SupportPac LA71: IBM Operational Decision Manager Integration for WebSphere Process Server

Getting started with
IBM Business Process Manager

Task 1 - Business Process Author defines a Decision as part of a BPMN Business Process

Copyright

Copyright notice

Trademarks

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

Overview

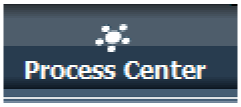
In Business Process Manager 8.0 Business Process Authors can use Process Designer to quickly define and simulate the business processes needed. This task provides a completed sample of a business process that has been created in this way including all the information models used in the tutorial. This will form the basis for the remainder of the tutorial.
You should complete all steps in this task to understand the tutorial scenario.

Step 1. Import Process Application Definition

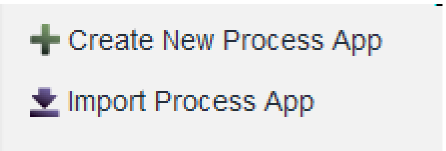
The Business Process author will define a business process application to contain the Business process and decisions that they require. This must be imported into Process Center before you can examine it.

Login to Process Designer as admin / admin

Navigate to Process Center by clicking the icon below

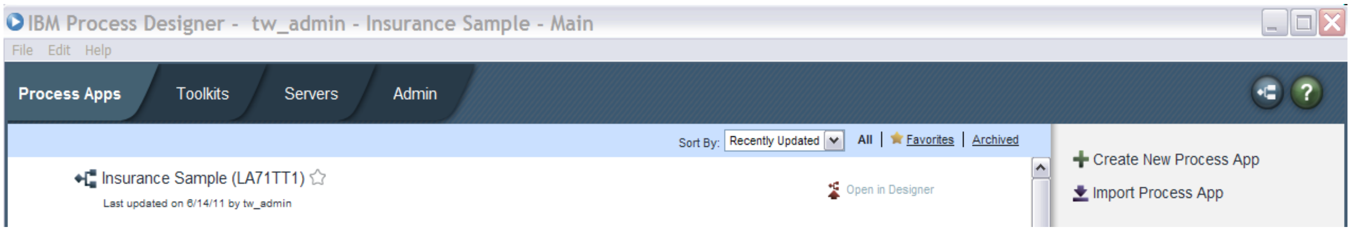


Import the completed Process App by selecting the Import Process App link.



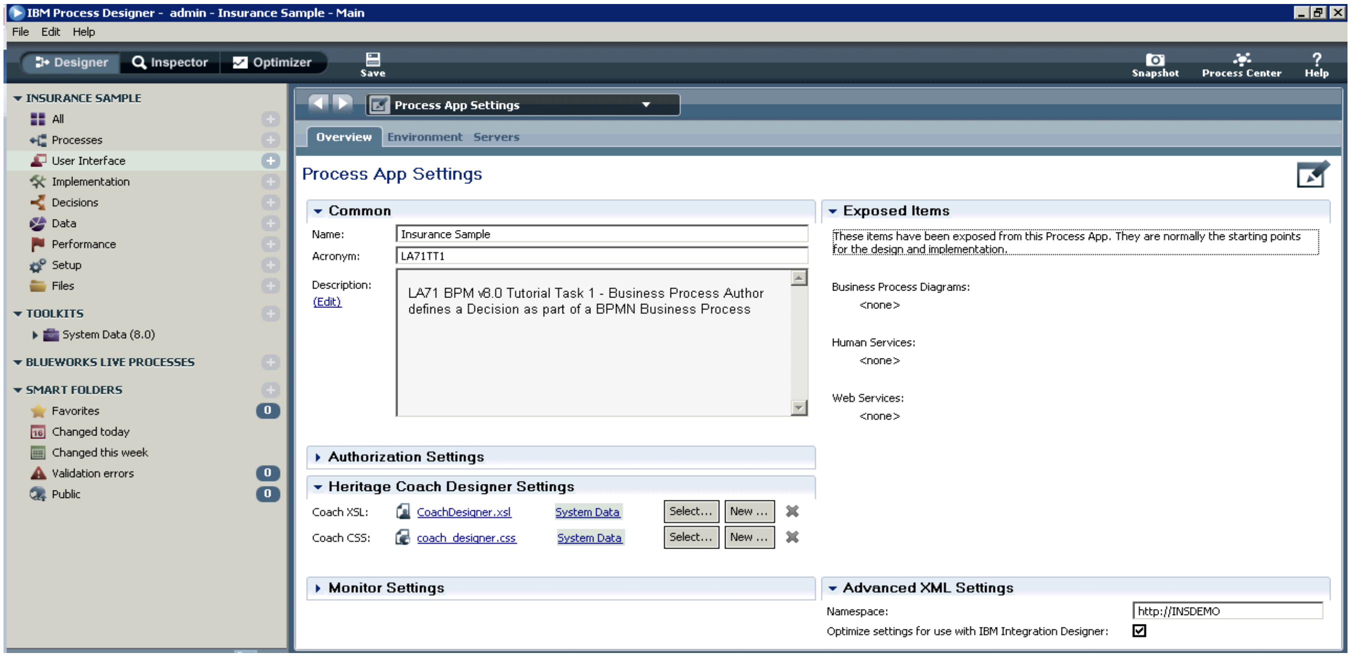
Browse to **[SupportPac LA71 Path]\BPMTutorial\task1\Insurance_Sample – LA71TT1.twx** and click **OK**.
In the **Will be Imported** screen check the name of the Process App and click **Import**.

The Insurance Sample (LA71TT1) - Process App should be visible in Process Center when it finishes importing.


IBM Process Designer - tw_admin - Insurance Sample - Main
File Edit Help
Process Apps
Toolkits
Servers
Admin
Sort By: Recently Updated
All | Favorites | Archived
Insurance Sample (LA71TT1)
Last updated on 6/14/11 by tw_admin
Open in Designer
Create New Process App
Import Process App

Select the **Open in Designer** link and Process Designer opens the Insurance Sample.

The Process App Settings screen shows the main details for the Process Application. Note that in this case the default Namespace used for all the Business Objects and interfaces has been defined as http://INSDEMO to keep consistency with the schemas that will be used by the development team using Integration Designer..



Process App Settings

Name: Insurance Sample

Acronym: LA71TT1

Description: (Edit)

LA71 BPM v8.0 Tutorial Task 1 - Business Process Author defines a Decision as part of a BPMN Business Process

These items have been exposed from this Process App. They are normally the starting points for the design and implementation.

Business Process Diagrams:
<none>

Human Services:
<none>

Web Services:
<none>

Coach XSL: CoachDesigner.xsl System Data

Coach CSS: coach_designer.css System Data

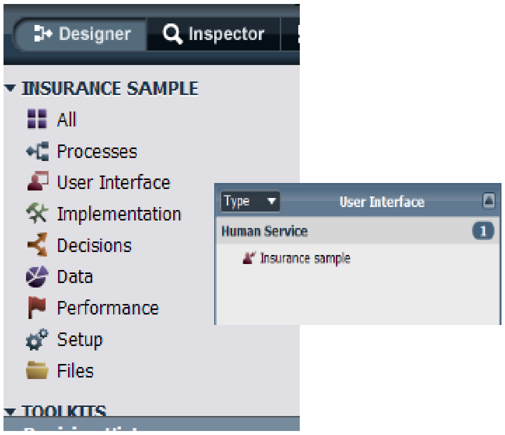
Namespace: http://INSDEMO

Optimize settings for use with IBM Integration Designer: ☑

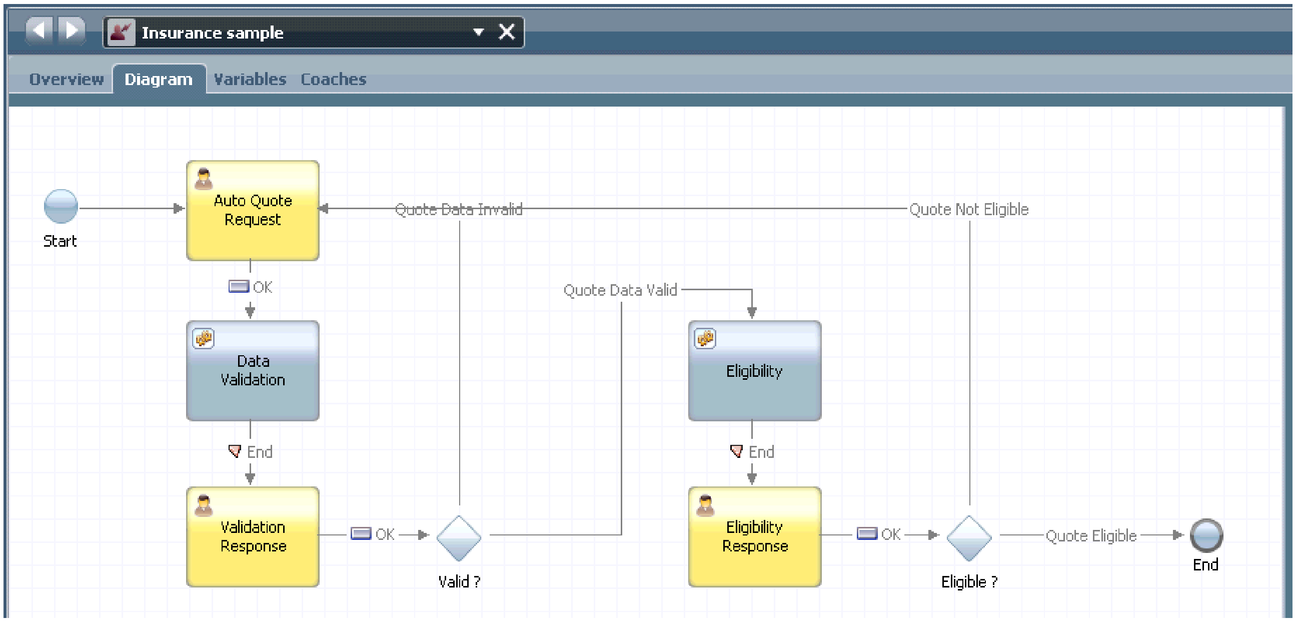
Step 2. Examine the Business Process.

The Insurance sample has a very simple process consisting only of a User Interface flow. To examine the flow:

Select the **User Interface** icon and from the pop-up double-click on the **Insurance sample** Human service.



The diagram tab shows the human service flow that will be used in this scenario.

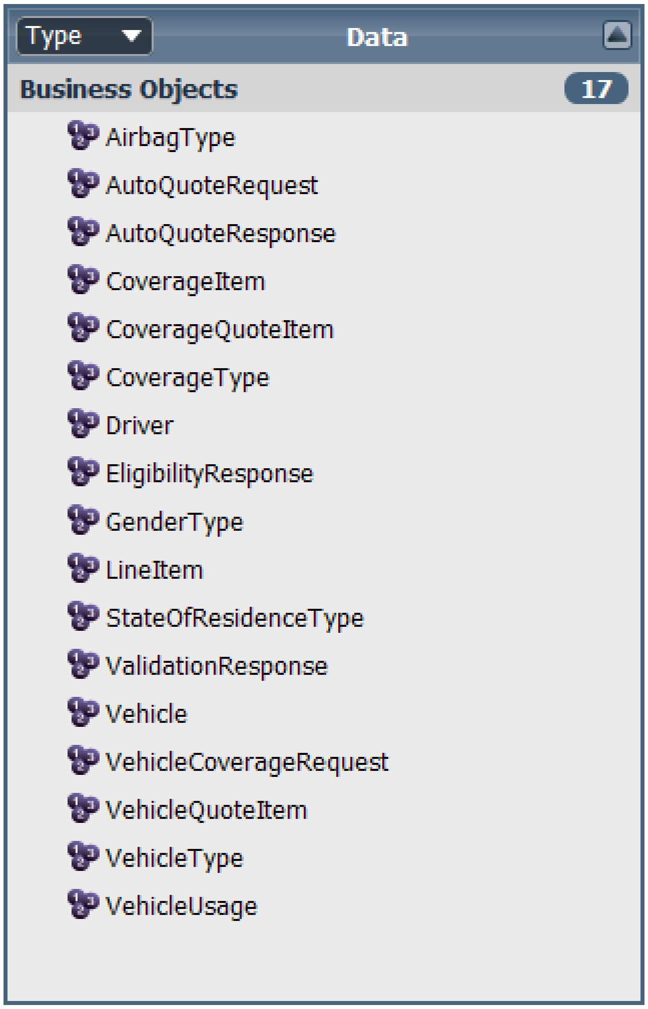


The **Auto Quote Request** Coach provides a means to set up details of the driver and vehicle for which the insurance quote is required. The **Data Validation** and **Eligibility** Nested Services are place-holders for the decisions to be made. These initially invoke the embedded rules services provided in Process Designer but will later invoke the Advanced Integration Services provided by development through Integration Designer.

