IBM Software
zSeries

WEBSPHERE DECISION SERVER

WAS z/OS -CICS Events - WBE & ILOG BRMS Connectivity and Integration on System z

Lab Exercises

IBM

An IBM Proof of Technology

Author	: ASSOU Alipoe Yao
Email	: yao_assou@fr.ibm.com

Date	: April 2011
Version	: V001

Contents

Lab 1 CICS Event Processing

1.1 Overview

The purpose of this lab exercise is to illustrate some of the capabilities of the CICS Event Processing Support. This lab exercise is based on the CICS Sample application managing a record in VSAM File. This application is a simple create, browse and delete record in the file. The Event Processing scenario implemented in this lab exercise implements a delete record scenario which can cause a damage in terms of business.
Before implementing this CICS Event Processing lab exercise you should be familiar with the Sample application and have tested to see that it is active on your system.

1.1.1 CICS Event Processing

A business event represents business conditions such as a large stock trade or an extremely large item sale. Event Processing allows you to be notified of these business conditions so you can take appropriate action.
The notification of the specific business condition is called emitting an event.
CICS provides a non-invasive way to signal most business conditions or you can add an EXEC CICS SIGNAL EVENT command to your application to allow CICS to emit an event for a program-detected business condition.
With event processing in CICS, you specify

- When an event should be emitted (i.e. under what conditions)
- The content of the event (any application data that should be inserted into the event)
- To where and in what format the event should be emitted

Definition of events is performed using the 'Event Binding Editor' in the CICS Explorer or RDz V7.6, and is intended to be straight-forward to use. Conceptually, it allows a business analyst to specify the business view of an event, and an application analyst to relate this to processing within the application.
Once events are defined, event bindings are deployed to your CICS system as part of a CICS Bundle.
Depending on your CICS system configuration, you can use either the CICS Explorer or RDz to deploy event bindings to your CICS system.
An event binding is an XML file that defines one or more business events to CICS. A business event is represented by an event specification which is composed of multiple parts. The parts are created and edited in the CICS event binding multi-tabbed editor.
The event binding is a grouping of events and is the unit for formatting, routing enabling, and disabling CICS events.
An event specification describes an event and its data in business terms. An event specification is associated with a capture specification which indicates when/where the event should be emitted in application terms. The overall event binding has an event processing adapter specification that indicates where and in what format the events in the binding should be emitted.

CICS can emit events at a number of clearly defined points known as capture points. A capture point is just a potential place where an event may be emitted. Capture points are provided before and after selected EXEC CICS API commands and at program start. Capture points are indicated in the capture specification. Once the capture point is indicated you can add filter criteria.

One way of thinking about the development of an event binding is that the business analyst creates an event specification and the application analyst then completes one or more corresponding capture specifications to indicate how CICS can capture the required event. In practice, the business analyst might supply the application analyst with information about the events required, and the application analyst will then use the Event Binding Editor tooling to create the event specification and its capture specification(s).
The CICS Explorer or RDz can be used as the development environment for CICS Event processing.
Event bindings are placed in a .evbind file within a bundle project. You might want multiple event bindings because the various events are to be formatted and emitted in different ways. You can deploy the bundle into CICS where you can install, enable, disable, and uninstall the bundle. Once a bundle containing event bindings is installed into a CICS region you can use the CICS Explorer, RDz, the CICSPlex SM Web User

Interface, or CEMT to display the BUNDLE resource and the EVENTBindings. Using the CICS Explorer or the CICSPlex SM WUI, you can also view the capture specifications that are enabled as a result of installing an event binding.
CICS can emit events in WBE format, in CBE format, to a CICS TS Queue, start a transaction, or you can emit events in any format by implementing user-written code.

1.2 Scenario

A business event is created by CICS Transaction Server when ANY record is deleted from file using program and Transaction defined.
The event is then passed to a queue on MQ Series
The criteria and actions are defined using functionality embedded within the CICS Explorer product.
The COBOL copybook reflect application data that will be emitted as CICS events as this structure :

- Program name
- User id of the person updating the account
- Customer Record No.
- Customer Name
- Customer Wage
- Record Deletion:

CICS transaction DT01 is executed, using DELETE RECORD action request, based on a customer record number.

CICS emits the following events (and their values) to MQ; destination WBE

- Program name
- User id of the person updating the account
- Customer Record No.
- Customer name
- Customer Wage

Action: DELETE customer record

WBE consumes the events from MQ ,DETECTs customer record UPDATE request from the business events that were sent from CICS

User id of the person updating the account
Customer Record No.
Customer Wage > $15,000

Action: **UPDATE** customer record; **Customer Wage** value

Based on the **UPDATE** request and **Customer Wage > $15,000**, WBE, generates a **web-service** request to ILOG - BRMS for rules determination
Data sent to ILOG

- **User id** of the person updating the account
- **Customer Record No**.
- **Customer Name**
- **Customer Wage** > <**variable input**> (any value higher than $15,000)

ILOG JRULE uses the **Customer Wage** value to determine the new promotion status:

- Calculate customer level
- Calculate interest premium
- calculate interest rate

1.2.1 Calculate customer level

The customer level is identified based on the average balance, or wage amount maintained in the bank or the number of years of relationship with the bank, as shown in this table :

Customer since (in years)	Average balance or Wage amount (in $, €, £)	**Customer Level**
> 10	> 100k	**P (Platinum)**
> 5 <10	> 50k <100k	**G (Gold)**
> 3 <5	> 30k < 50k	**S (Silver)**
Other	Other	**R (Regular)**

1.2.2 Calculate interest premium

Depending on the customer level, the interest premium is defined, as shown in this table :

Customer Level	**Interest Premium (in %)**
P	**0**
G	**0.25**
S	**0.5**
R	**0.75**

1.2.3 Calculate interest rate

Depending on the customer credit rating, loan amount and duration of loan, the interest rate is added to the base interest;

Credit Rating	Principle (in $, €,£)	Terms (in years)	**Add interest rate (in %)**
> 540 < 700	< 430 k	15	**1.90**
		20	**2.00**
		30	**2.50**
	> 430	15	**1.95**
		20	**2.05**
		30	**3.00**
> 700 < 830	< 430	15	**1.00**
		20	**1.5**
		30	**1.8**
	> 430	15	**1.25**
		20	**1.75**
		30	**1.85**

1.3 Generating CICS Events

In this lab we setup the following procedures:

_ Creating a BUNDLE project

_ Creating an Event Binding

_ Specifying adapters

_ Installing a bundle definition

_ Testing an event specification

1.3.1Creating the HFS directories

Use the TSO ISHELL to create a HFS directory, to which we later export the event binding file.

Example : Created HFS directory

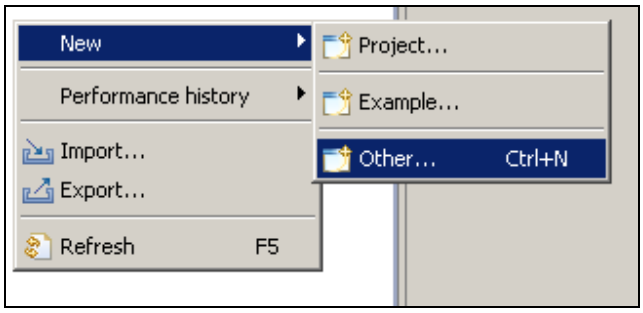
/etc/cicscfg/sharedir/suci1xx/bundles/ ==> *Where **xx** is your team number*

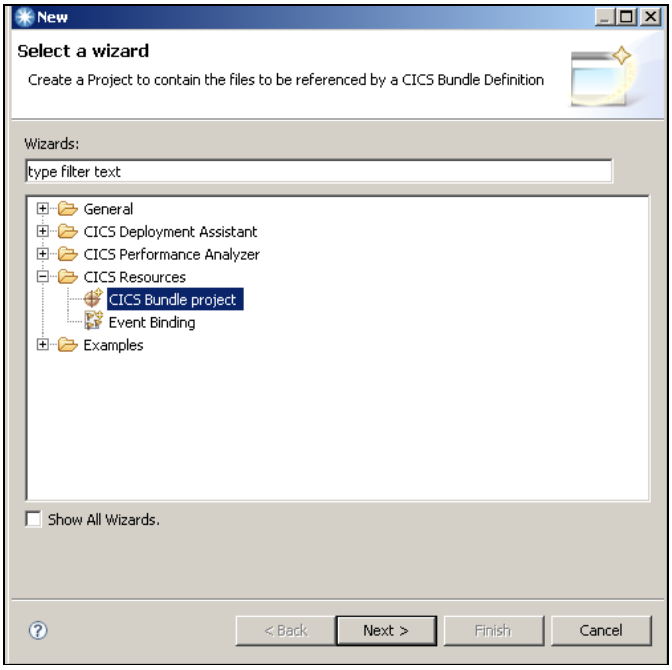
1.3.2Creating the bundle project

We use IBM CICS Explorer to create the event binding file.

1. Open IBM CICS Explorer, and select the resource perspective. Right-click in

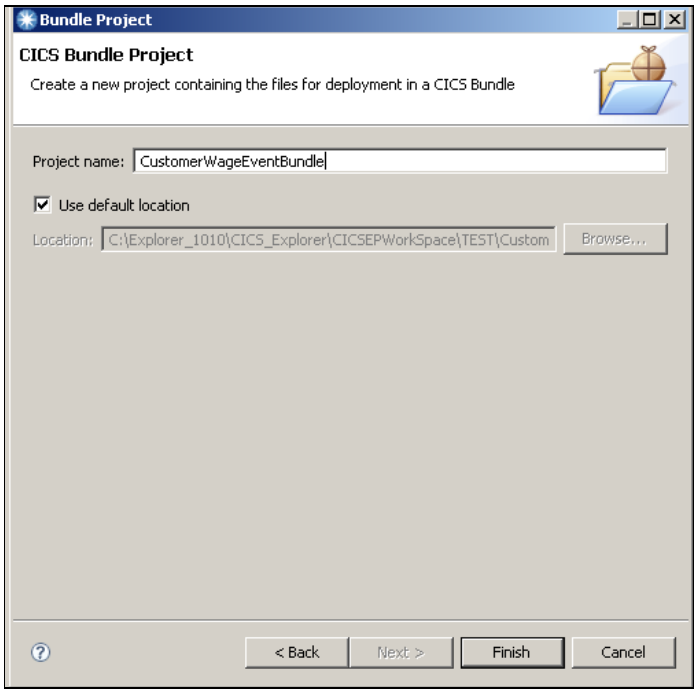
the Project Explorer view, click **New**, **Other**... and click **CICS Bundle project**





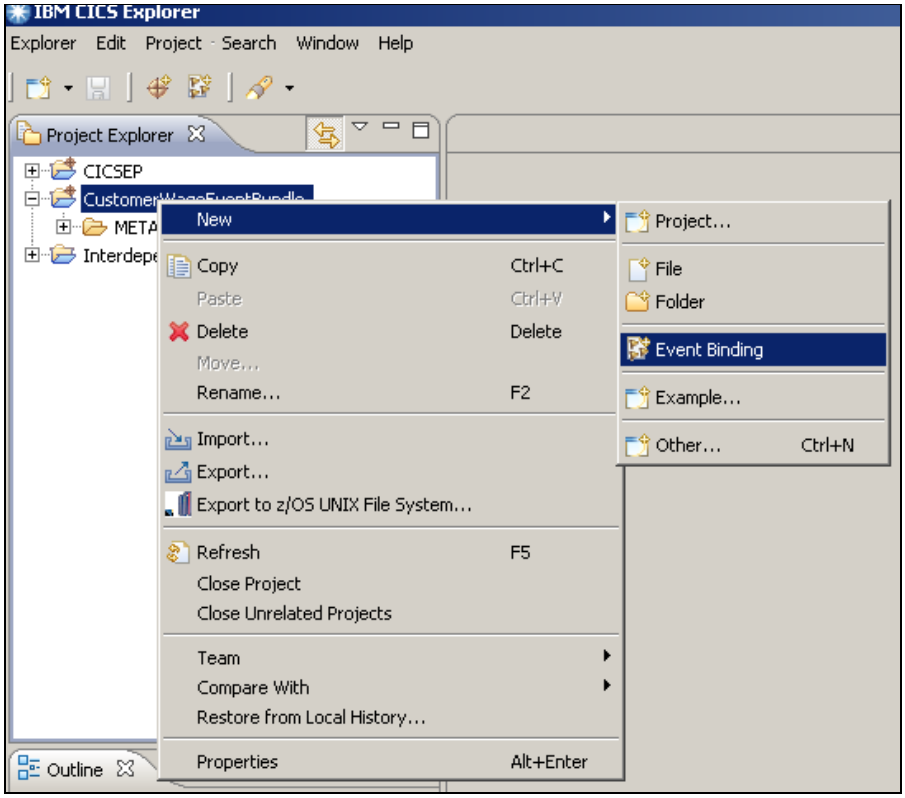
The Bundle Project view displays

2. Specify the project name, ***CustomerWageEventBundle***, , and click **Finish**.



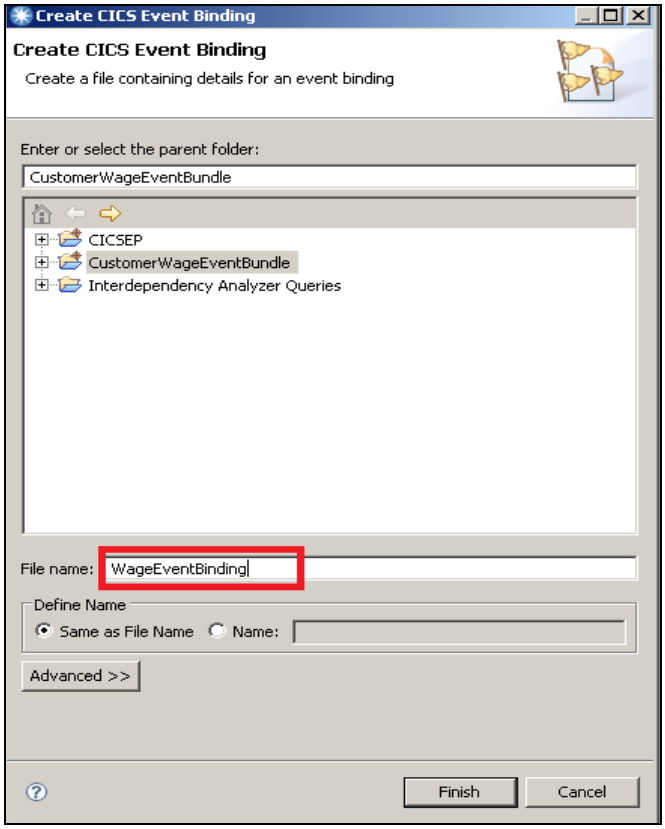
3. Create the event binding within this bundle. In the Project Explorer view

