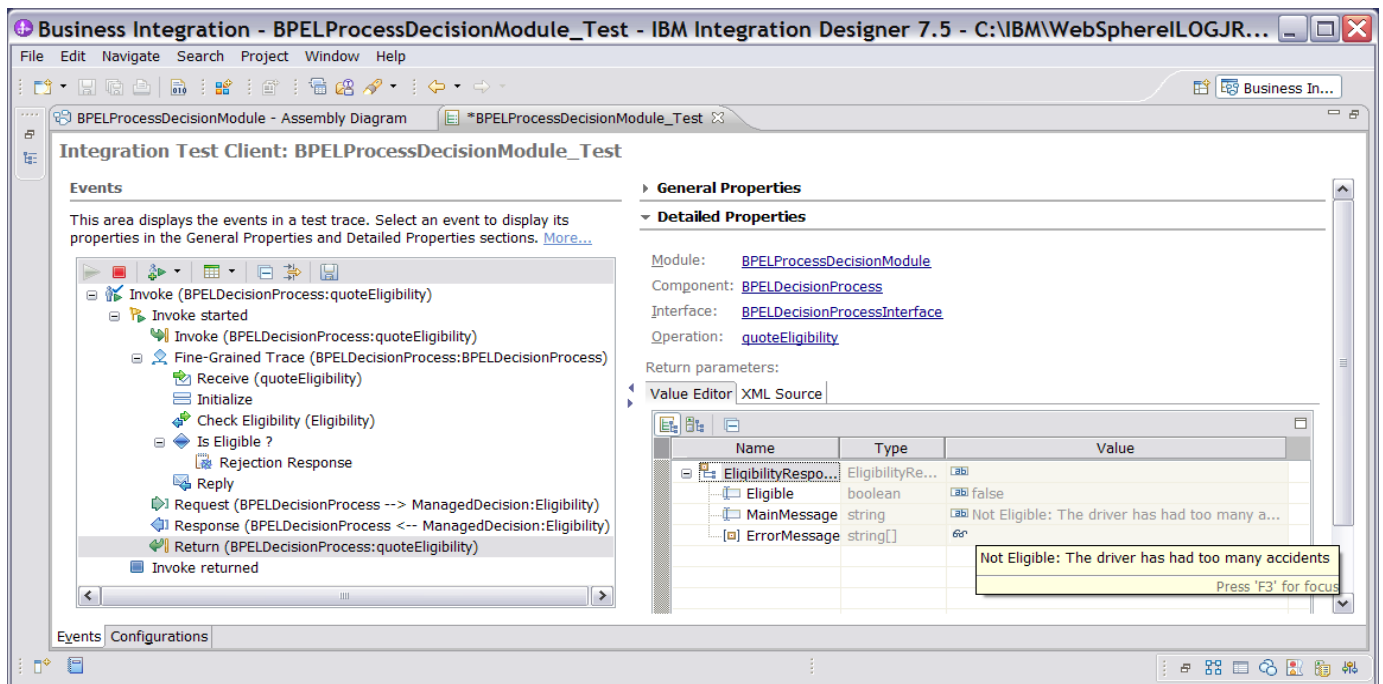
Check that the deployment location is correct and click **Finish**.



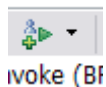
Check that the User Login characteristics are correct and click OK.



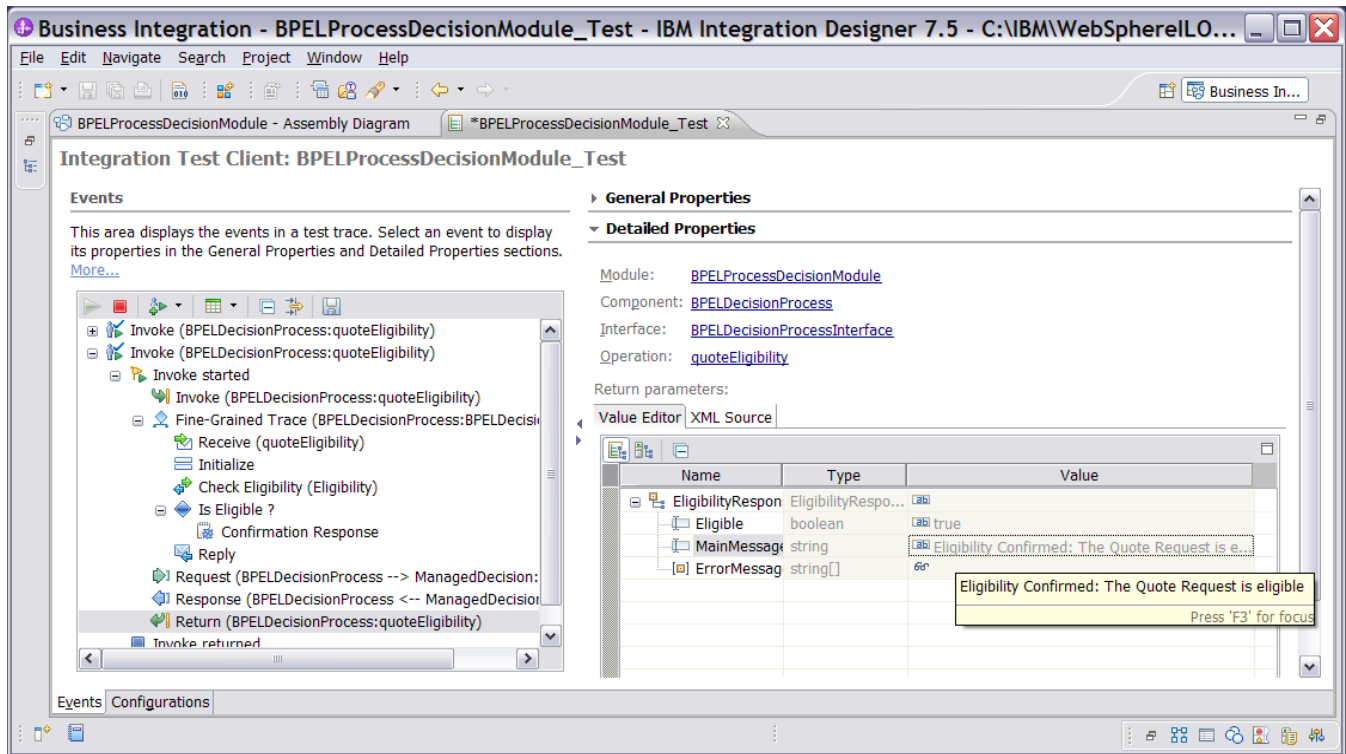
The process runs to completion showing the fine grained trace in the events panel. The **EligibilityResponse** from the process shows that the **Rejection Response** has correctly interpreted the response from the decision service



Try running the integration client again by selecting Invoke but this time select value of 2 for the number of accidents.



This time the Eligible branch of the flow should be followed.



In this task you have developed and tested a BPEL Process that uses a BRMS Managed Decision Component.