Validation Response and **Eligibility Response** are Coaches that show the respective decision response. The **Valid ?** Gateway checks the response from the **Data Validation** decision and if any data is invalid returns to the **Auto Quote Request** Coach to re-enter the data. The **Eligible ?** Gateway checks if the quote request was deemed eligible for the quote and if not, again returns to re-enter the data. While not a realistic scenario this allows the two decisions to be easily evaluated.

Step 3. Examine the Information Model used in the process.

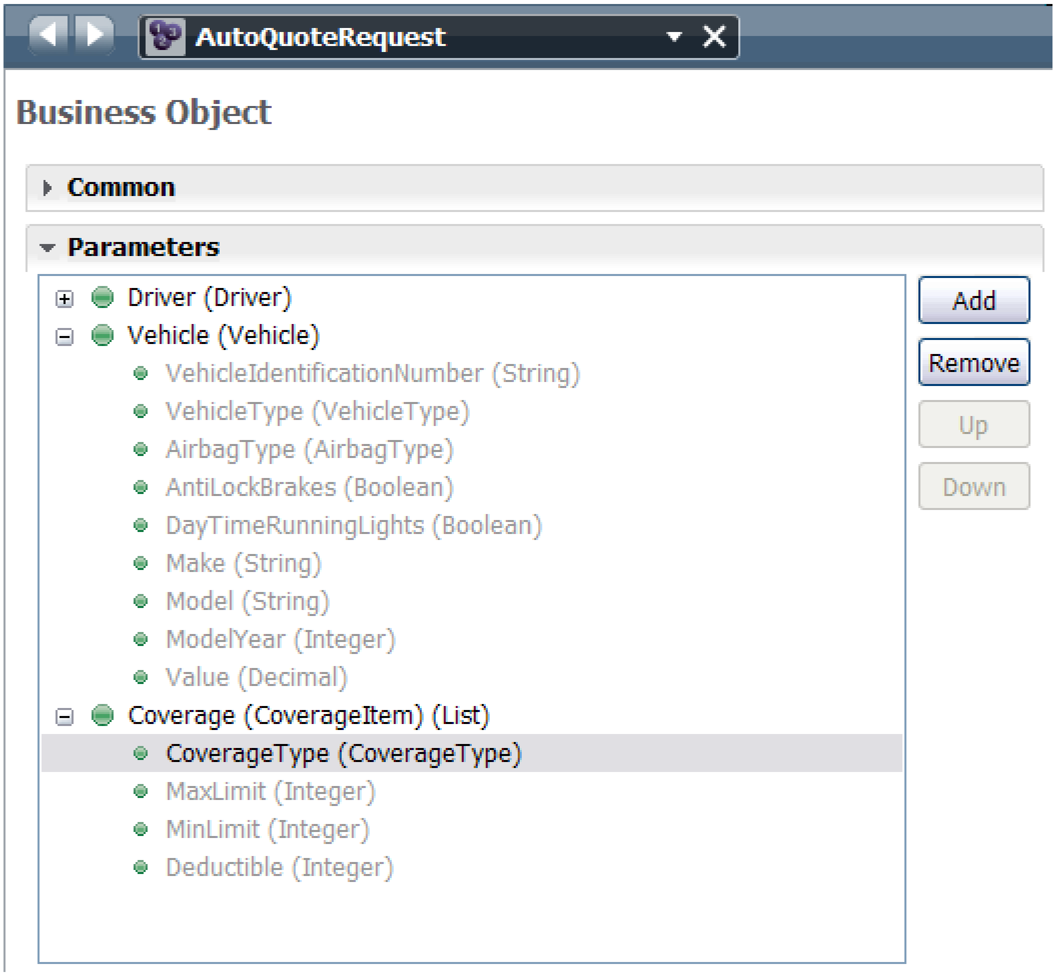
The Business Process Author will define an information model describing the characteristics of the quote that are important to the business. This would include details of the drivers, vehicles and type of cover required. All this information is described in the form of Business Objects in Process Designer allowing the Business Process Author to quickly express the sort of information they need. To examine the Information models select the **Data** icon and double-click the **AutoQuoteRequest** Business Object from the pop-up.



Business Objects 17

- AirbagType
- AutoQuoteRequest
- AutoQuoteResponse
- CoverageItem
- CoverageQuoteItem
- CoverageType
- Driver
- EligibilityResponse
- GenderType
- LineItem
- StateOfResidenceType
- ValidationResponse
- Vehicle
- VehicleCoverageRequest
- VehicleQuoteItem
- VehicleType
- VehicleUsage

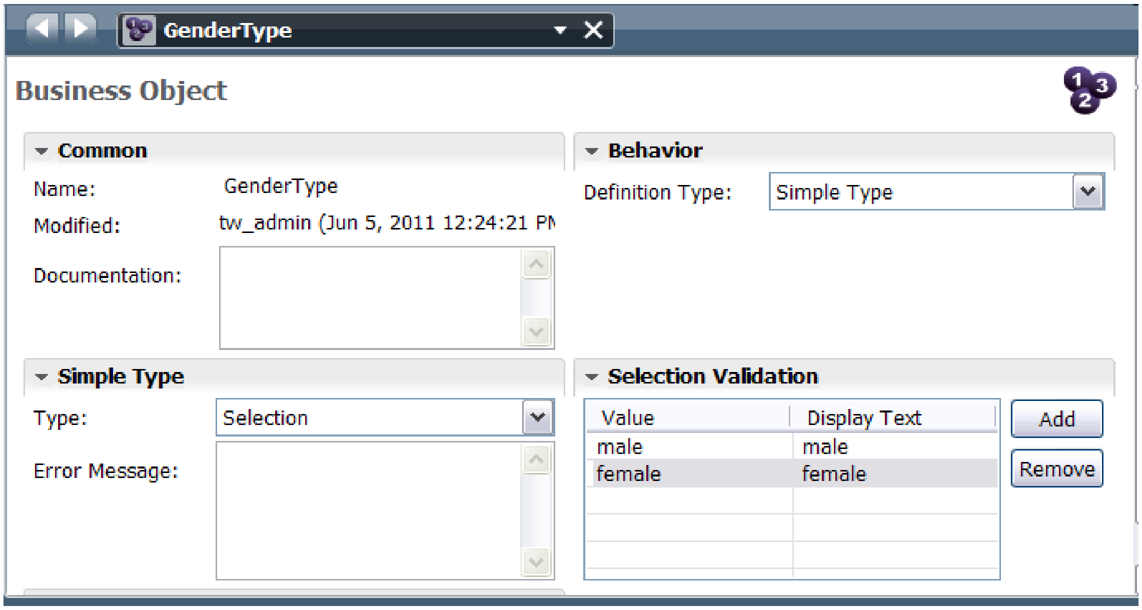
This will result in the **AutoQuoteRequest** Business Object editor opening where the structure and fields of the object can be viewed and edited.



Each parameter can be a simple structure, another business object or a list.

Enumerations can be supported by making the business object a simple type selection as shown below. This will appear as a domain in any rule vocabulary allowing a selection to made from the choices available.

Double-click the **GenderType** business object under the **Data** icon to see an example.



Name: GenderType

Modified: tw_admin (Jun 5, 2011 12:24:21 P

Documentation:

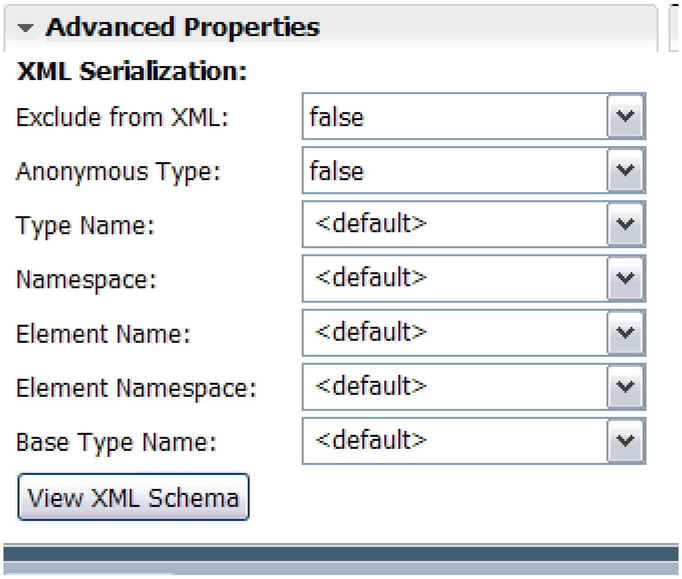
Definition Type: Simple Type

Type: Selection

Error Message:

Value	Display Text
male	male
female	female

Internally these information models are converted into schemas which are used later when interfacing to Integration Services and Rules. These schemas may be viewed in the advanced properties panel by pressing the **View XML Schema** button.



Exclude from XML: false

Anonymous Type: false

Type Name: <default>

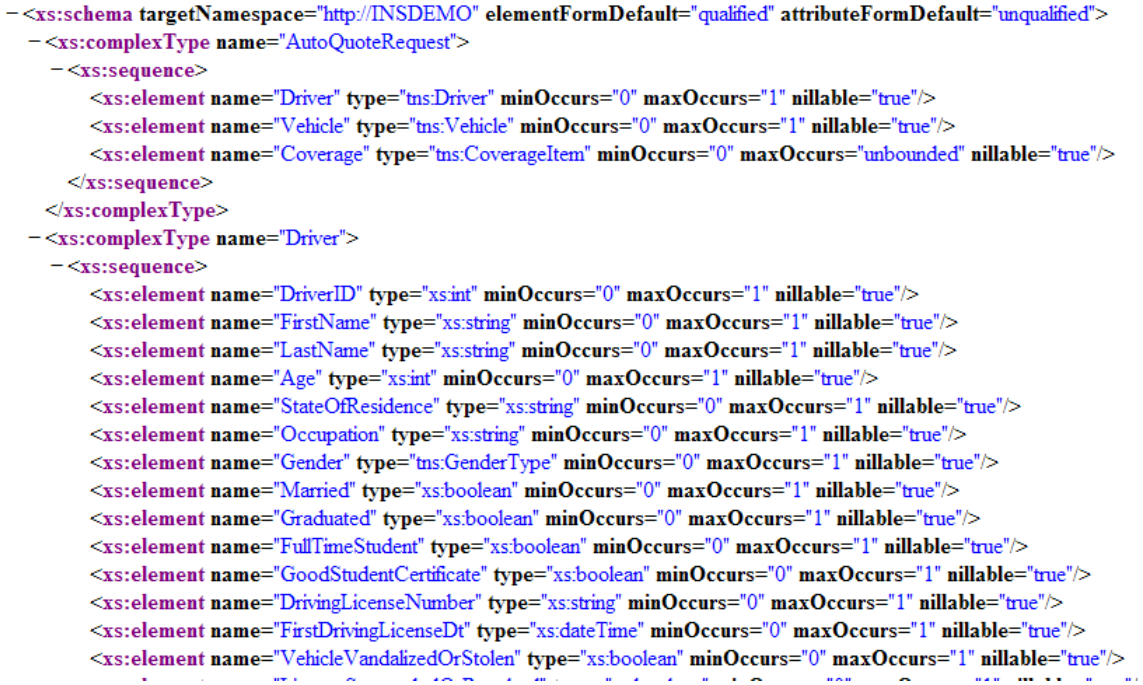
Namespace: <default>

Element Name: <default>

Element Namespace: <default>

Base Type Name: <default>

This results in a schema that can be viewed and saved as shown below. (Take business object **AutoQuoteRequest** as an example)