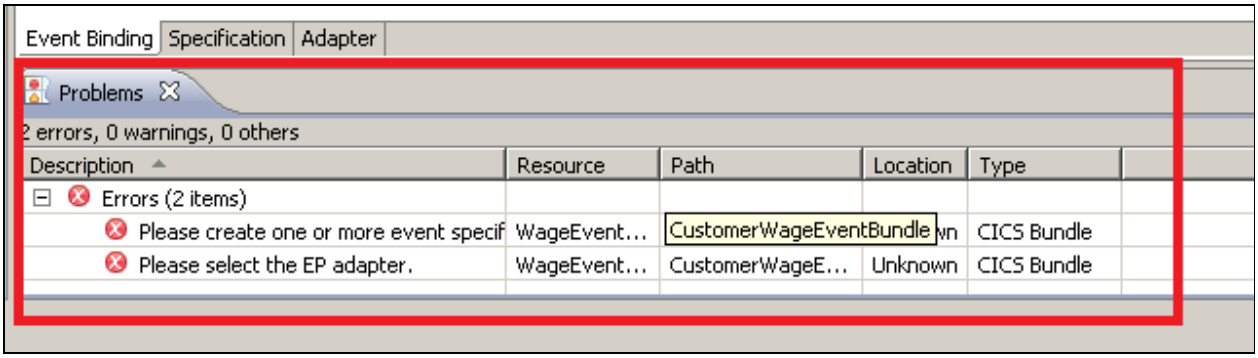
right-click the just created project, ***CustomerWageEventBundle***. Click **New Event Binding** as shown in Figure below.



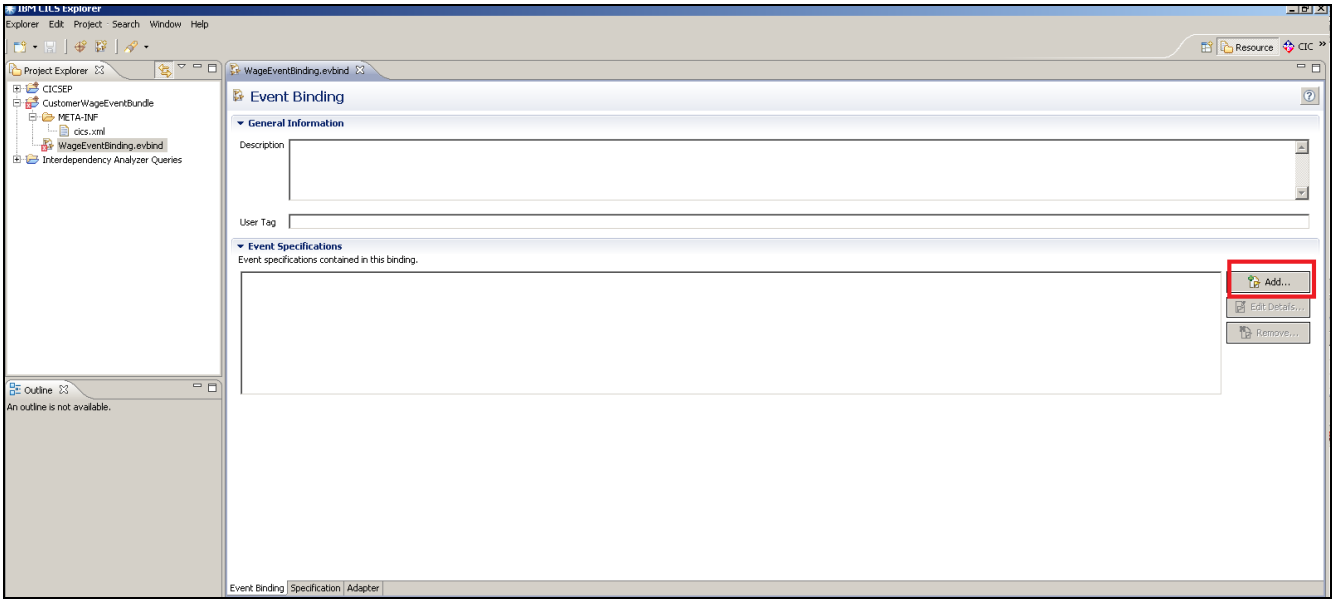
4. Enter **WageEventBinding** in the file name field and click **Finish**.

Note: Some errors appear in the Problems pane, these are not concerns but are caused by information not yet entered

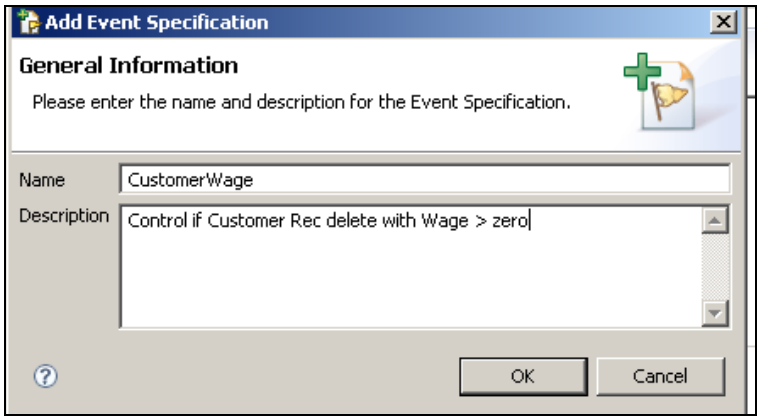




5. Add an event specification to the event binding. In the view presented, enter a description for the event binding, and click **Add**.



6. Enter the name of the Event Specification, ***CustomerWage*** , a description, and click **OK** .



We want the event to include these fields so we add these as items of emitted business information.

Name
Program_Name
Record_Name
Customer_Name
Deleting_Userid
Wage

7. Click the **Specification** tab, and click **Add** next to the Emitted Business Information table. In the Emitted Business Information view, enter the following values:

– Name: Program_Name

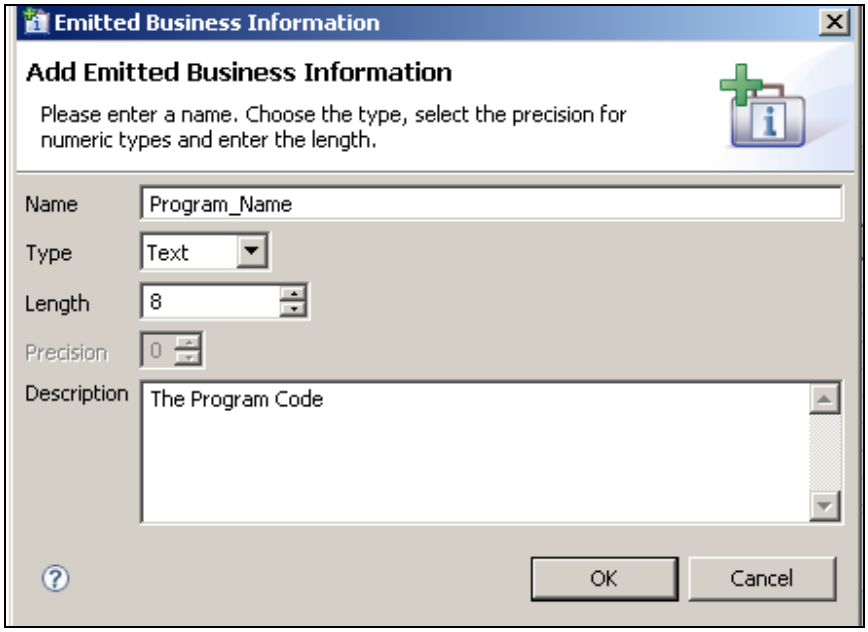
– Type: Text

– Length: 8

– Precision: 0

– Description: The Program Code

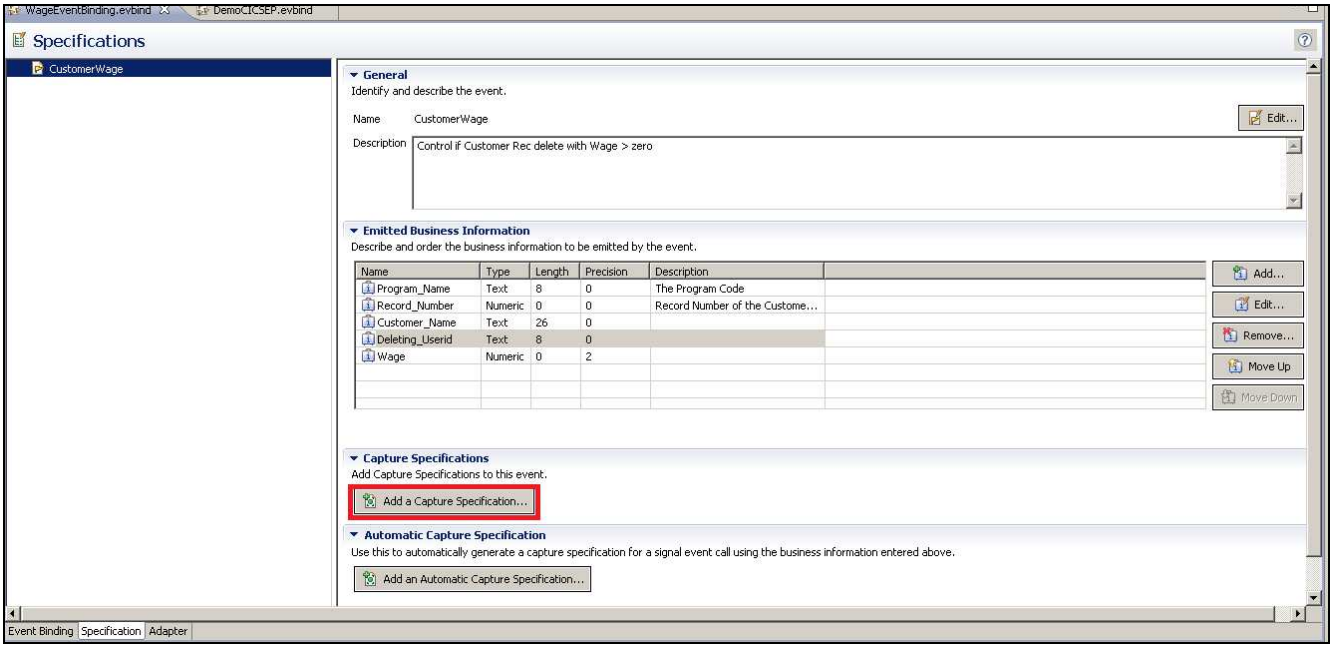
Click OK.



8. Repeat the previous steps for the remaining data items :

Name	Type	Length	Precision
Record_Name	Numeric	0	0
Customer_Name	Text	26	0
Deleting_Userid	Text	8	0
Wage	Numeric	0	2

9. Add a Capture Specification to indicate how CICS can capture this event from the application. In the Specifications view click **Add a Capture Specification**.