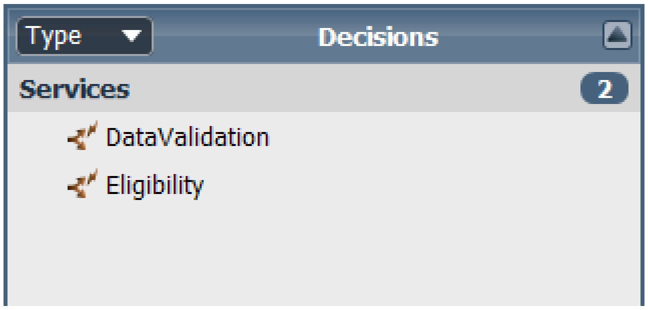


```
− <xs:schema targetNamespace="http://INSDEMO" elementFormDefault="qualified" attributeFormDefault="unqualified">
  − <xs:complexType name="AutoQuoteRequest">
    − <xs:sequence>
        <xs:element name="Driver" type="tns:Driver" minOccurs="0" maxOccurs="1" nillable="true"/>
        <xs:element name="Vehicle" type="tns:Vehicle" minOccurs="0" maxOccurs="1" nillable="true"/>
        <xs:element name="Coverage" type="tns:CoverageItem" minOccurs="0" maxOccurs="unbounded" nillable="true"/>
      </xs:sequence>
    </xs:complexType>
  − <xs:complexType name="Driver">
    − <xs:sequence>
        <xs:element name="DriverID" type="xs:int" minOccurs="0" maxOccurs="1" nillable="true"/>
        <xs:element name="FirstName" type="xs:string" minOccurs="0" maxOccurs="1" nillable="true"/>
        <xs:element name="LastName" type="xs:string" minOccurs="0" maxOccurs="1" nillable="true"/>
        <xs:element name="Age" type="xs:int" minOccurs="0" maxOccurs="1" nillable="true"/>
        <xs:element name="StateOfResidence" type="xs:string" minOccurs="0" maxOccurs="1" nillable="true"/>
        <xs:element name="Occupation" type="xs:string" minOccurs="0" maxOccurs="1" nillable="true"/>
        <xs:element name="Gender" type="tns:GenderType" minOccurs="0" maxOccurs="1" nillable="true"/>
        <xs:element name="Married" type="xs:boolean" minOccurs="0" maxOccurs="1" nillable="true"/>
        <xs:element name="Graduated" type="xs:boolean" minOccurs="0" maxOccurs="1" nillable="true"/>
        <xs:element name="FullTimeStudent" type="xs:boolean" minOccurs="0" maxOccurs="1" nillable="true"/>
        <xs:element name="GoodStudentCertificate" type="xs:boolean" minOccurs="0" maxOccurs="1" nillable="true"/>
        <xs:element name="DrivingLicenseNumber" type="xs:string" minOccurs="0" maxOccurs="1" nillable="true"/>
        <xs:element name="FirstDrivingLicenseDt" type="xs:dateTime" minOccurs="0" maxOccurs="1" nillable="true"/>
        <xs:element name="VehicleVandalizedOrStolen" type="xs:boolean" minOccurs="0" maxOccurs="1" nillable="true"/>
```

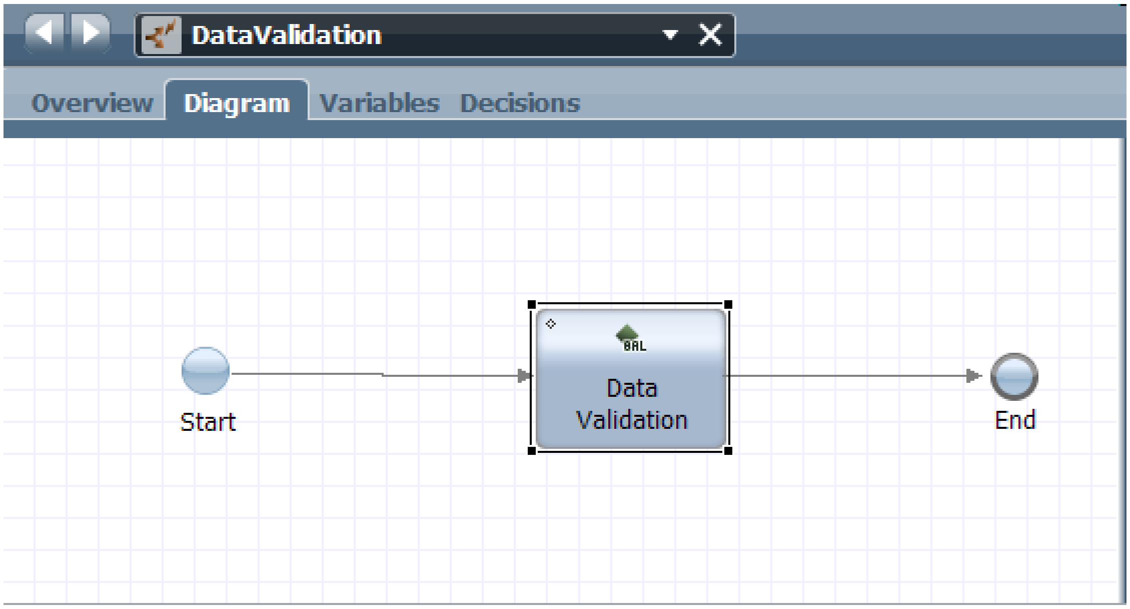
These schemas then form the basis of the information model for both decisions and processes.

Step 4. Definition of Decision services and associated vocabulary.

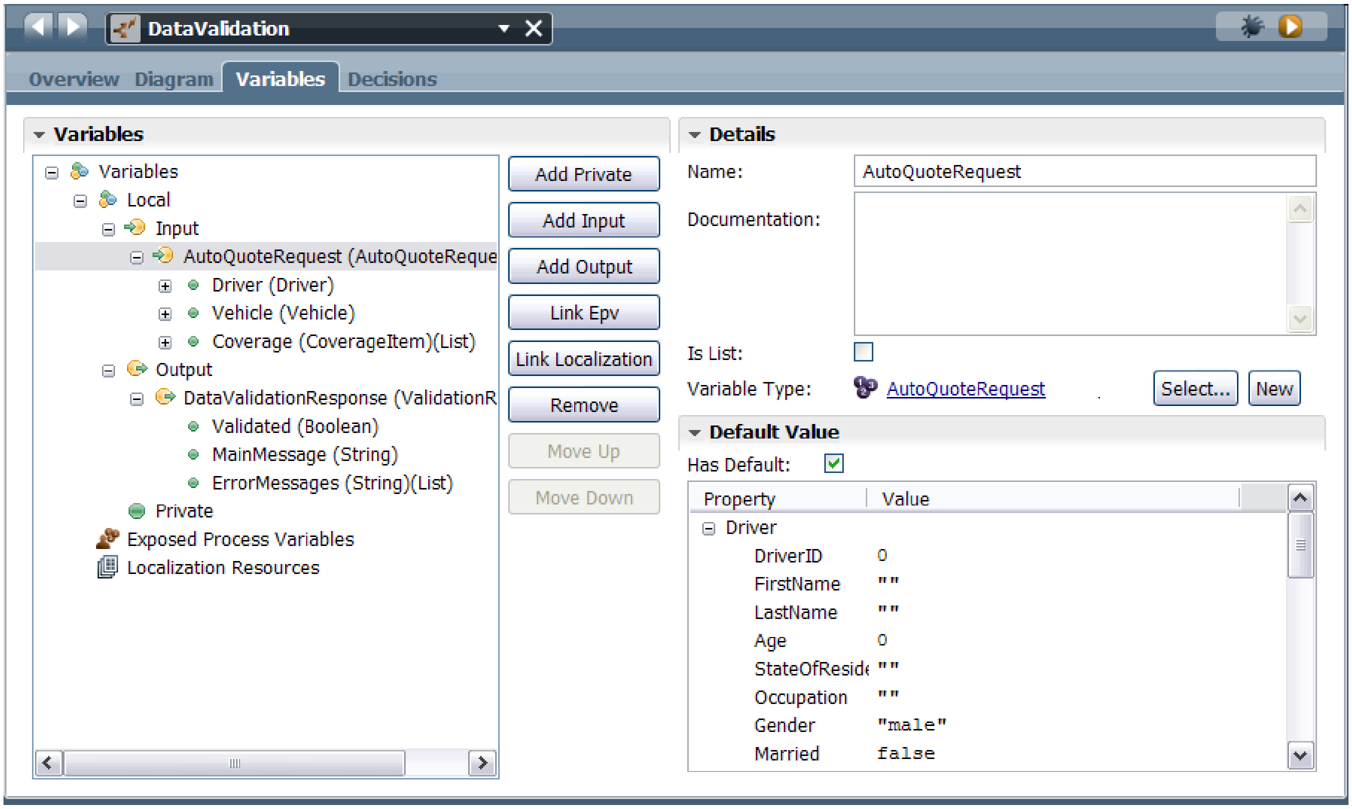
The decisions within the process are defined as decision services and may be examined by selecting the **Decisions** icon in the navigator.



This shows the two decision services that the Business Process Designer has created. Selecting each of these shows the flow that represents the decision service. For the **DataValidation** decision service, this includes a single BAL Rule called **Data Validation**. Each BAL Rule will result in its own rule project when exported.



The signature of the decision service is setup in the **Variables** tab. This references the Business Objects defined earlier and forms the basis for the vocabulary used when authoring rules. The parameters appear as variables in the rules while the business objects appear as object types within the vocabulary.



Variables

- Variables
  - Local
    - Input
      - AutoQuoteRequest (AutoQuoteReque
        - Driver (Driver)
        - Vehicle (Vehicle)
        - Coverage (CoverageItem)(List)
    - Output
      - DataValidationResponse (ValidationR
        - Validated (Boolean)
        - MainMessage (String)
        - ErrorMessages (String)(List)
    - Private
  - Exposed Process Variables
  - Localization Resources

Details

Name: AutoQuoteRequest

Documentation:

Is List: ☐

Variable Type: AutoQuoteRequest

Default Value

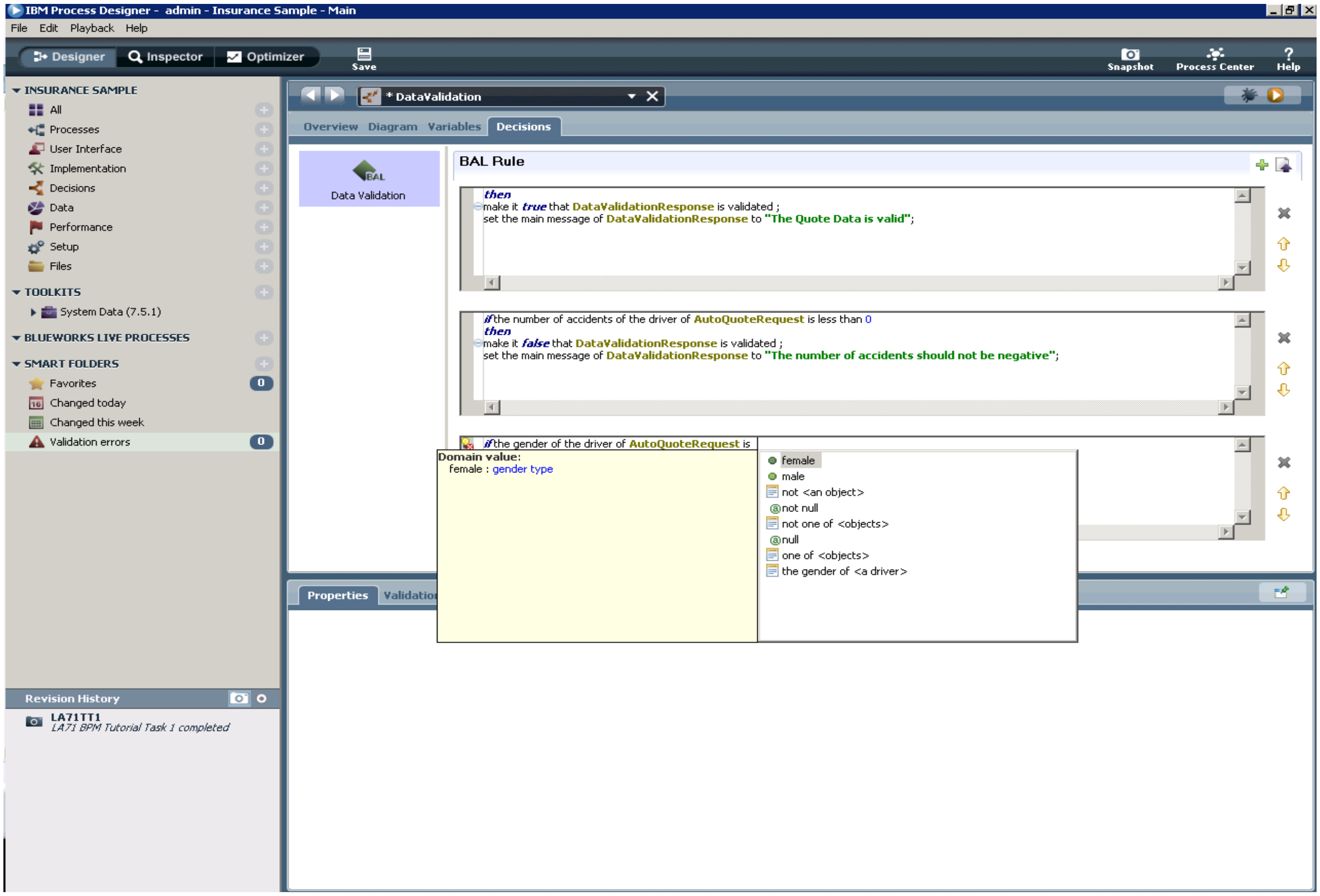
Has Default: ☑

Property	Value
Driver	
DriverID	0
FirstName	""
LastName	""
Age	0
StateOfResid	""
Occupation	""
Gender	"male"
Married	false

Step 5. Definition of the decision rules.

Now the vocabulary has been defined, the rule editor can be used to define the behavior required as shown below. The rule editor is context sensitive and prompts for possible terms to complete the rule including showing the possible values of domains as shown here.

Navigate to the **Decisions** tab which should show the **Data Validation** BAL Rule.
Add a new rule by clicking the **+** icon and then experiment with the intellirule editor.



```
then
make it true that DataValidationResponse is validated ;
set the main message of DataValidationResponse to "The Quote Data is valid";
```

```
if the number of accidents of the driver of AutoQuoteRequest is less than 0
then
make it false that DataValidationResponse is validated ;
set the main message of DataValidationResponse to "The number of accidents should not be negative";
```

```
if the gender of the driver of AutoQuoteRequest is
```

```
Domain value:
  female : gender type
```

```
female
male
not <an object>
not null
not one of <objects>
null
one of <objects>
the gender of <a driver>
```

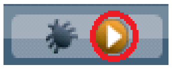
Delete the rule you created before progressing further in the tutorial.

Step 6. Testing the embedded rules decisions in the process

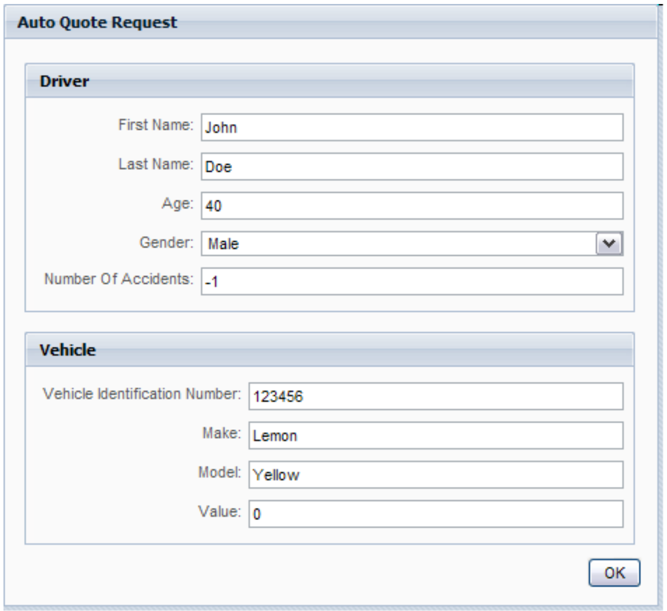
The Business Process Designer can test and iterate around the information models and the decisions that are required. Using the coaches the data required can be entered and the decisions tested.

In this tutorial you will modify the number of accidents to check each of the decisions. Select the **User Interface** icon and double-click on the **Insurance sample** human service.

Click the **Run** icon.



In the **Auto Quote Request** Coach, type **-1** in the **Number Of Accidents** field. Click **OK**, which should result in the **Data Validation** decision showing an invalid response.

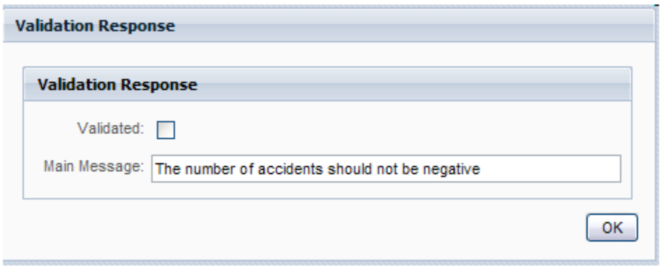

Auto Quote Request

Driver

First Name: John
Last Name: Doe
Age: 40
Gender: Male
Number Of Accidents: -1

Vehicle

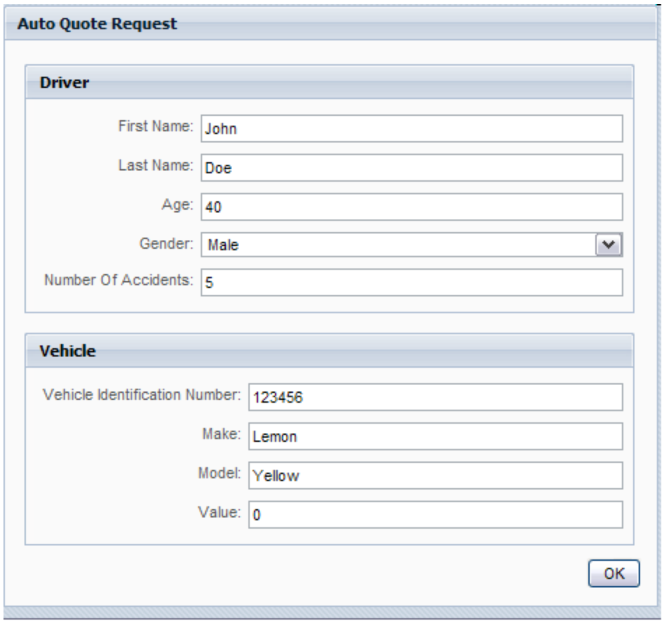
Vehicle Identification Number: 123456
Make: Lemon
Model: Yellow
Value: 0


Validation Response

Validation Response

Validated: ☐
Main Message: The number of accidents should not be negative

Press **OK** again to get back to the **Auto Quote Request** coach.
This time set the **Number of Accidents** to **5**. Click **OK** to move to the next screen, which should provide a valid response. However, when you click **OK** again to move to the next screen, the quote should not be eligible.

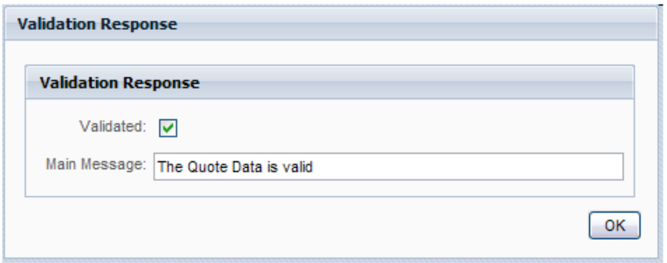

Auto Quote Request

Driver

First Name: John
Last Name: Doe
Age: 40
Gender: Male
Number Of Accidents: 5

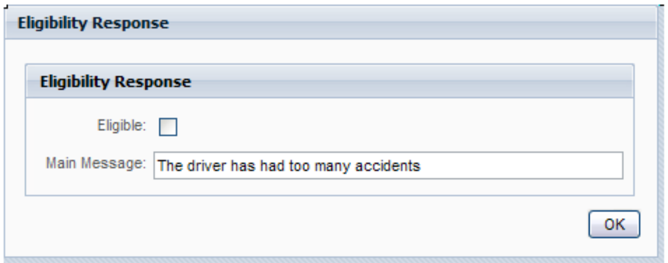
Vehicle

Vehicle Identification Number: 123456
Make: Lemon
Model: Yellow
Value: 0


Validation Response

Validation Response

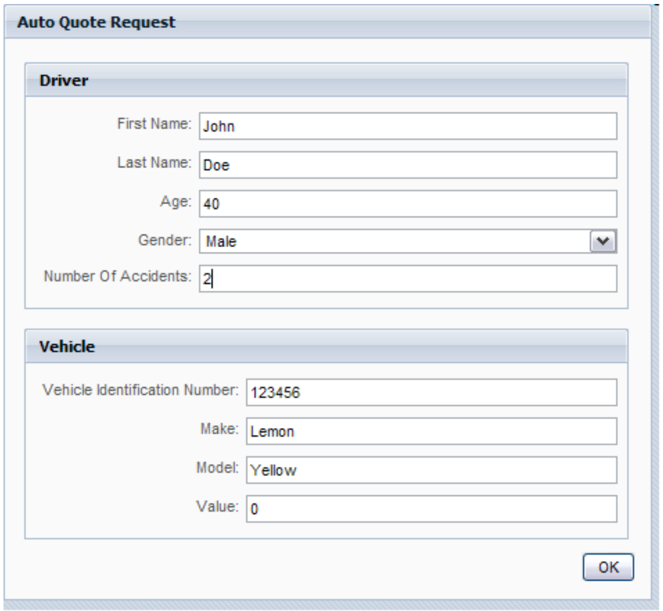
Validated: ☑
Main Message: The Quote Data is valid


Eligibility Response

Eligibility Response

Eligible: ☐
Main Message: The driver has had too many accidents

Finally if the **Number of Accidents** is set to **2**, the validation response should be validated and the quote should be eligible.



Auto Quote Request

Driver

First Name: John

Last Name: Doe

Age: 40

Gender: Male

Number Of Accidents: 2

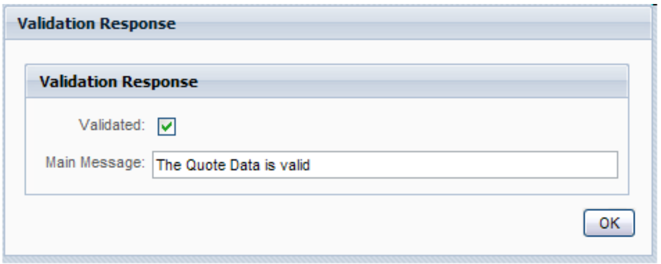
Vehicle

Vehicle Identification Number: 123456

Make: Lemon

Model: Yellow

Value: 0

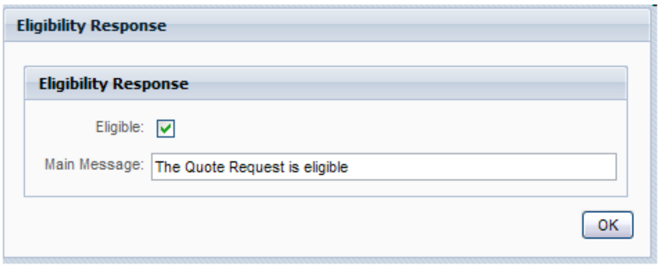


Validation Response

Validation Response

Validated: ☑

Main Message: The Quote Data is valid



Eligibility Response

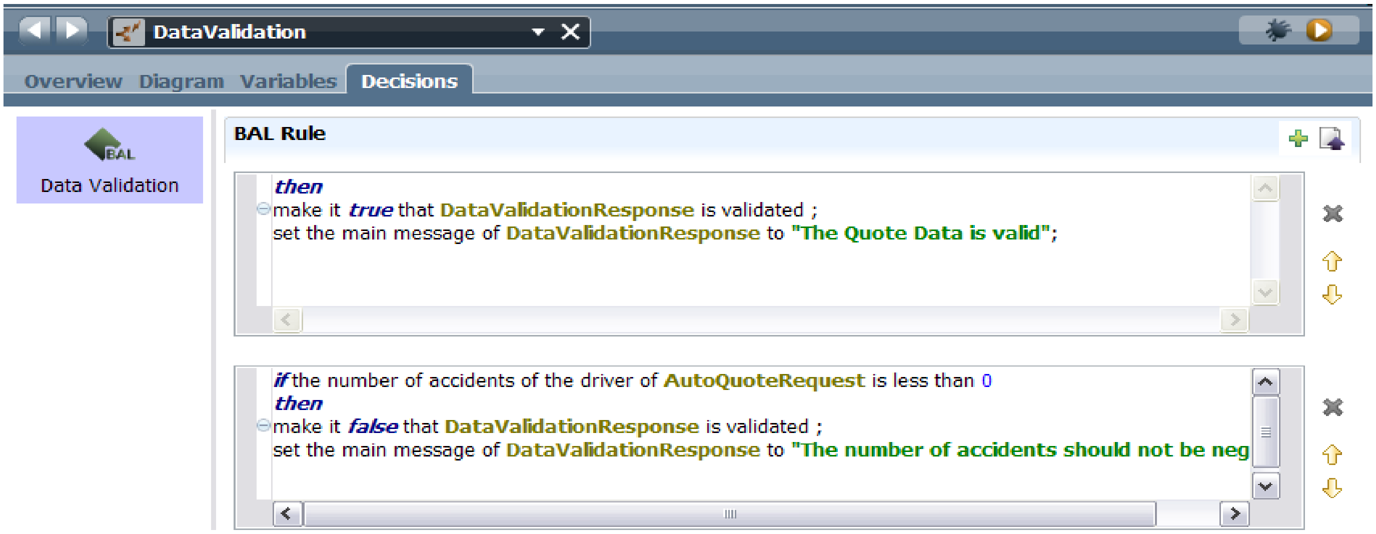
Eligibility Response

Eligible: ☑

Main Message: The Quote Request is eligible

Step 7. Export the decisions for IT to create a managed decision service.