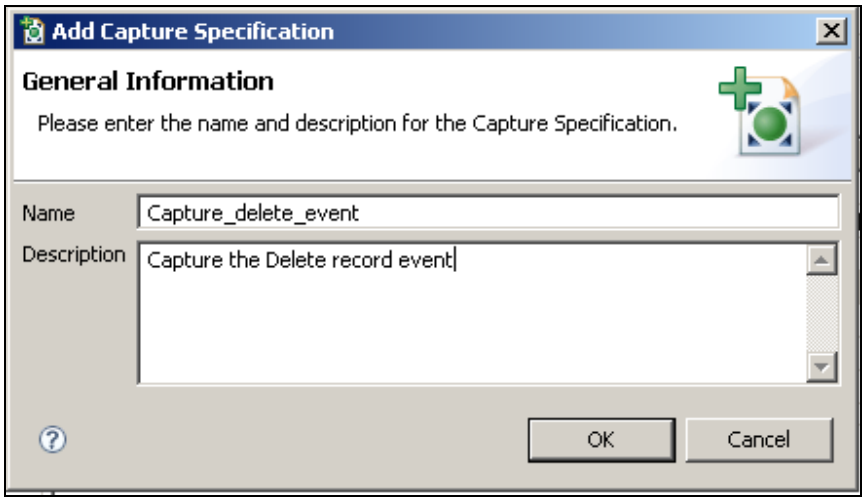


10.In the Add Capture Specification view, enter the information:

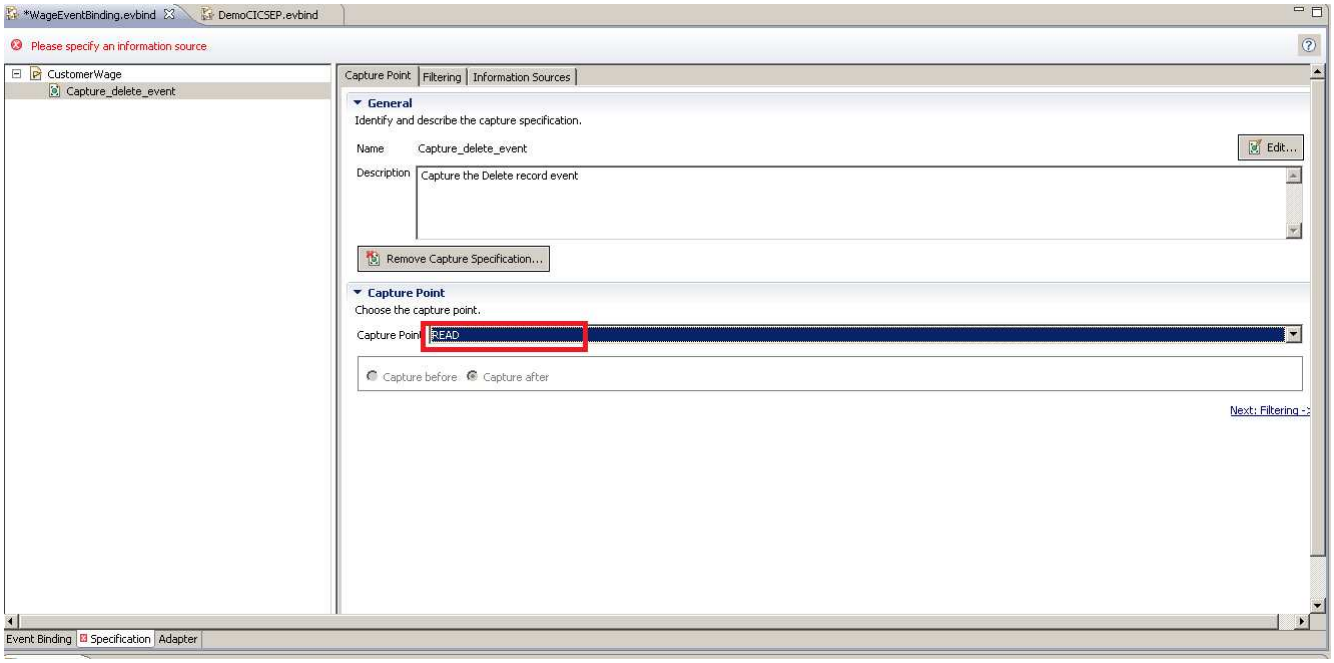
– Name: CaptureQueryEvent

– Description: Capture the Query Event

Click **OK**.



11. From the **Capture Point tab** (across the top), in the **Capture Point section**, use the **Capture Point pull-down** menu to select the **READ** command. Note that you may need to scroll to the right to see the Capture Point pull-down tab.
Click **Next:Filtering**,

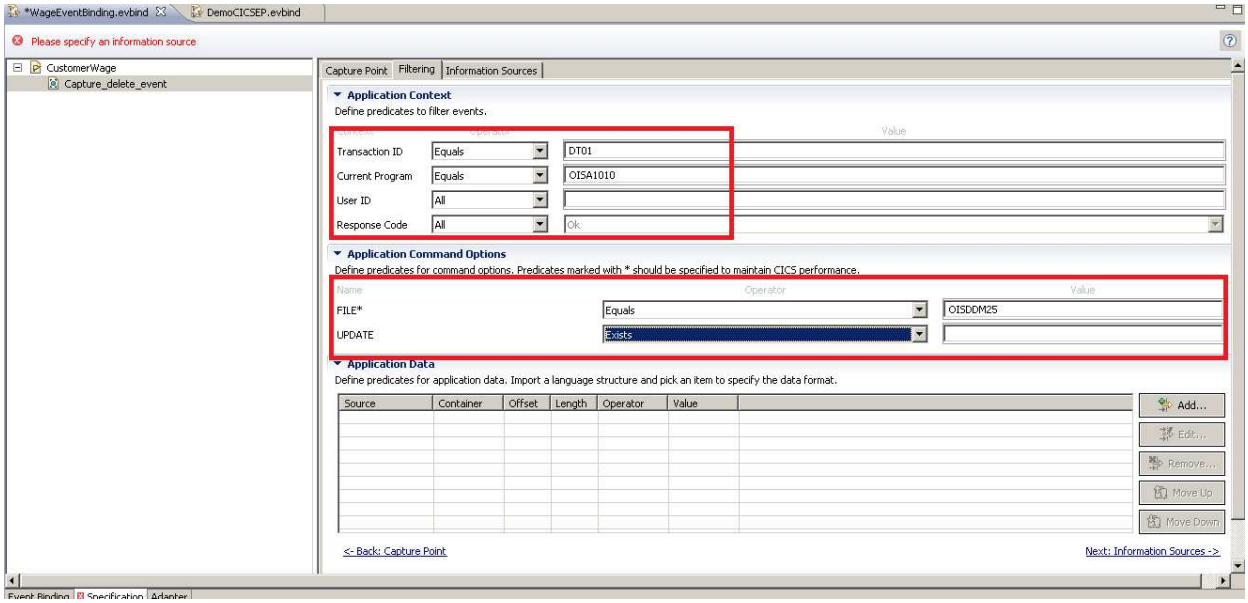


The Application Context lists four general predicates that can be used to filter any event. We can ensure that events only trigger when the transaction is DT01 and the running program is **OISA1010** and the REWRITE statement has succeeded (Response Code is Ok).

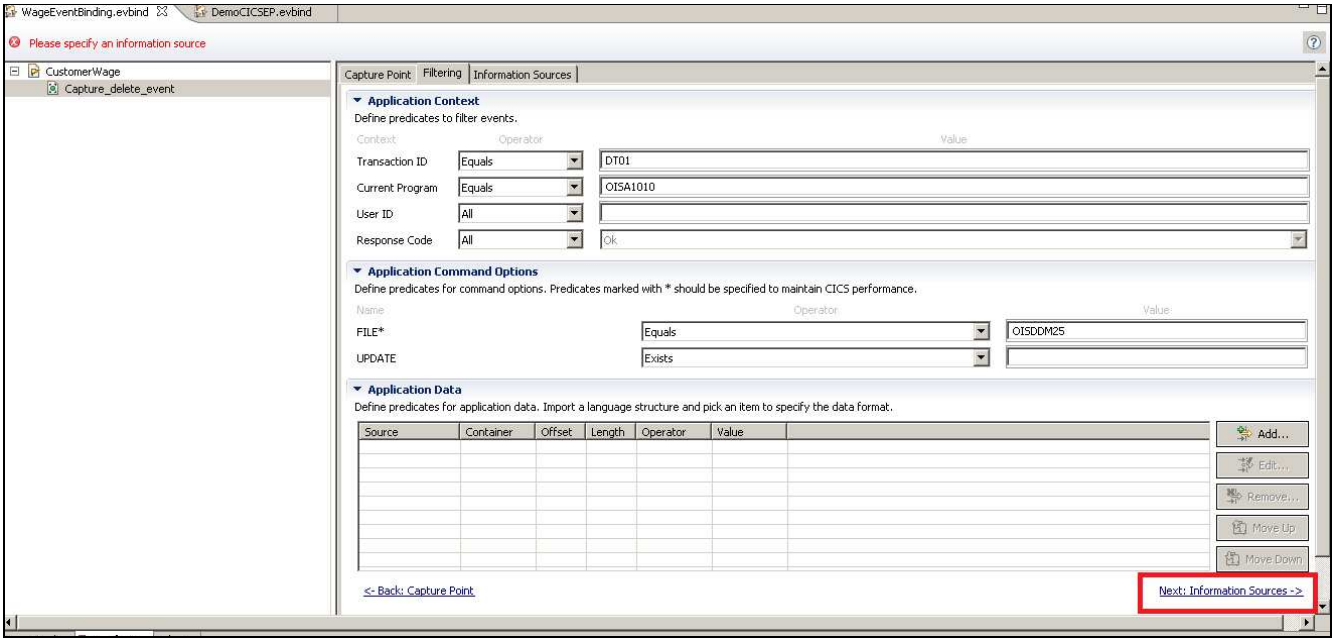
12. From the **CustomerWage** editor, the **Filtering tab** (across the top), **indicate** the values below. Note that a value of All on User ID indicates that we are not filtering on User ID

Context	Operator	Value
Transaction ID	Equals	DT01
Current Program	Equals	**OISA1010**
User ID	All	
Response Code	Equals	Ok

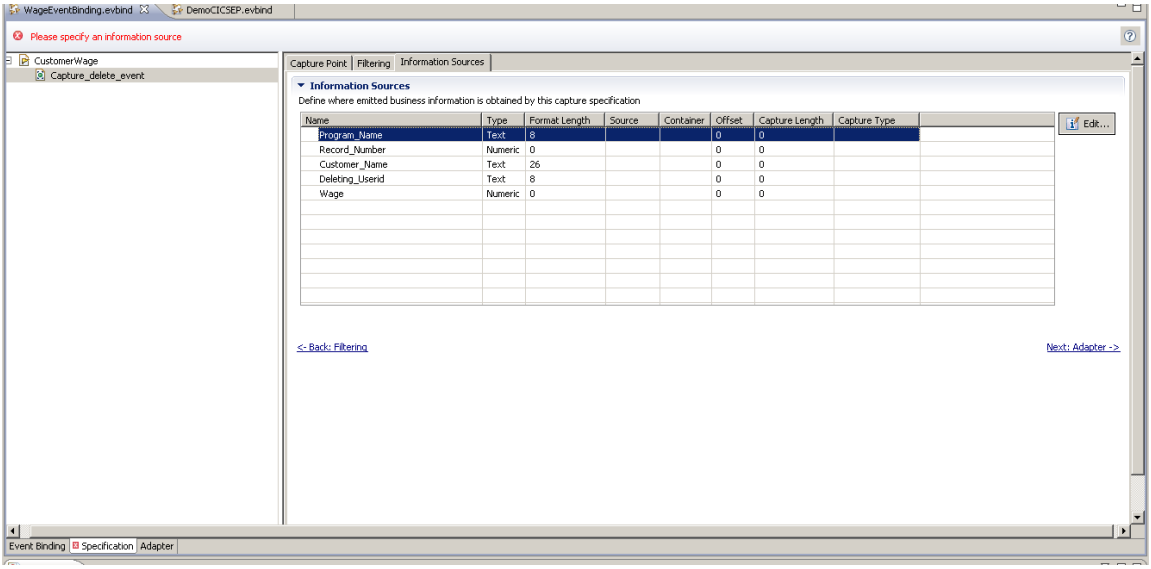
From the **CustomerWageBinding** editor, the **Filtering tab**, in the **Application Command Options section**, for **FILE***, set **Operator** to **Equals**, and **Value** to **OISDDMxx**. (where xx is your team member number).



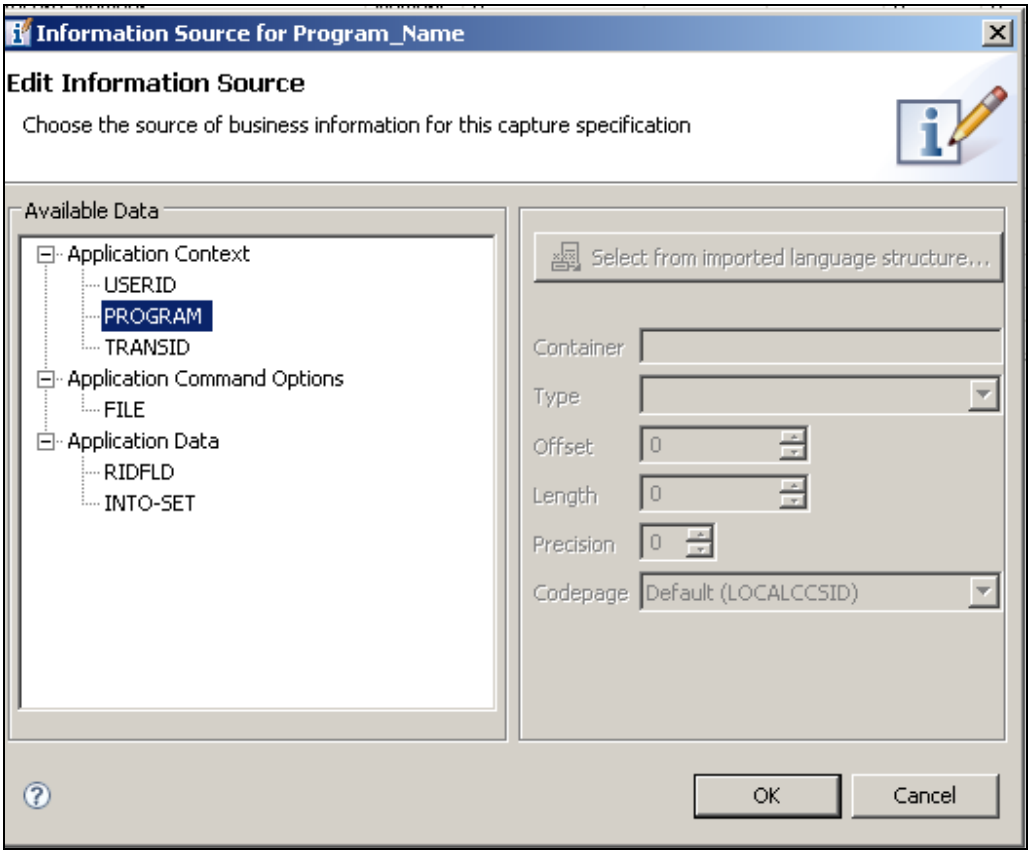
14.Click **Next: Information Sources** on the Filtering panel to indicate how CICS can capture the data requested as emitted business information.