Once the Business Process Author is satisfied that the decisions are correct each ruleset is then selected and exported as a project. Select the **Decisions** icon and double-click on the **DataValidation** decision service. Open the BAL Rules in the **Decisions** tab.



```
then
make it true that DataValidationResponse is validated ;
set the main message of DataValidationResponse to "The Quote Data is valid";
```

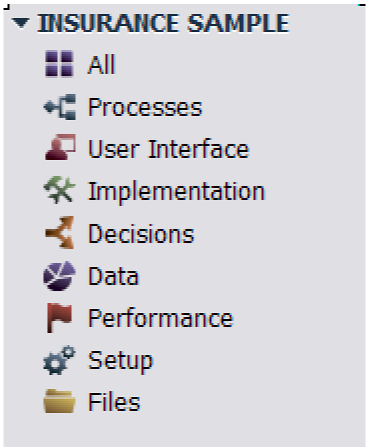
```
if the number of accidents of the driver of AutoQuoteRequest is less than 0
then
make it false that DataValidationResponse is validated ;
set the main message of DataValidationResponse to "The number of accidents should not be neg
```

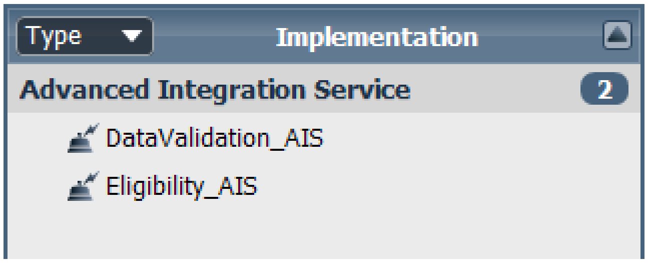
To export the ruleset press the **Export** icon and select the location to save the file.

These rules have already been saved to **[SupportPac LA71 Path]\BPMTutorial\task1** and will be needed for task 2.

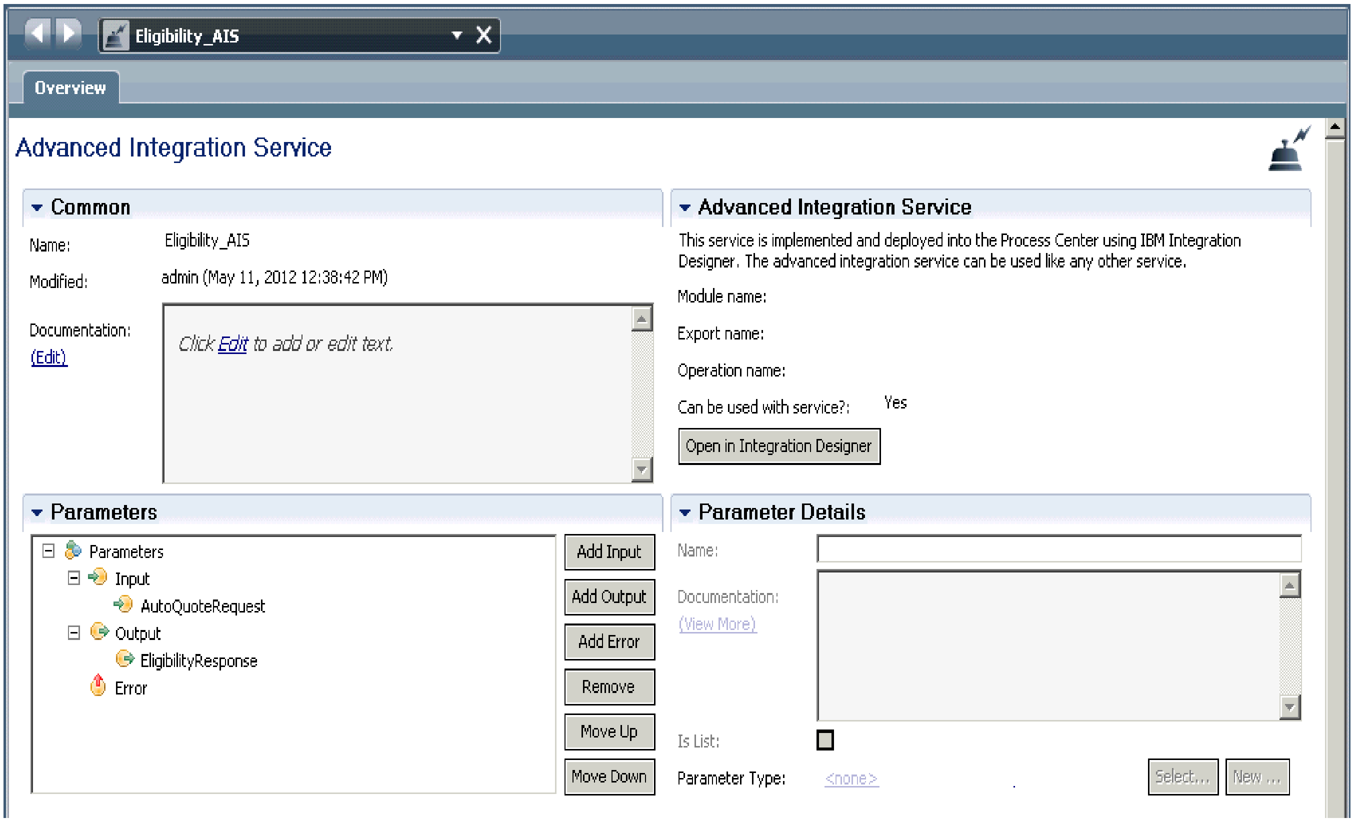
Step 8. Definition of Advanced Integration Service interfaces for the decisions.

The final step in the Business Process Authors role is to define an Advanced Integration Service that can utilize decisions running as SCA component in Process Server.
These service definitions are visible by selecting the **Implementation** icon.





Selecting an implementation then shows the interface for each of these services.



Name: Eligibility_AIS
Modified: admin (May 11, 2012 12:38:42 PM)
Documentation: (Edit) *Click Edit to add or edit text.*

This service is implemented and deployed into the Process Center using IBM Integration Designer. The advanced integration service can be used like any other service.

Module name:
Export name:
Operation name:
Can be used with service?: Yes

Name:
Documentation: (View More)
Is List: ☐
Parameter Type: <none>

When the IT developers open the AIS in an Integration Designer workspace, the implementation module, exports and operation are defined. Since the parameters are identical to the existing decision services, these AIS implementations can be easily interchanged with the current decision services.

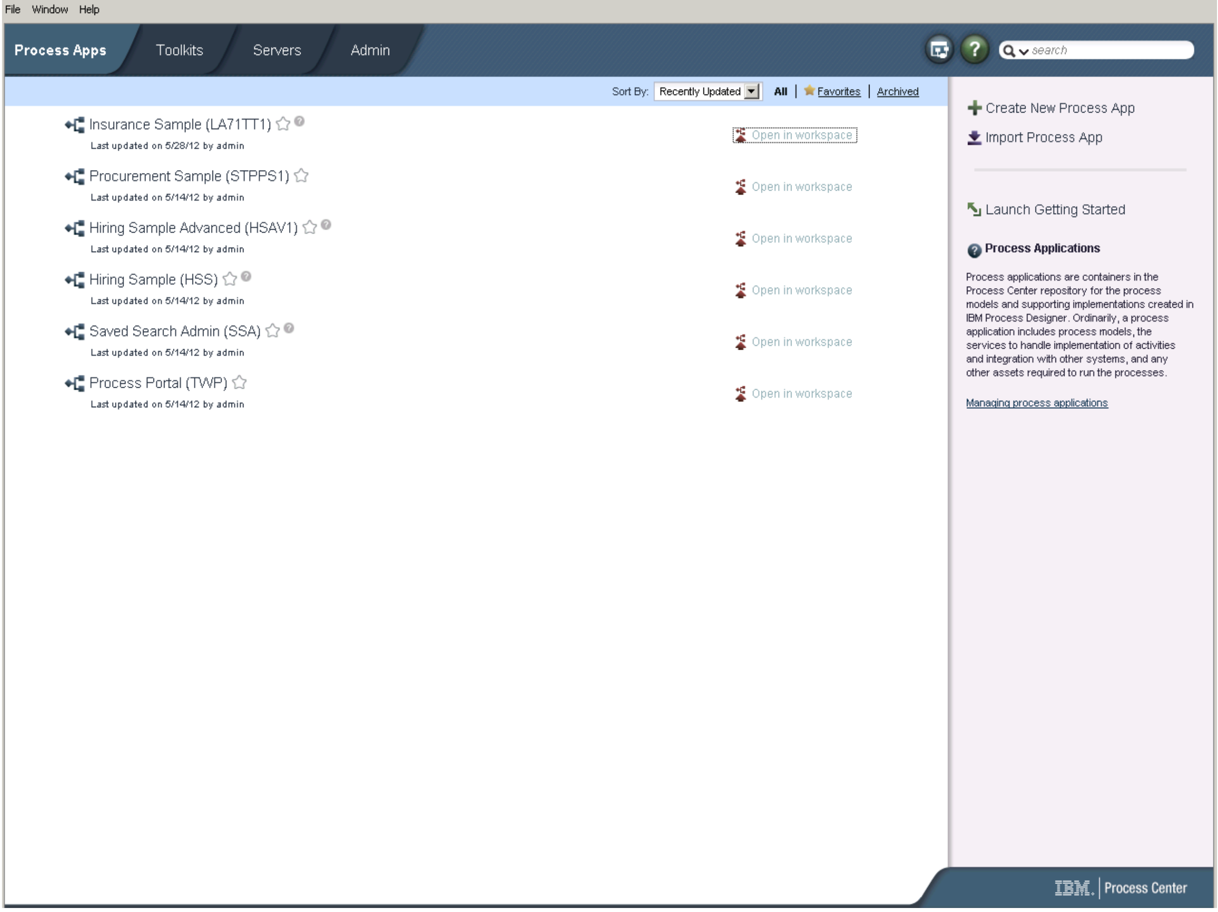
This completes the Business Process Authors definition of the decisions that they wish to manage within this business process. You will revisit the Business Process Author role when the decision implementations are completed.

Step 9. (Optional) Integration with Decision Wizard Plugin.

In this task you will create a RuleProject together with a Rule App that will form the implementation of a decision service that can be managed by LOB users in Decision Center.

You will use the Eligibility_AIS interface you created in the previous task to specify the decision information model and ruleset parameters for the decision ruleset.