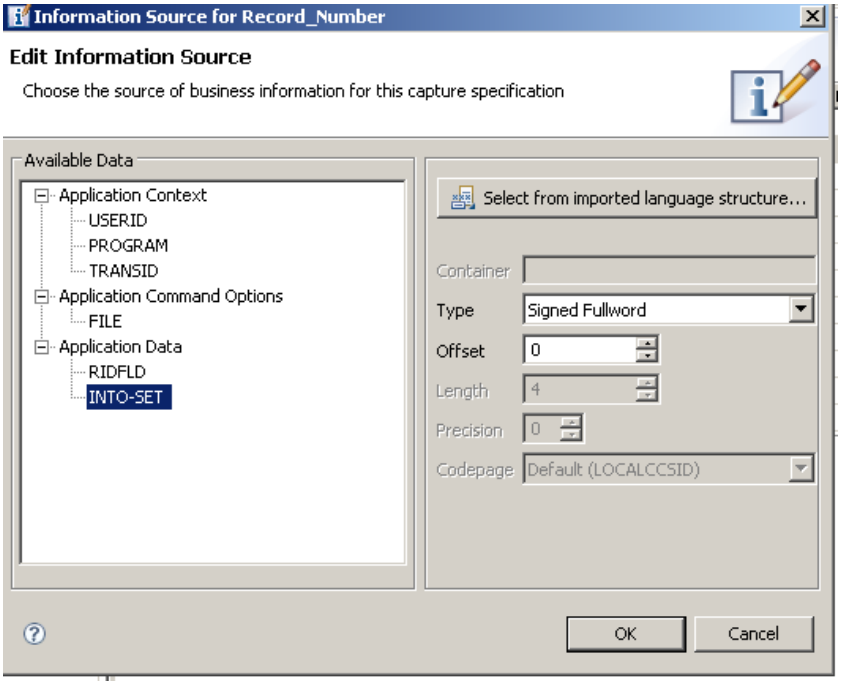
15.Because we already have added Emitted Business Information the “Information Source” panel has already been filled in. To fill in the information source, select the **Program_Name** business information and click **Edit**.



16.On the “Edit Information Source” panel in the Available Data section, click **PROGRAM** on the Application Context

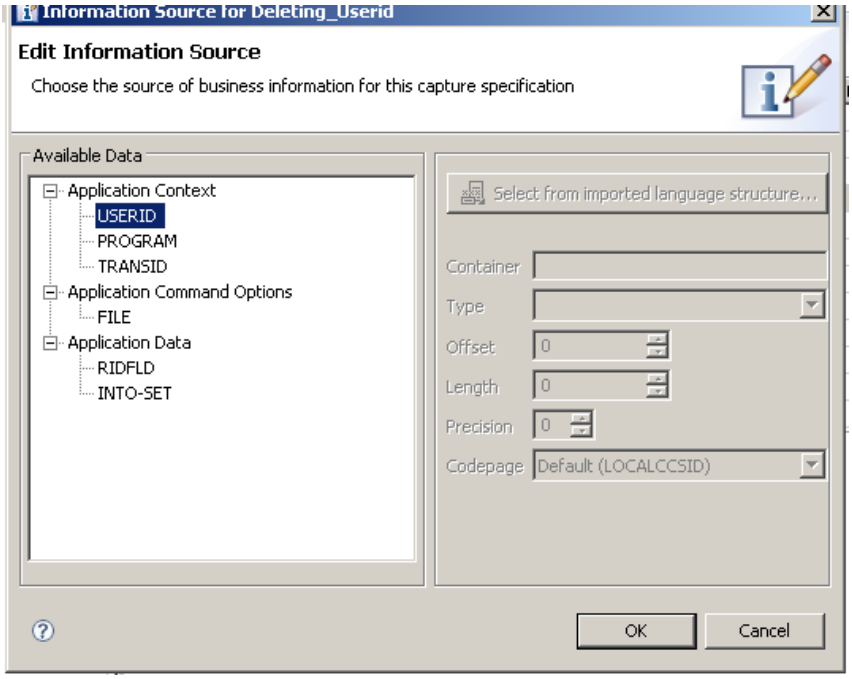


17.Select the **Record_Number** business information and click **Edit**



18.On the “Edit Information Source” panel in the Available Data section, click **INTO-SET** on the Application Data and select Type "***Signed Fullword***"

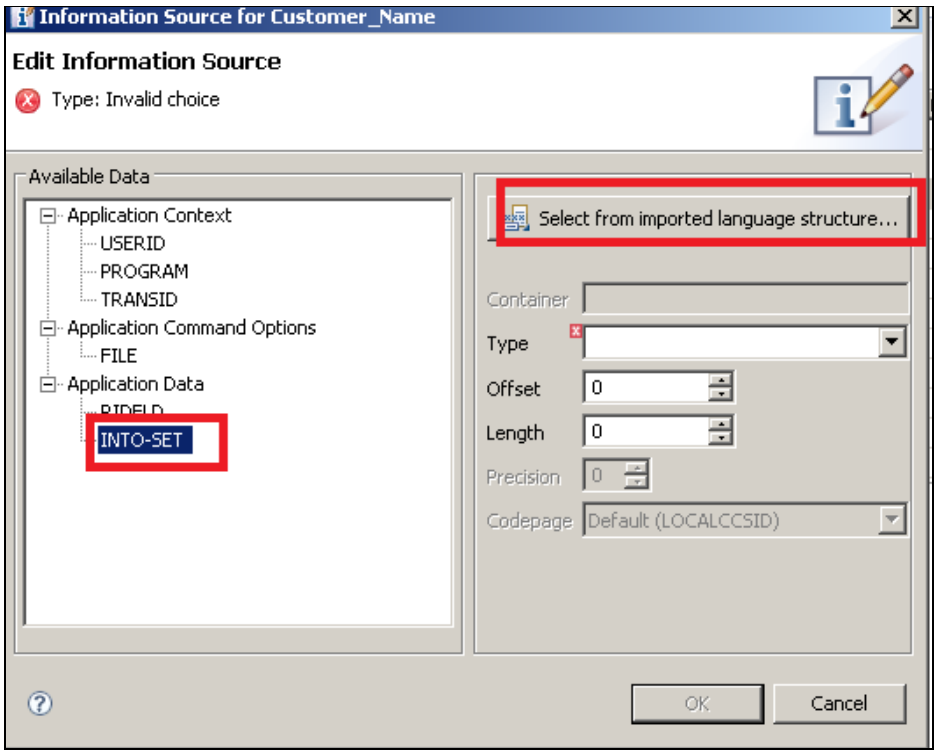
19.Select the **Delete_Userid** business information and click **Edit**



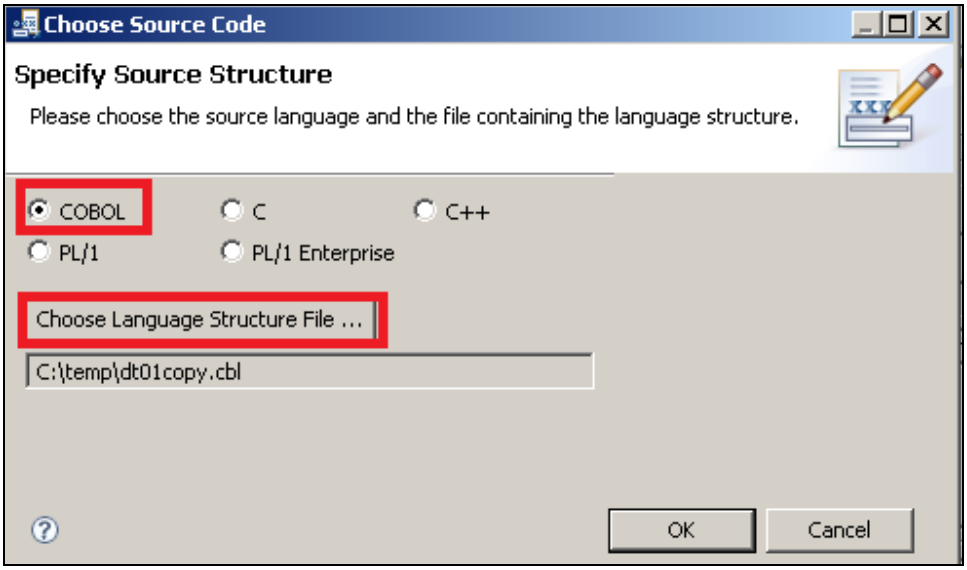
20.On the “Edit Information Source” panel in the Available Data section, click **USERID** on the Application Context

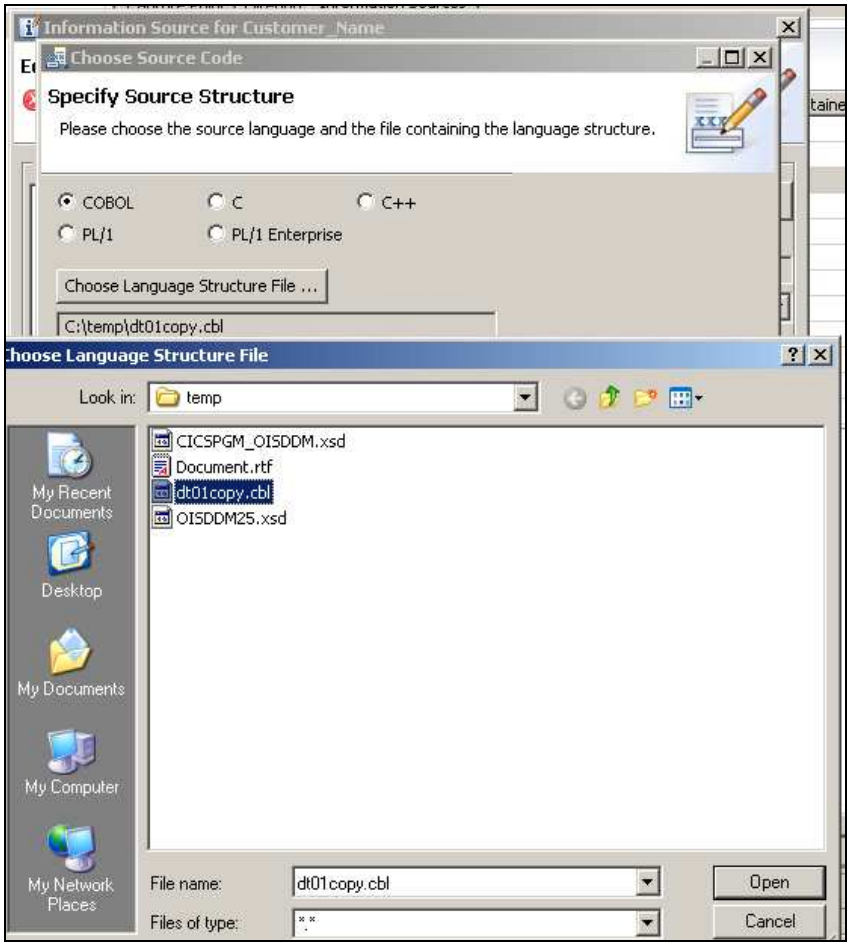
21.Select the **Customer_Name** business information and click **Edit**

Click on "INTO-SET" on Application Data section ==> "Select from imported language structure ..."

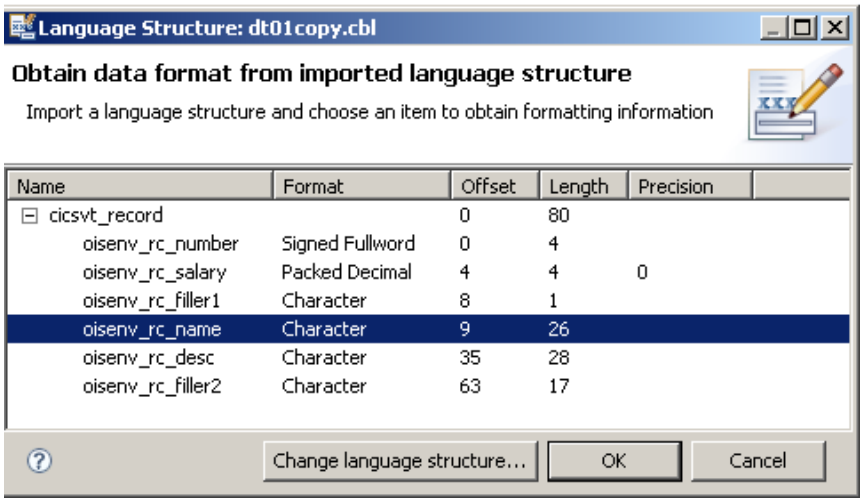


On "Choose Source Code" windows, select COBOL and click "Choose Language Structure File..." to select the Copy book of CICS COBOL program



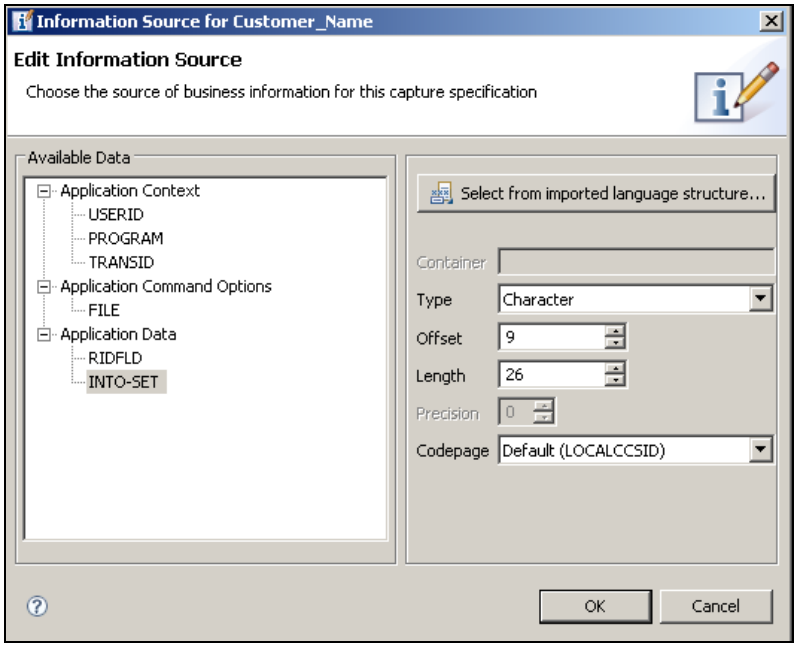


Select the Copybook file and click "**Open**"