1. Open IID with **Process Center** Perspective.



Insurance Sample (LA71TT1)
Last updated on 5/28/12 by admin
Open in workspace

Procurement Sample (STPPS1)
Last updated on 5/14/12 by admin
Open in workspace

Hiring Sample Advanced (HSAV1)
Last updated on 5/14/12 by admin
Open in workspace

Hiring Sample (HSS)
Last updated on 5/14/12 by admin
Open in workspace

Saved Search Admin (SSA)
Last updated on 5/14/12 by admin
Open in workspace

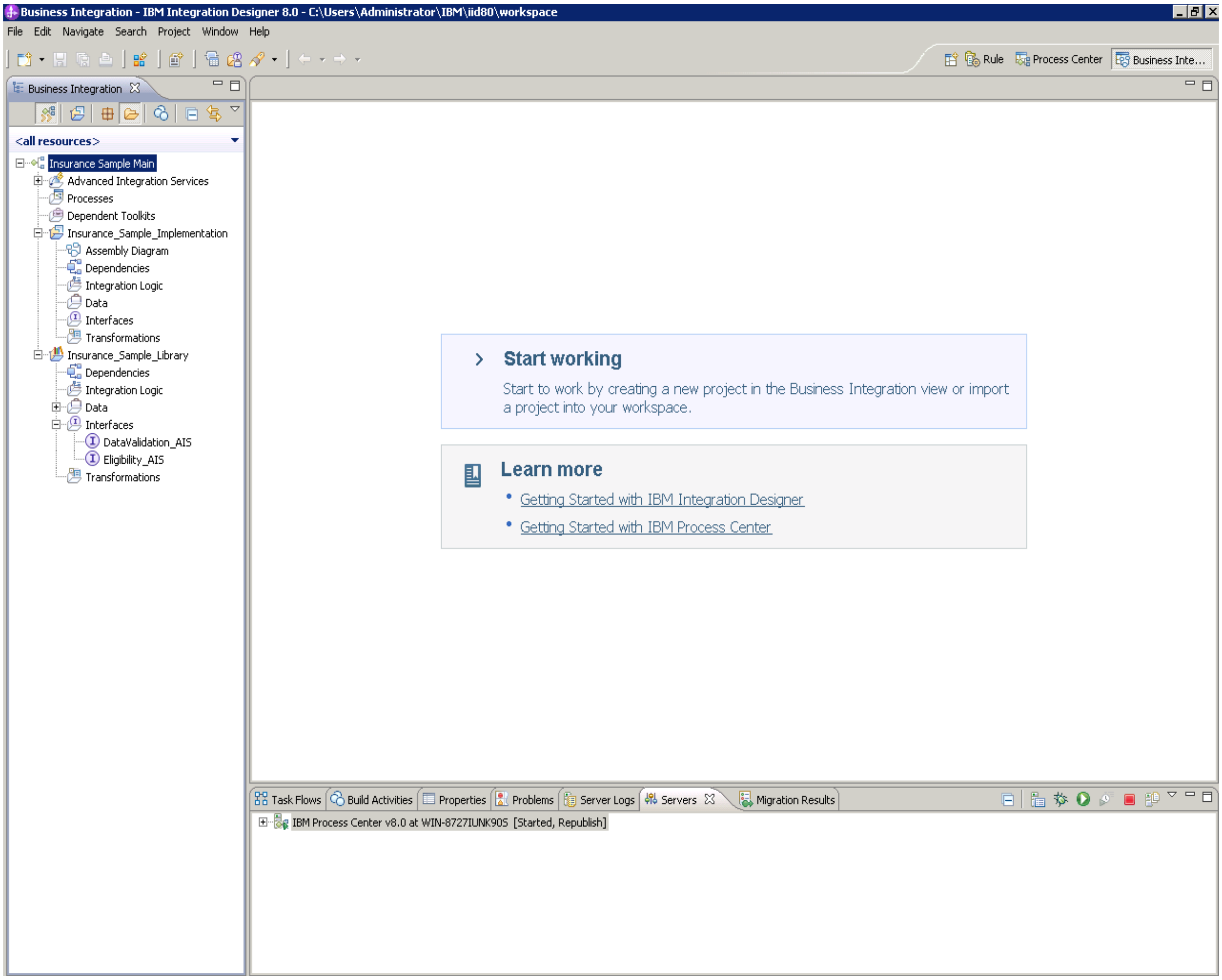
Process Portal (TWP)
Last updated on 5/14/12 by admin
Open in workspace

Process Applications

Process applications are containers in the Process Center repository for the process models and supporting implementations created in IBM Process Designer. Ordinarily, a process application includes process models, the services to handle implementation of activities and integration with other systems, and any other assets required to run the processes.

Managing process applications

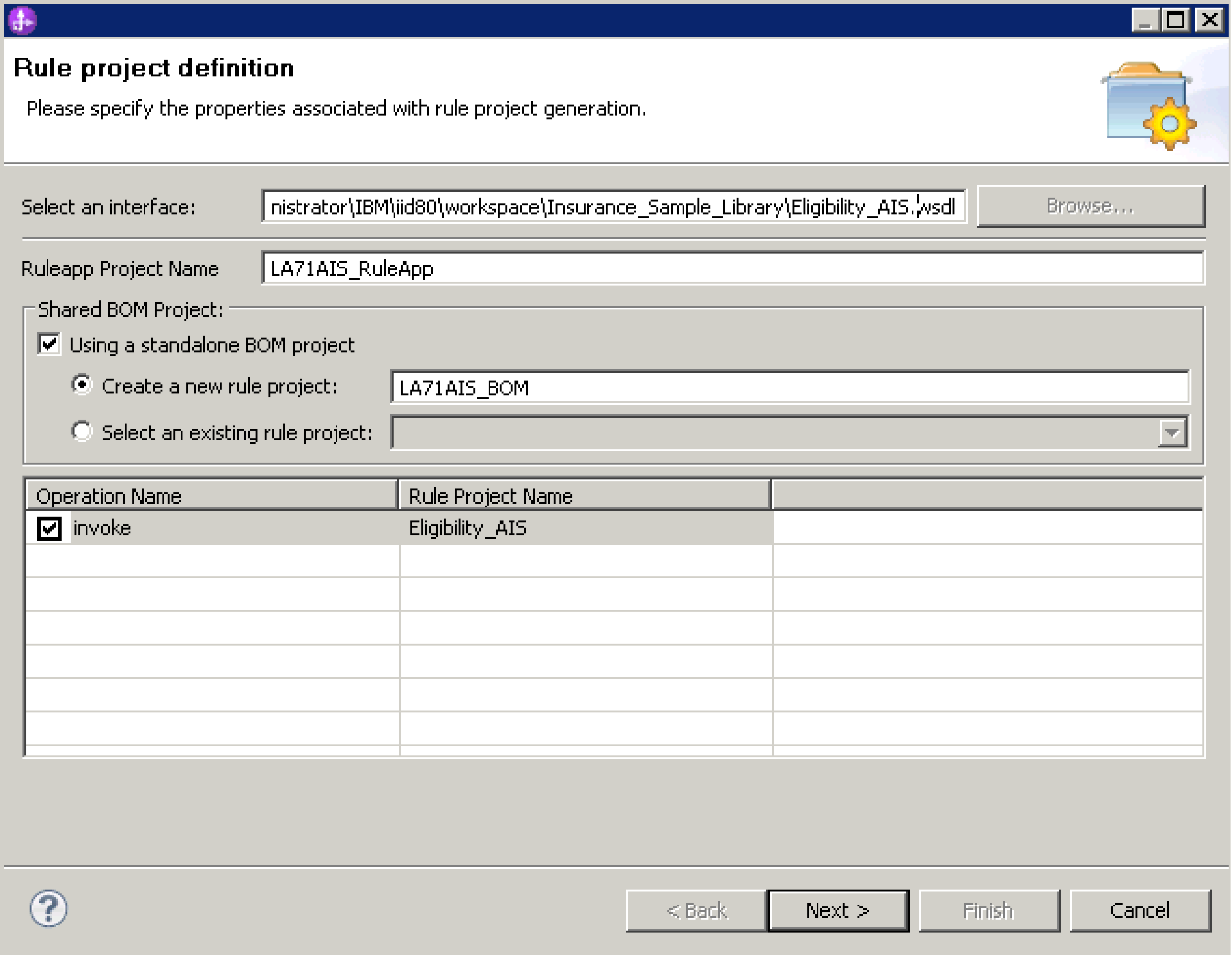
2. Open the **Insurance Sample (LA71TT1)** Process App in workspace.



Start to work by creating a new project in the Business Integration view or import a project into your workspace.

3. Switch to **Rule** Perspective and right click on **Eligibility_AIS.wsdl** in **Insurance_Sample_Library** project, and click **SupportPac LA71** > **Create Decision Service.**

4. Input the Ruleapp Project Name as **LA71AIS_RuleApp**, and new BOM project name as **LA71AIS_BOM.** Check the **invoke** operation, and update the corresponding rule project name as **Eligibility_AIS**.



Rule project definition

Please specify the properties associated with rule project generation.

Select an interface: nistrator\IBM\iid80\workspace\Insurance_Sample_Library\Eligibility_AIS.wsdl

Ruleapp Project Name: LA71AIS_RuleApp

Shared BOM Project:

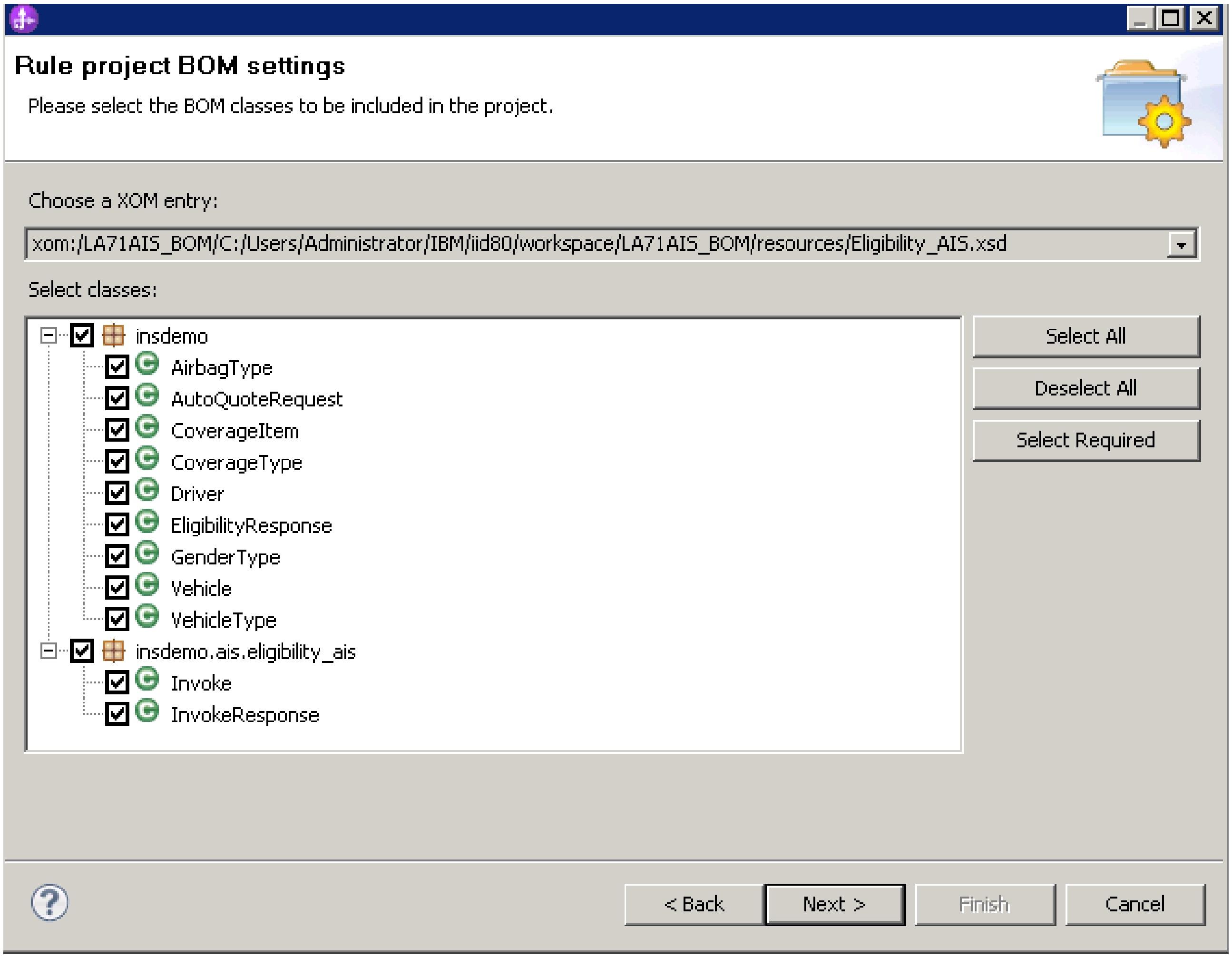
☑ Using a standalone BOM project

● Create a new rule project: LA71AIS_BOM

○ Select an existing rule project:

Operation Name	Rule Project Name
☑ invoke	Eligibility_AIS

5. Click **Next>** button.



Rule project BOM settings

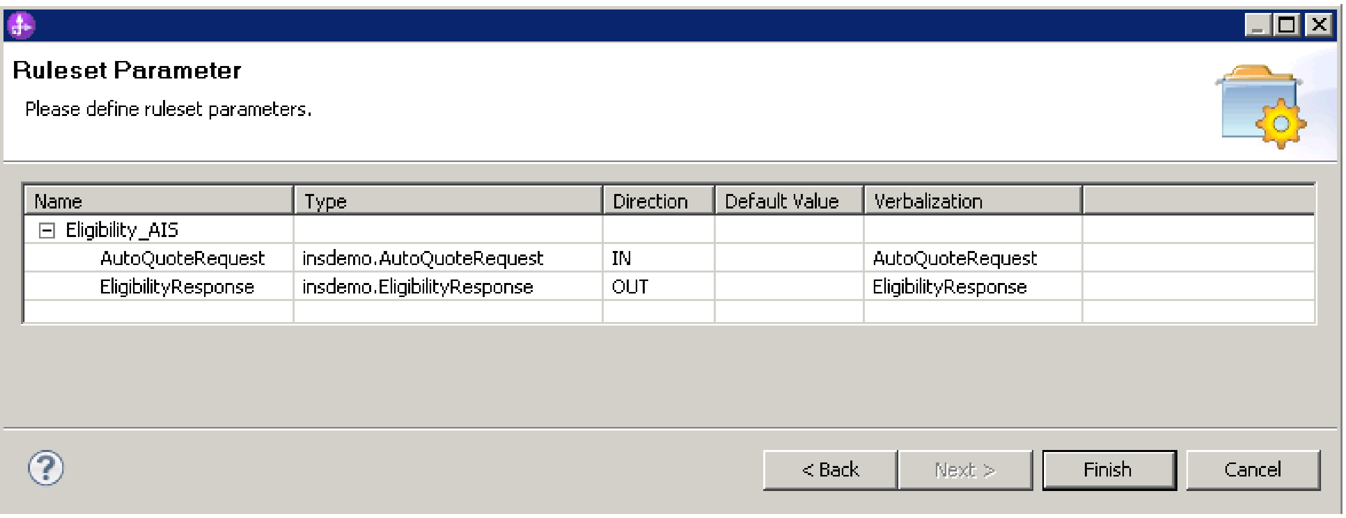
Please select the BOM classes to be included in the project.

Choose a XOM entry: xom:/LA71AIS_BOM/C:/Users/Administrator/IBM/iid80/workspace/LA71AIS_BOM/resources/Eligibility_AIS.xsd

Select classes:

- ☑ insdemo
  - ☑ AirbagType
  - ☑ AutoQuoteRequest
  - ☑ CoverageItem
  - ☑ CoverageType
  - ☑ Driver
  - ☑ EligibilityResponse
  - ☑ GenderType
  - ☑ Vehicle
  - ☑ VehicleType
- ☑ insdemo.ais.eligibility_ais
  - ☑ Invoke
  - ☑ InvokeResponse

Deselect the **insdemo.ais.eligibility_ais** package and click **Next >** button.



Ruleset Parameter

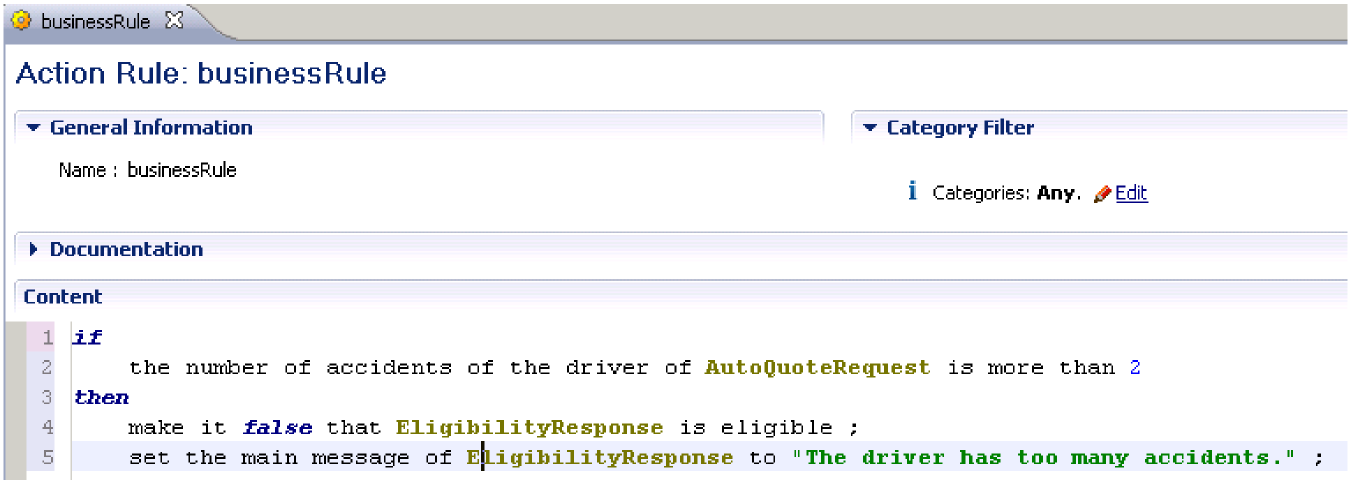
Please define ruleset parameters.

Name	Type	Direction	Default Value	Verbalization	
⊟ Eligibility_AIS					
AutoQuoteRequest	insdemo.AutoQuoteRequest	IN		AutoQuoteRequest	
EligibilityResponse	insdemo.EligibilityResponse	OUT		EligibilityResponse	

Click **Finish** Button to complete generating the projects.

6. Open the **Eligibility_AIS** rule project.

Open **businessRule** under **rules** in package **Eligibility_AIS**, and author the rule as follows:



businessRule

Action Rule: businessRule

General Information

Name : businessRule

Category Filter

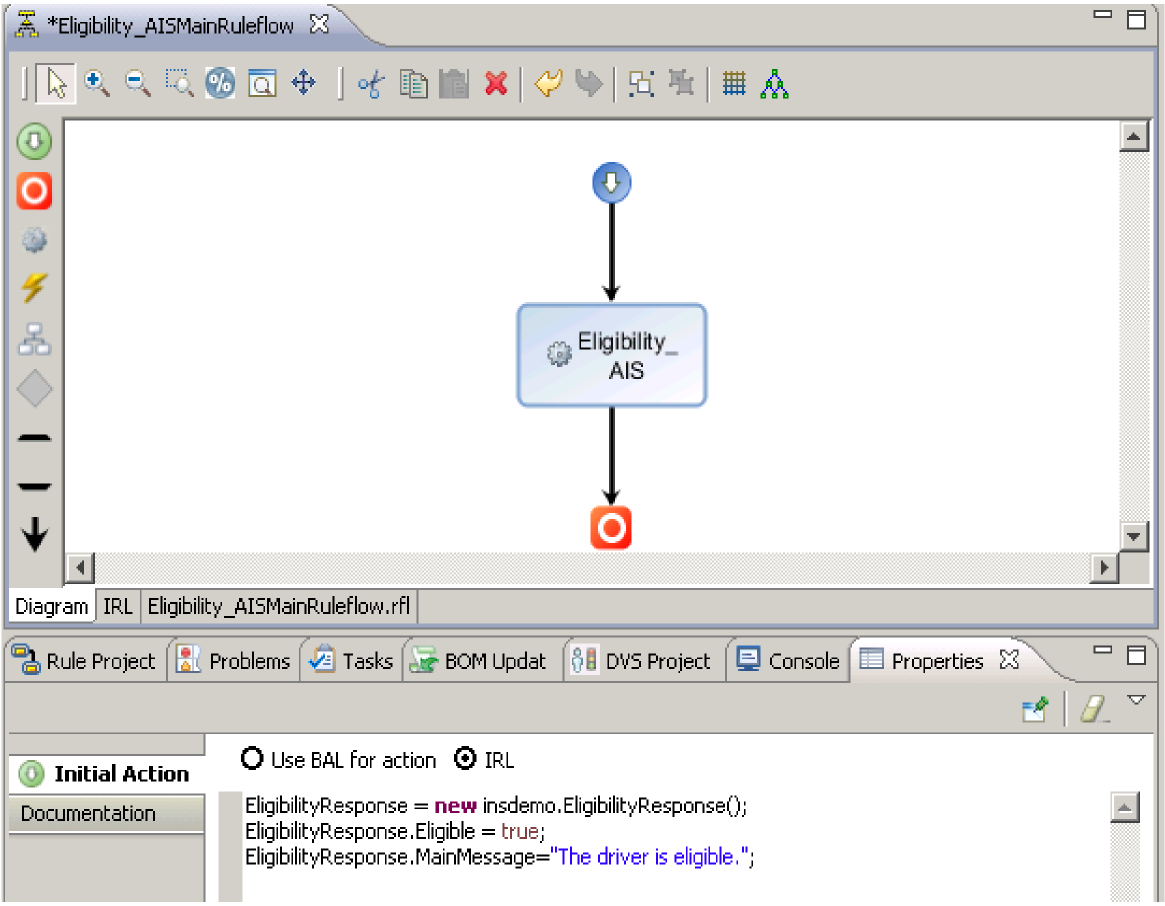
Categories: **Any**. Edit

Documentation

Content

```
if
    the number of accidents of the driver of AutoQuoteRequest is more than 2
then
    make it false that EligibilityResponse is eligible ;
    set the main message of EligibilityResponse to "The driver has too many accidents." ;
```

Save the rule.
Open the **Eligibility_AISMainRuleflow** under **rules**.
Add **Initial Action** in the properties view of **Start Node**. Select the **IRL** radio button.
Input the content as



Initial Action

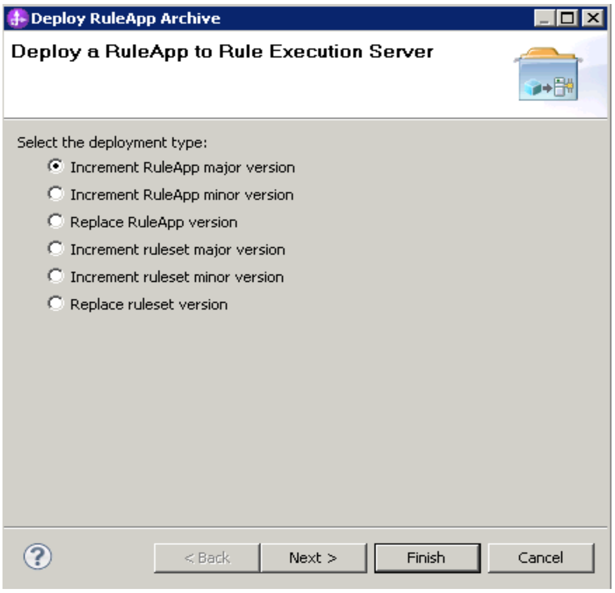
○ Use BAL for action ● IRL

```
EligibilityResponse = new insdemo.EligibilityResponse();
EligibilityResponse.Eligible = true;
EligibilityResponse.MainMessage="The driver is eligible.";
```

Save it.