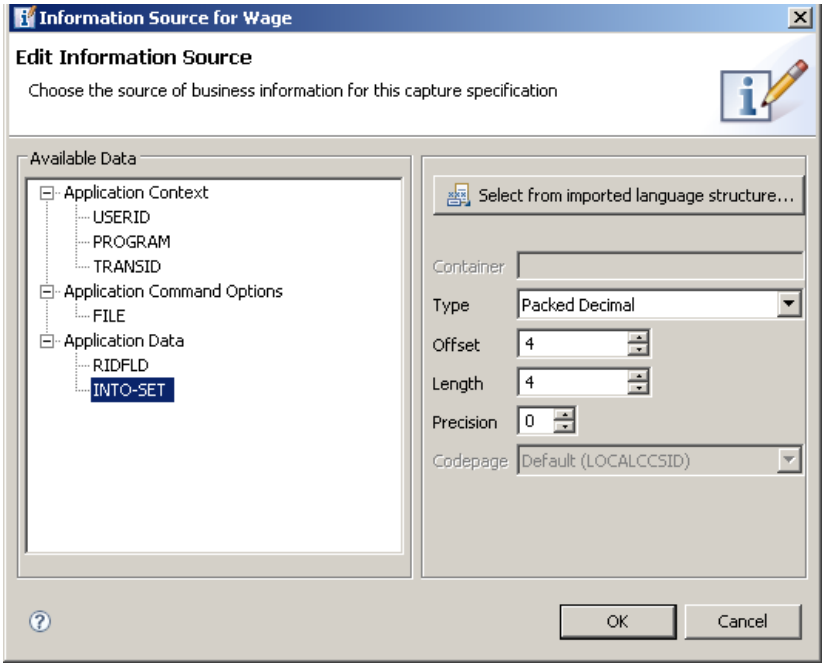
The structure of the copybook open.

select "*oisenv_rc_name*" field to map "**Customer_Name**" as shown in figure below.

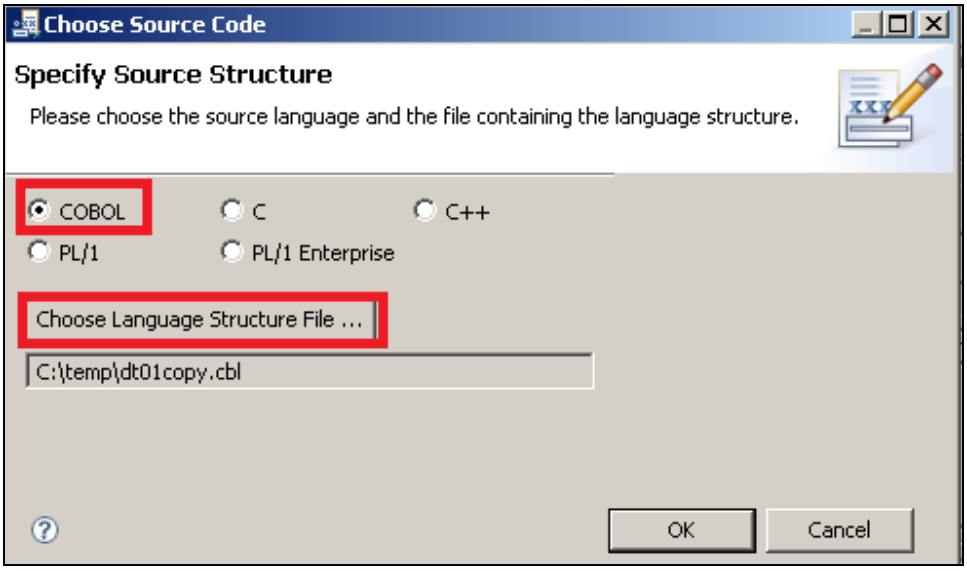


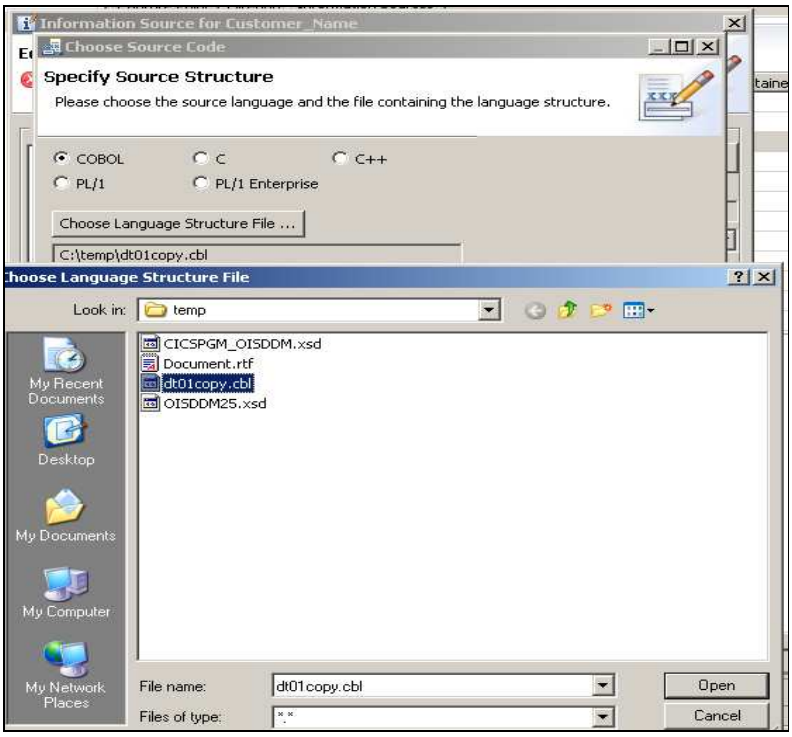
22.Select the **Wage** business information and click **Edit**

Click on "INTO-SET" on Application Data section ==> "Select from imported language structure ..."

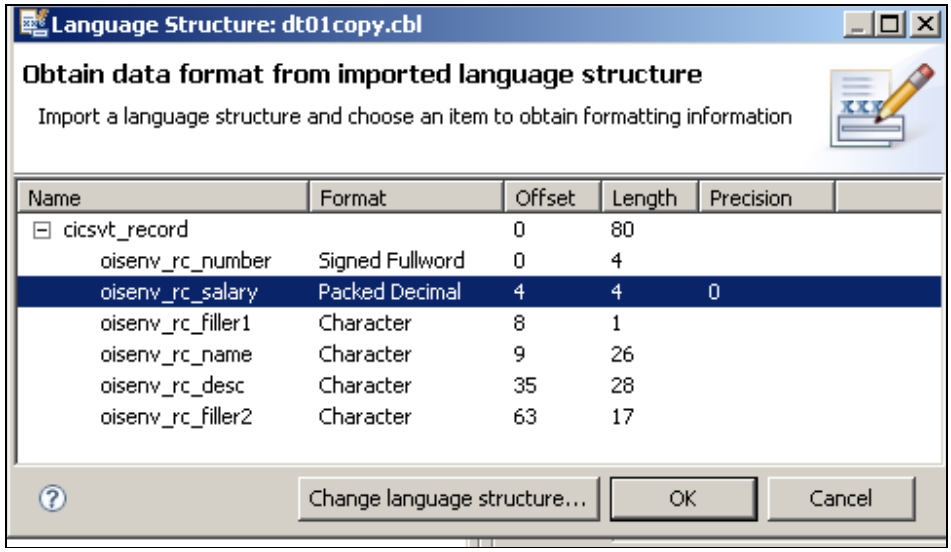


On "Choose Source Code" windows, select COBOL and click "Choose Language Structure File..." to select the Copy book of CICS COBOL program



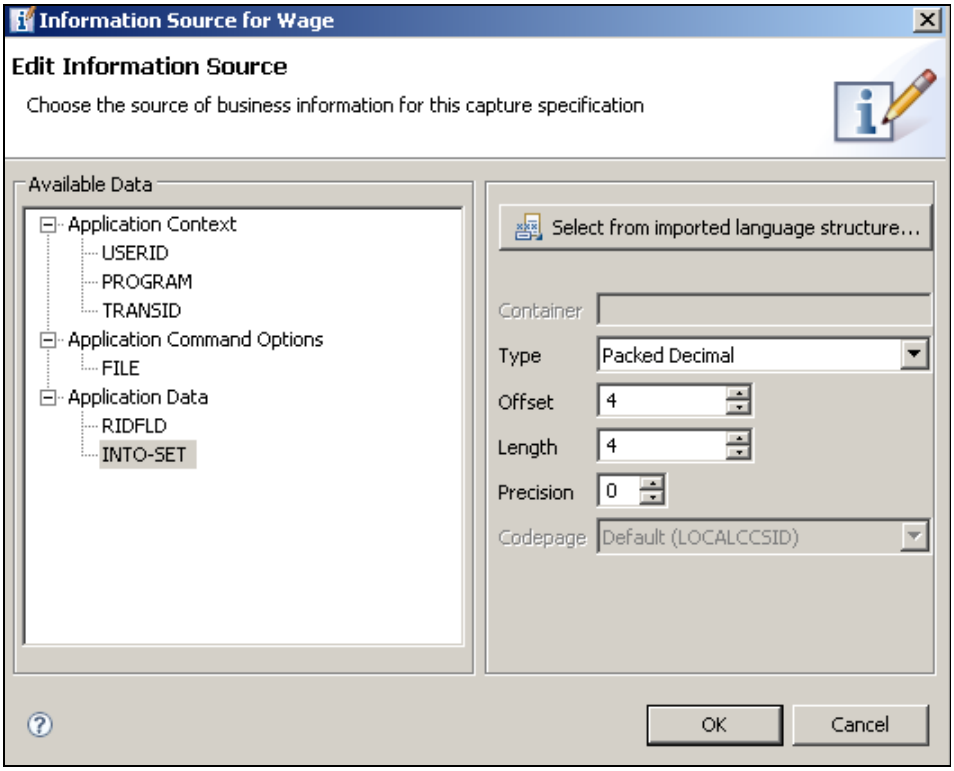


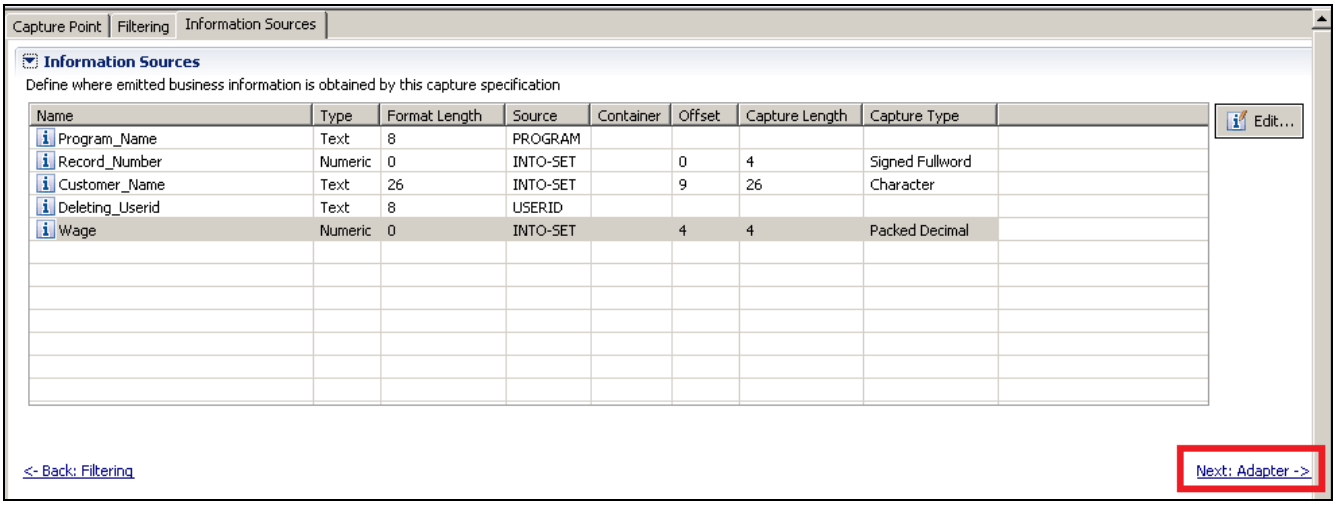
Select the Copybook file and click "**Open**"



The structure of the copybook open.

select "*oisenv_rc_salary*" field to map "**Wage**" as shown in figure below.





23.Click **Next: Adapter**.

1.3.3Specifying Adapters

1.3.3.1 Defining the Temporary Storage queue adapter

For test purpose we start by using the Temporary Storage Queue EP adapter. It allow us to easily verify the emitted business information, by either using the CEBR or the CECI CICS supplied transaction.

On the adapter panel, define the following information:

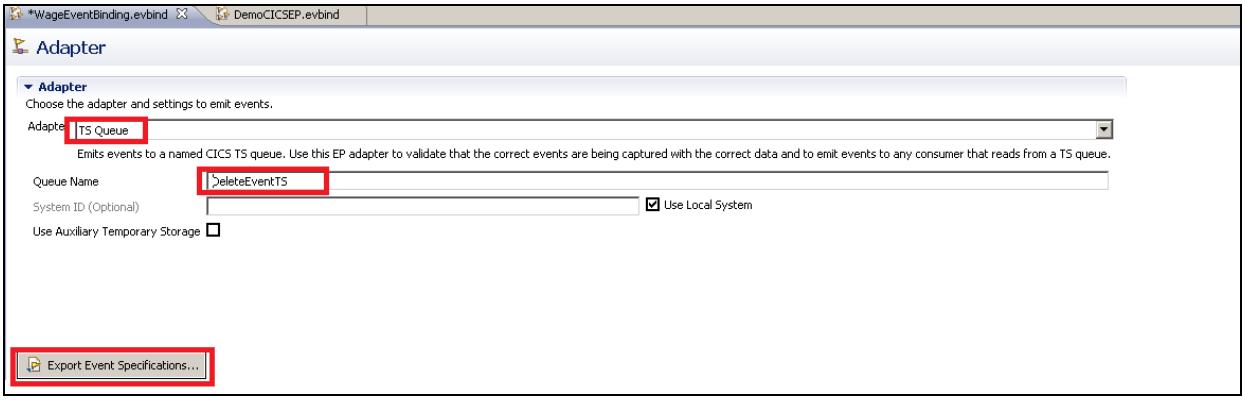
_ Adapter TS : Queue

_ QueueName : *DeleteEventTSq*

_ System ID : We leave blank (use local TS queue)

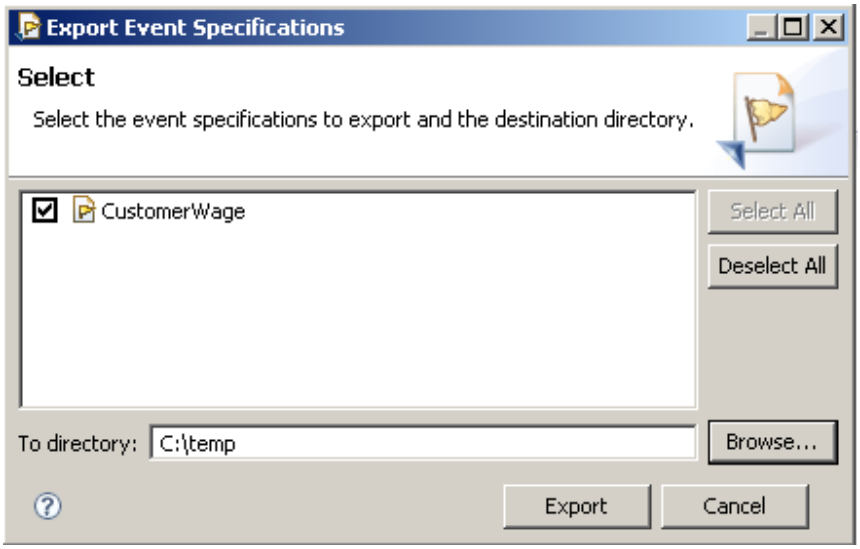
_ Use Auxiliary Temporary Storage : We leave unchecked (use main storage TS queue)

Press **Ctrl-S** to save the configuration.



1.3.3.2 Exporting the event specification

Click **Export Event Specification**, in the Export Event Specification view. Select the box next to ***CustomerWage***. Enter ***C:\temp*** in the To directory box and click **Export**



The exported Event Specification is shown in Example below. The exported copybook can be used for inclusion in a COBOL program written to read the temporary storage queue.

Example 5-2 Exported Event Specification

```
      * Generated copybook for Event Specification
      * 'CustomerWage'
       01 CustomerWage.
           05 ContextData.
              COPY DFHEPFEO.
           05 EventData.
              10 Program_Name                  PIC X(8).
              10 Record_Number                 PIC +9(10).
              10 Customer_Name                 PIC X(26).
              10 Deleting_Userid               PIC X(8).
              10 Wage                          PIC +9(7).99.
```

To map the context data in the context container, the following copybooks are shipped with CICS:

_ DFHEPFE0: COBOL

_ DFHEPFED: Assembler

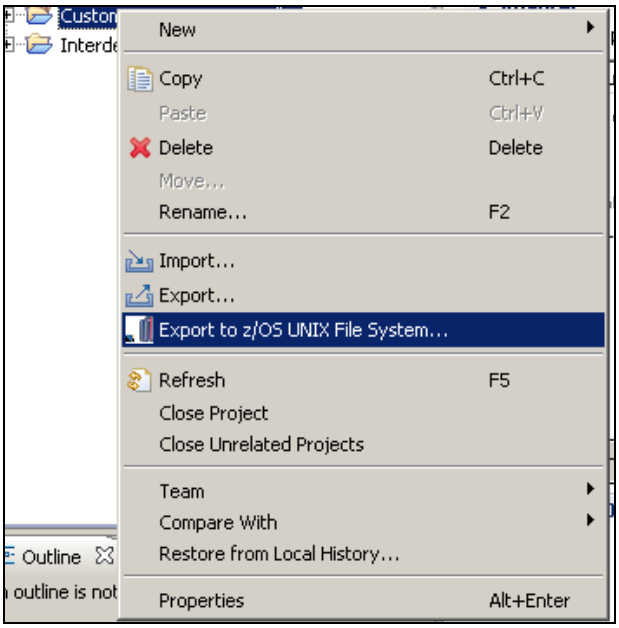
_ DFHEPFEL: PL/I

_ DFHEPFEH: C

1.3.3.3 Exporting the bundle project

To use the event specifications in CICS TS we first export the bundle project. We export the bundle direct to a System z HFS, where CICS can read it directly.

In the project view of IBM CICS Explorer, right-click the ***CustomerWageEventBundle*** project and click **Export to System z HFS**



From the **Export to System z HFS** dialog, **specify** the following **values** and click the **Finish** button.
If you are prompted to create the bundles directory, reply yes.
If you re-export your event binding, you will be asked if you want to delete the ***CustomerWageEventBundle*** directory. Reply yes to this prompt.

Name	Value
Host	zt01.pssc.mop.fr.ibm.com
FTP Port	21
Username	SUCI1xx (where xx is your team number)
Password	<your_password>
HFS directory	/etc/cicscfg/sharedir/suci1xx/bundles/ ==> *Where xx is your team number*
Delete current contents of HFS directory	Unchecked

Note: The Username must be a valid TSO user who has write access to the specified HFS directory.

Note: As an alternative to Export to System z HFS, the bundle can be exported to the local file system, and a File Transfer Program can be used to copy the event binding file to System z HFS. The file transfer must be in binary**.**

The export creates two files:

_ /etc/cicscfg/sharedir/suci1xx/bundles/CustomerWageEventBundle/META-INF/cics.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<manifest xmlns="http://www.ibm.com/xmlns/prod/cics/bundle" bundleVersion="1" bundleRelease="0" build="Not Found">
   <meta_directives>
      <timestamp>2011-04-05T15:51:08.881+01:00</timestamp>
   </meta_directives>
   <define name="WageEventBinding" type="http://www.ibm.com/xmlns/prod/cics/bundle/EVENTBINDING"
path="WageEventBinding.evbind"/>
</manifest>
```

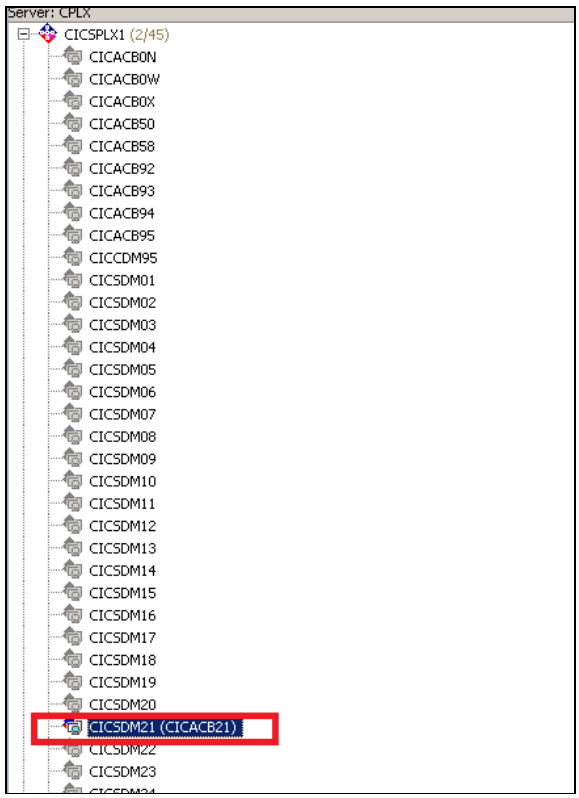
Note: A directory name of ***CustomerWageEventBundle*** is added by the Export function. The name comes from the CICS Explorer project name and must be used in the CICS resource definition.

_ Installing a bundle definition

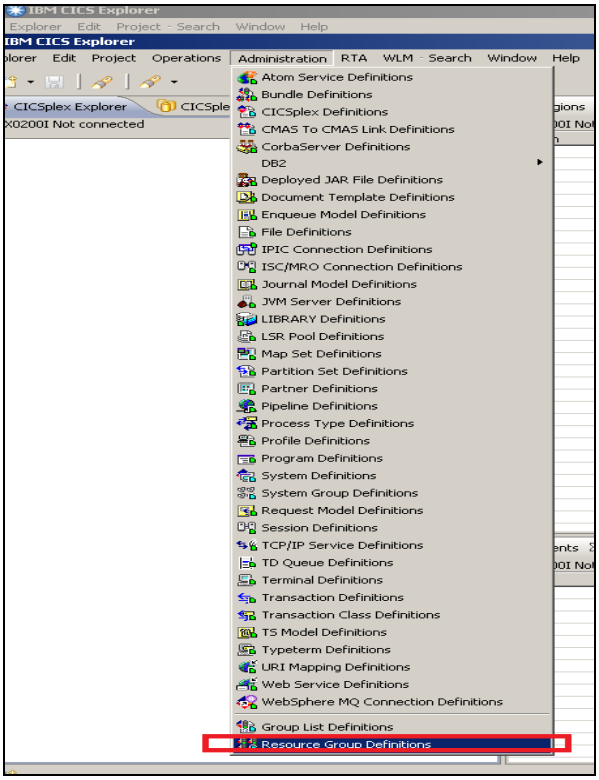
1.3.4 Installing a Bundle

To use the Event Binding definition in CICS, we use CICS Explorer to create the resources. In CICS Explorer we open the CICS SM perspective

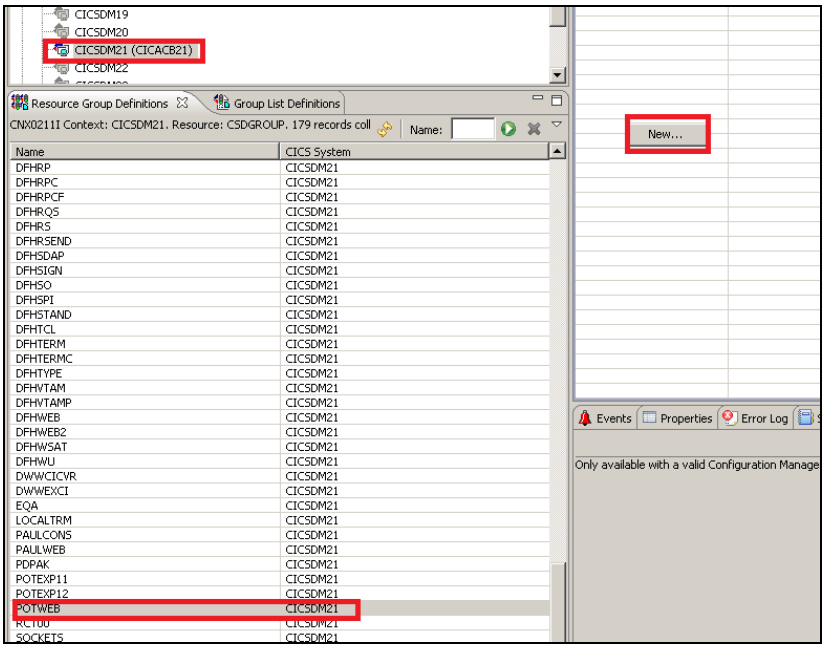
Expand the CICSPLX1 group in the CICSPLex Explorer Tab and select your CICS region.



a. Click **Administration Resource Group Definitions** and right-click the white space.



c. Click **Administration Bundle Definitions**. In the presented view, Select the existing group "**POTWEB**"



right-click the white space, click **New**, and enter the information:

• Data Repository: CICSPLX1

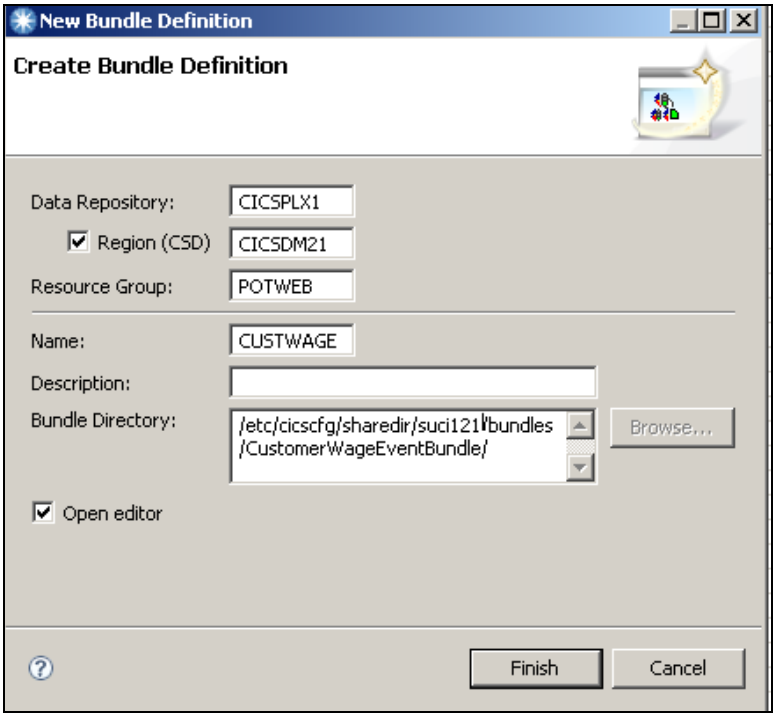
• Resource Group: SUCI110G

• Name: **CUSTWAGE**

• Description: The Shopping Bundle definition

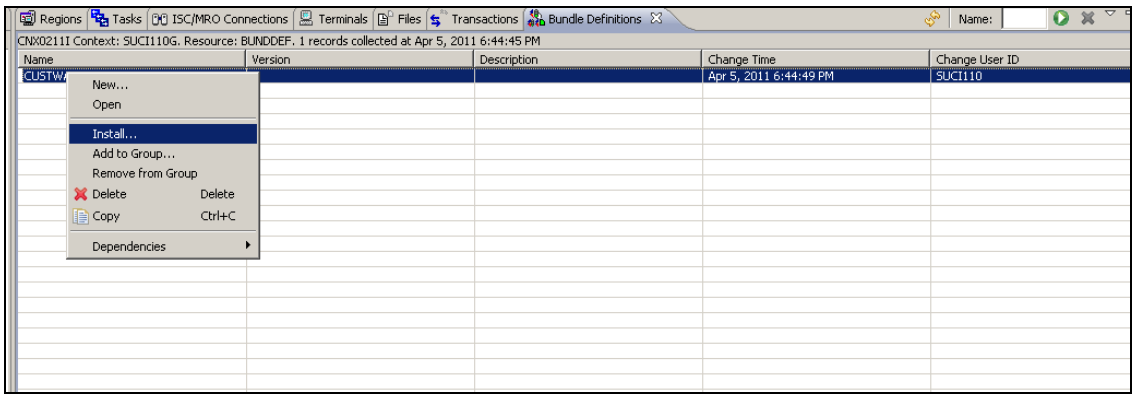
• Bundle Directory: ***/etc/cicscfg/sharedir/suci1xx/bundles/CustomerWageEventBundle/***

Click **Finish**.

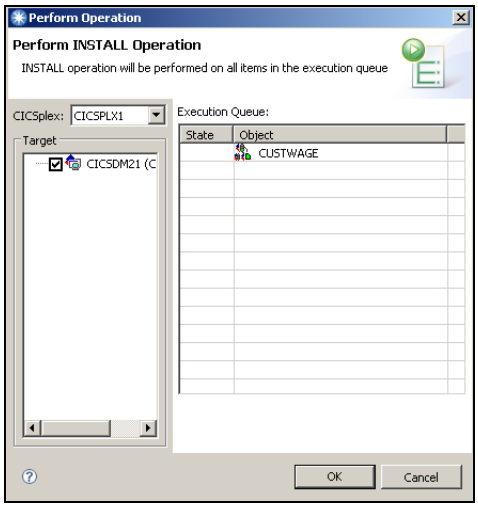


2. Install the **CUSTWAGE** bundle resource.

In the Bundle Definitions view right-click the **CUSTWAGE** bundle and click **Install.**

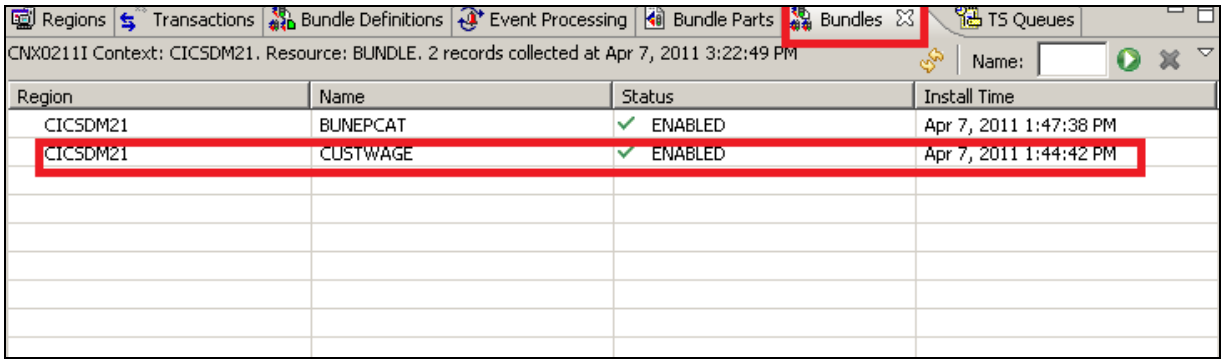


In the window check the target region for the install and click **OK**.

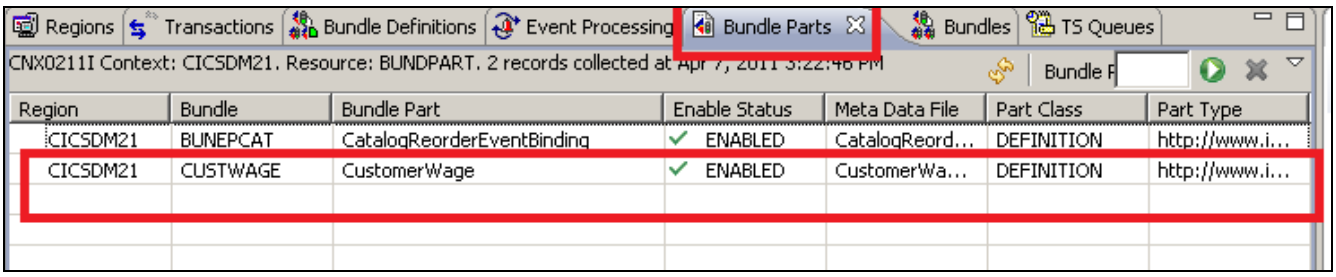


4. Verify the install. In CICS Explorer, click **Operations Bundles.** Shows the installed CUSTWAGE bundle resource.

- Bundles View



- Bundle Parts

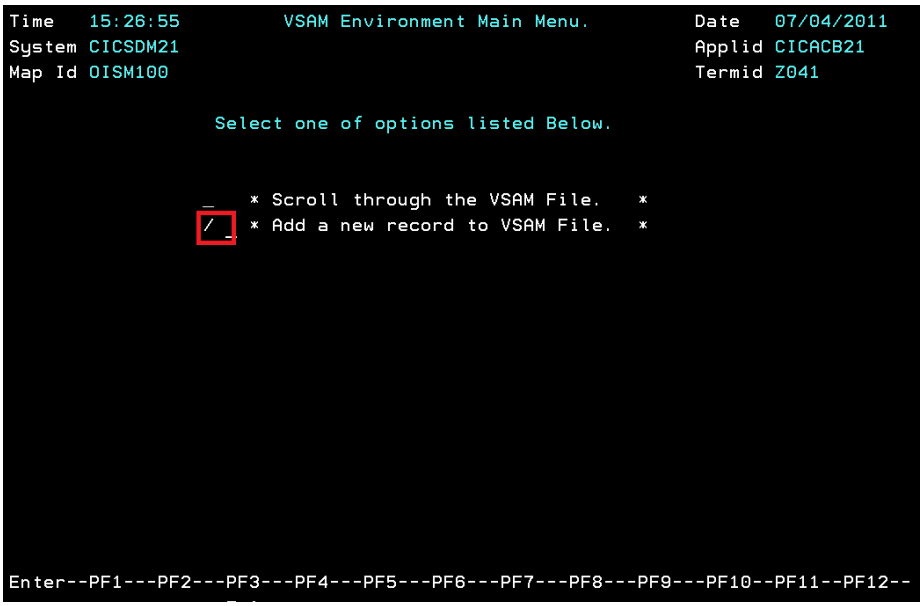


1.3.5 Testing an event specification

1.3.5.1 Testing the CICS WAGE application

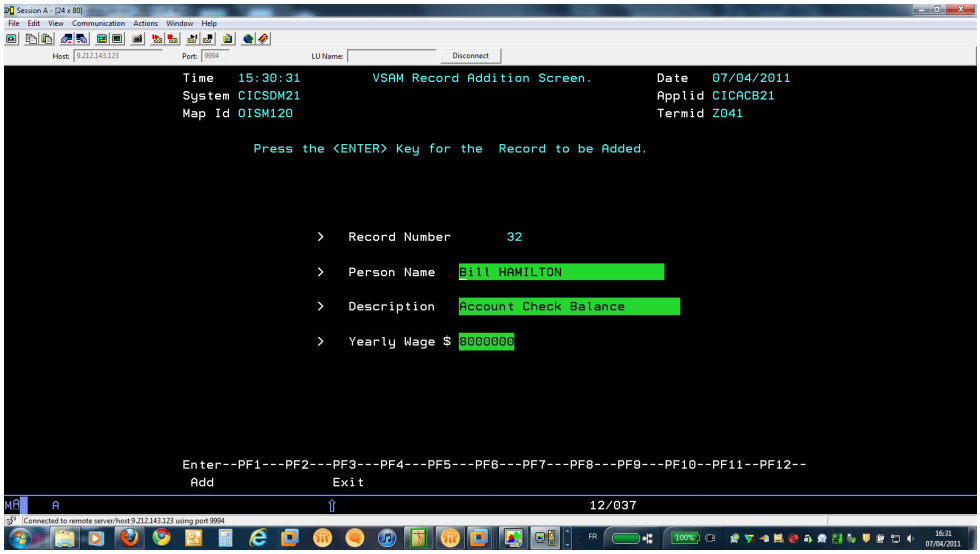
Now we test to see whether a delete record in our CICS application causes a delete event to be emitted.

1. On a CICS terminal enter the transaction ID DT01 and press **Enter**.

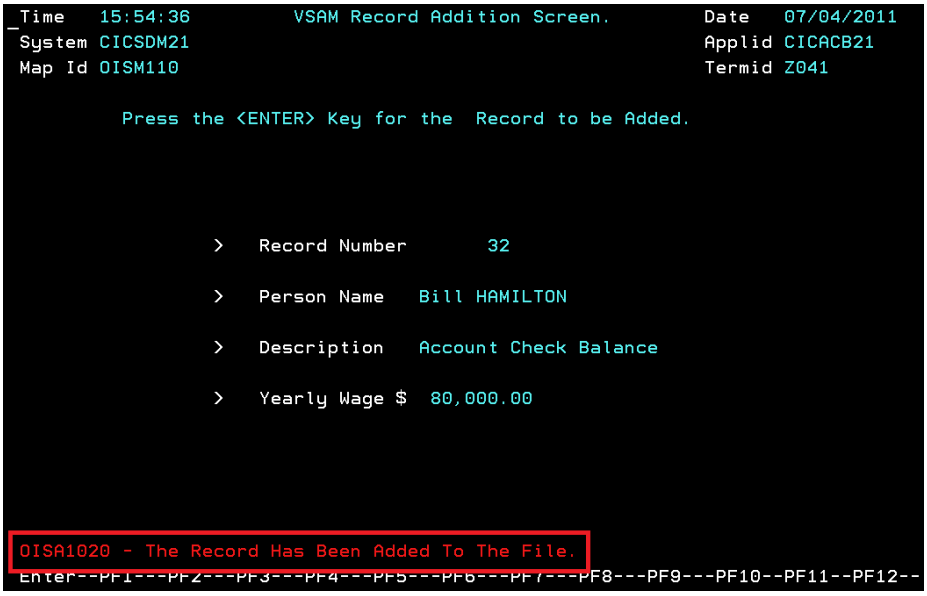


2 - Select *"*Add a new record to VSAM File. * "* using /

3 - Fill the fields to perform new record in the VSAM file

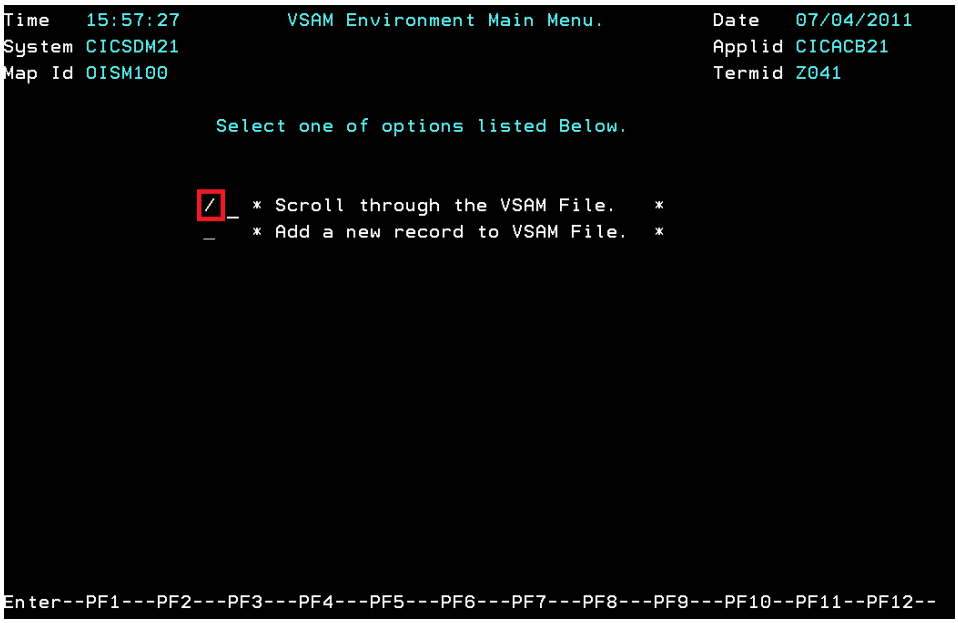


end hti "**Enter**"



You've got a message : "*OISA1020 - The record Has Been Added To The File*."

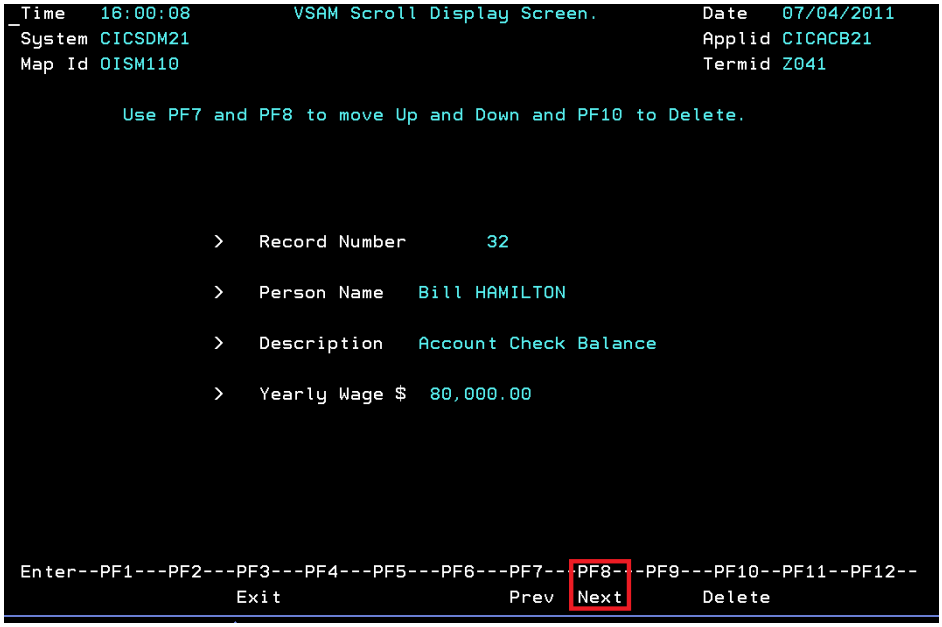
Hit "**PF3**" Key to return to the main menu.



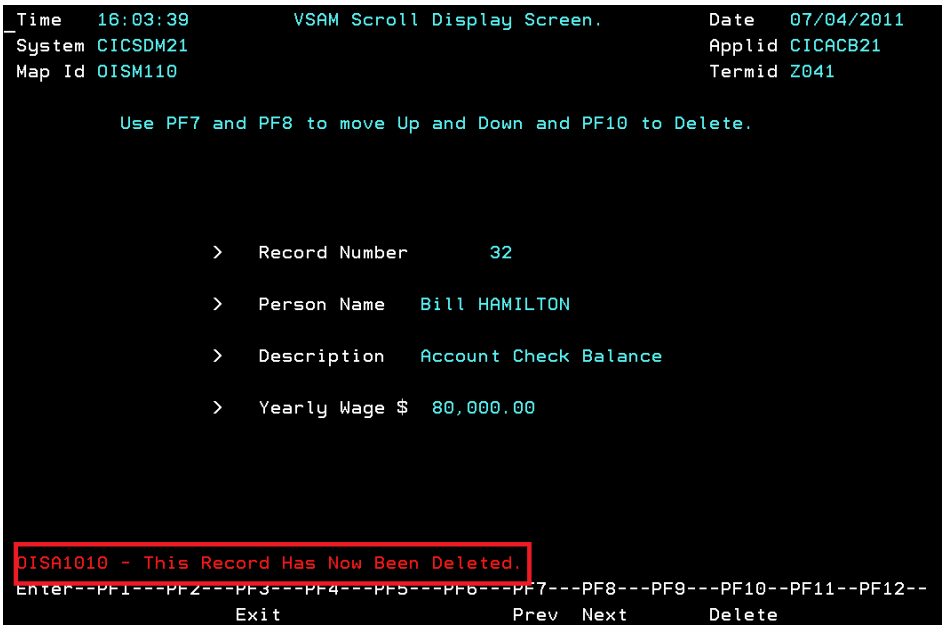
Now select "* Scroll through the VSAM File. *" and hit "**Enter**" Key.

You will obtain the first record in the VSAM file displayed.

Hit "PF8" key to browse the next records until obtained your created record.



When you obtained your record, then hit "PF10" key to delete it.

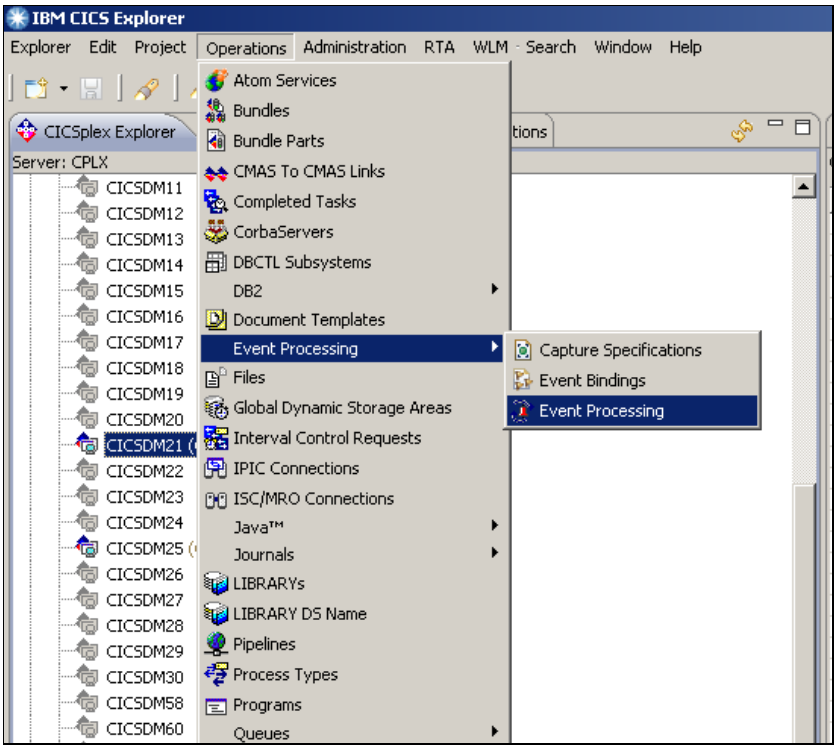


You've got a message : "*OISA1010 - The record Has Been Deleted*."

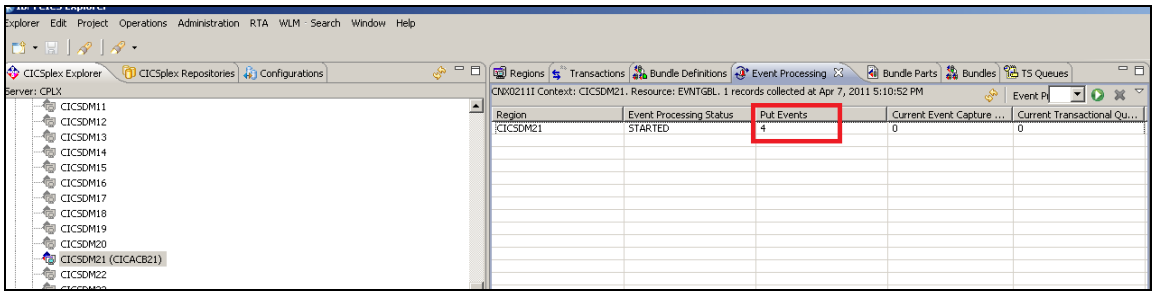
1.3.5.2 Verifying the event processing

We use CICSplex Explorer to verify event processing.

1. Select CICS region ***CICSDMxx*** within the CICSplex. Click **Operations Event Processing Event Processing**

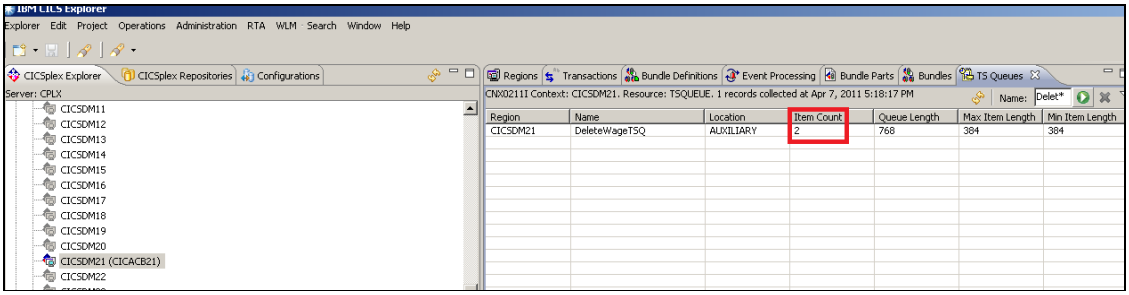


Column Put Events shows number of events which has been emitted since the bundle installation



2. Verify that a record has been written to the specified Temporary Storage queue ***DeleteWageTSQ***.

Use CICSplex Explorer to monitor the queue. The Item Count has a value of a number of delete record test.



3. Log on to CICS and use the **CEBR**

Note: You need to turn upper case off to issue CEBR. ==> ***CEOT NOUCTRAN*** can be used to do this.

The CICS supplied transaction CEBR can be used to display the queue. The command looks like this:

CEBR *DeleteWageTSQ*

Example below shows the data put to Temporary Storage queue, QueryEventTSq,