7. Deploy the **LA71AIS_RuleApp** into Rule Execution Server and test the **invoke** ruleset.

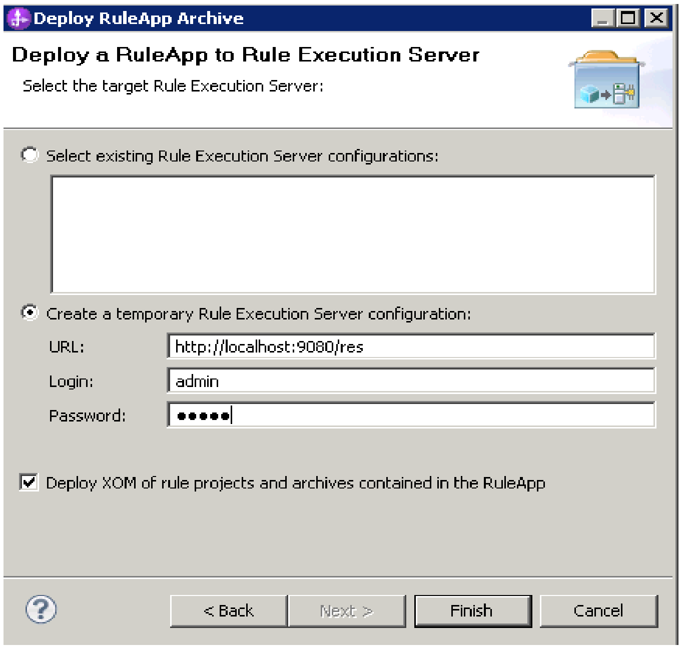
   First, right-click **LA71AIS_RuleApp** project and select **RuleApp** > **Deploy.**



Select the deployment type:

● Increment RuleApp major version
○ Increment RuleApp minor version
○ Replace RuleApp version
○ Increment ruleset major version
○ Increment ruleset minor version
○ Replace ruleset version

Click **Next >** Button.



Select the target Rule Execution Server:

○ Select existing Rule Execution Server configurations:

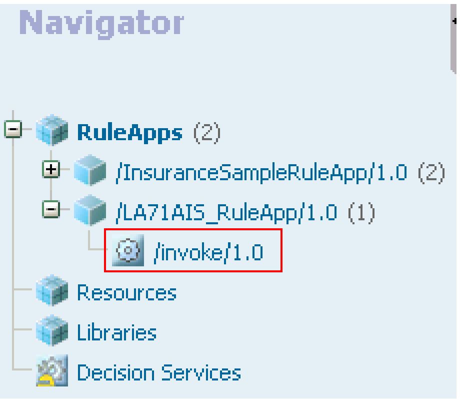
● Create a temporary Rule Execution Server configuration:

URL: http://localhost:9080/res
Login: admin
Password: •••••

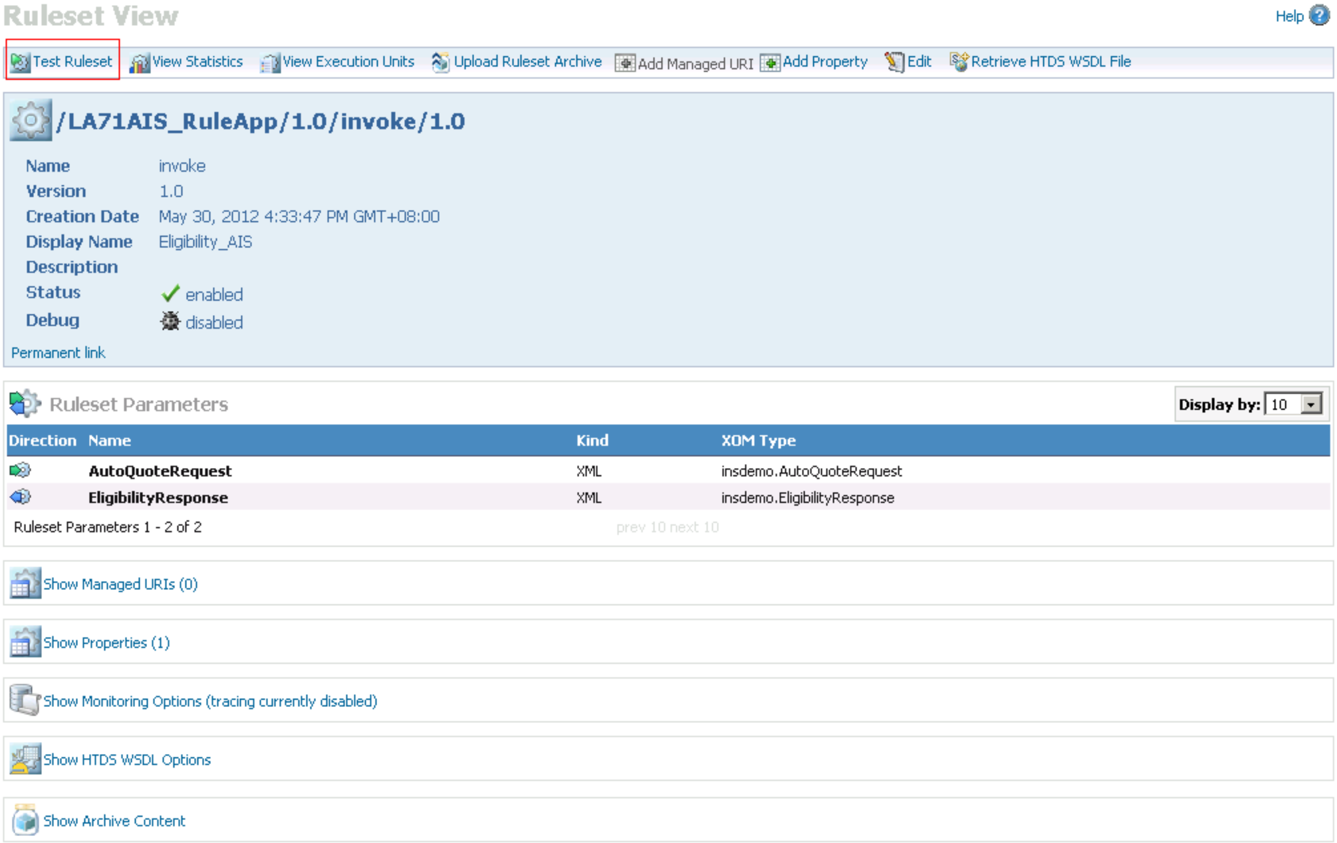
☑ Deploy XOM of rule projects and archives contained in the RuleApp

Click **Finish** Button.

Login to the RES Console at http://localhost:9080/res. Click on the **Explorer** tab, navigate to the **invoke** Ruleset, and click on the Ruleset to show the Ruleset View.



Click the **Test Ruleset** button in the Ruleset View.



/LA71AIS_RuleApp/1.0/invoke/1.0

Name	invoke
Version	1.0
Creation Date	May 30, 2012 4:33:47 PM GMT+08:00
Display Name	Eligibility_AIS
Description	
Status	✓ enabled
Debug	disabled

Permanent link

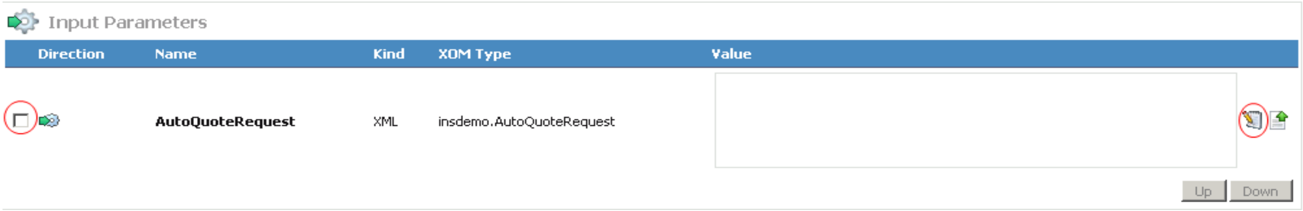
Ruleset Parameters

Display by: 10

Direction	Name	Kind	XOM Type
	AutoQuoteRequest	XML	insdemo.AutoQuoteRequest
	EligibilityResponse	XML	insdemo.EligibilityResponse

Ruleset Parameters 1 - 2 of 2 prev 10 next 10

Check the Input parameter **AutoQuoteRequest** and click the **Edit Model** button.

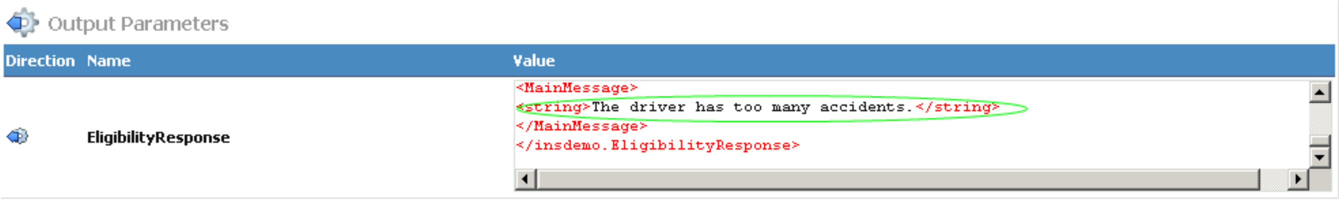


Input Parameters

Direction	Name	Kind	XOM Type	Value
☐	**AutoQuoteRequest**	XML	insdemo.AutoQuoteRequest	

Paste the content from AutoQuoteRequest.xml in <ODM_HOME>/SupportPacLA71v2.1/BPMTutorial/task1 into the text area.

Click **Execute** button to execute the test.
After the execution finished, check the **EligibilityResponse** in Output Parameters.

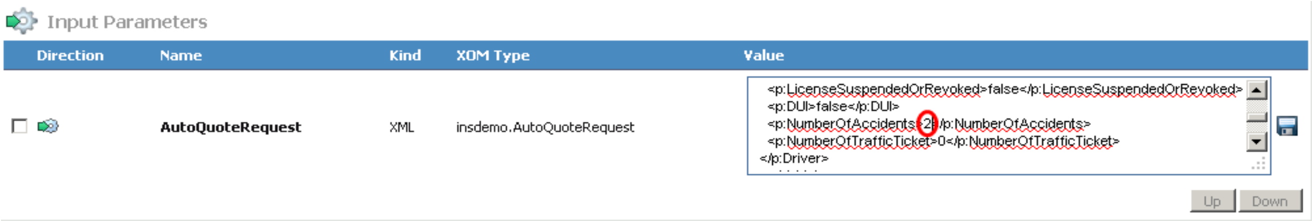


Output Parameters

Direction	Name	Value
	EligibilityResponse	

```
<MainMessage>
<string>The driver has too many accidents.</string>
</MainMessage>
</insdemo.EligibilityResponse>
```

Click the **Edit Model** again, update the **NumberOfAccidents** to 2, **Execute** again.

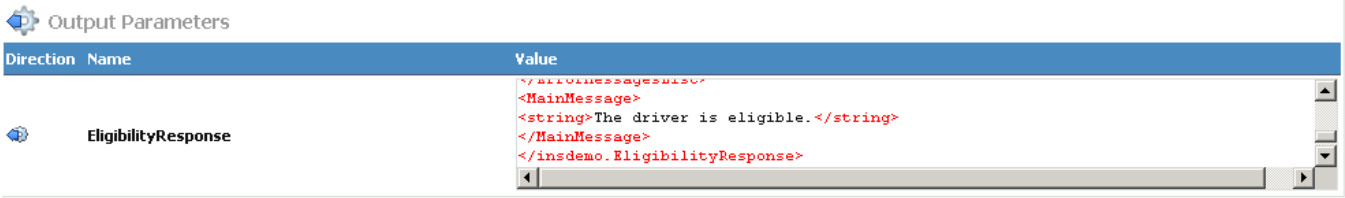


Input Parameters

Direction	Name	Kind	XOM Type	Value
☐	AutoQuoteRequest	XML	insdemo.AutoQuoteRequest	

```
  <p:LicenseSuspendedOrRevoked>false</p:LicenseSuspendedOrRevoked>
  <p:DUI>false</p:DUI>
  <p:NumberOfAccidents>2</p:NumberOfAccidents>
  <p:NumberOfTrafficTicket>0</p:NumberOfTrafficTicket>
</p:Driver>
```

After the execution finished, check the EligibilityResponse in Output Parameters.



Output Parameters

Direction	Name	Value
	EligibilityResponse	

```
<MainMessage>
<string>The driver is eligible.</string>
</MainMessage>
</insdemo.EligibilityResponse>
```