```
CEBR  TSQ DeleteWageTSQ    SYSID DM21 REC     1 OF     2    COL     1 OF   286
ENTER COMMAND ===> _
     ***************************  TOP OF QUEUE  ********************************
0001 EPFE0001CustomerWage                          OISDDM25
0002 EPFE0001CustomerWage                          OISDDM25
     ************************  BOTTOM OF QUEUE  *****************************
PF1 : HELP               PF2 : SWITCH HEX/CHAR     PF3 : TERMINATE BROWSE
PF4 : VIEW TOP           PF5 : VIEW BOTTOM         PF6 : REPEAT LAST FIND
PF7 : SCROLL BACK HALF   PF8 : SCROLL FORWARD HALF PF9 : VIEW RIGHT
PF10: SCROLL BACK FULL   PF11: SCROLL FORWARD FULL PF12: UNDEFINED
```

```
CEBR  TSQ DeleteWageTSQ    SYSID DM21 REC     1 OF     2    COL   214 OF   286
ENTER COMMAND ===> _
     ***************************  TOP OF QUEUE  ********************************
0001            OISA1010+0000000032AAAA                      SUCI121 +89000.00
0002            OISA1010+0000000032Bill HAMILTON             SUCI121 +80000.00
     ************************  BOTTOM OF QUEUE  *****************************
```

1.3.6Creating event to WebSphere Business Events

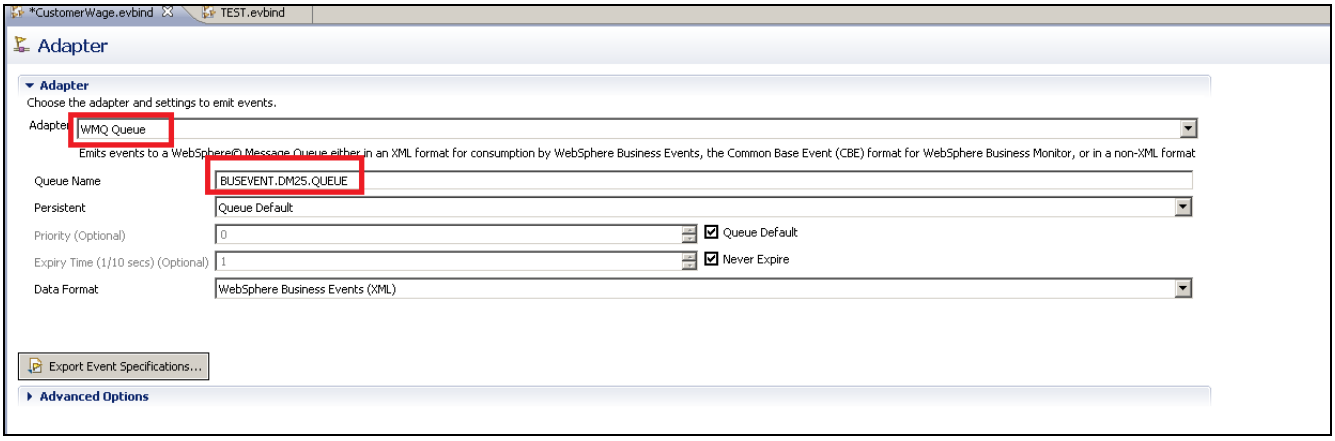
Next, test emitting the same event to WebSphere Business Event.

1. In CICS Explorer open the Resource perspective. In the Project Explorer window, expand the *CustomerWageEventBundle* project. **Right-click** *CustomerWage.evbind* and click **Open**. In the presented window, click the **Adapter** tab at the bottom of the window.

Set the following values:

Name	Value
Adapter	WMQ Queue
Queue Name	BUSEVENT.DMxx.QUEUE (Where xx is your team number)
Data Format	WebSphere Business Events (XML)

Press **Ctrl-S**.



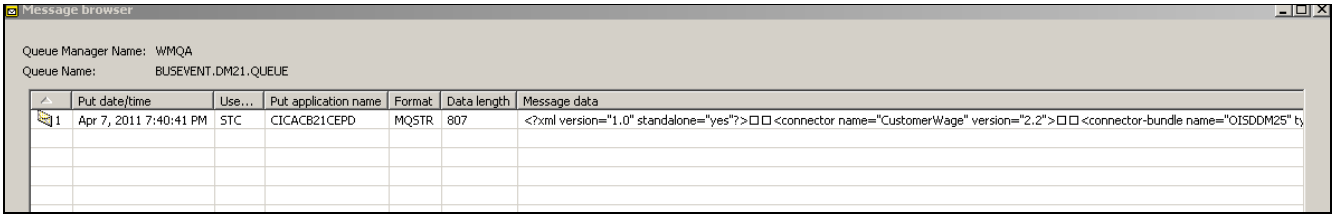
We show the WebSphere Business Events EP adapter specifics from the event binding file.

```
<eventDispatcherSpecification>
      <eventDispatcher>
         <eventDispatcherPolicy>
            <dispatchPriority>normal</dispatchPriority>
            <eventsTransactional>false</eventsTransactional>
            <adapterUserid useContextUserid="false"></adapterUserid>
            <adapterTranId></adapterTranId>
         </eventDispatcherPolicy>
         <eventDispatcherAdapter>
            <wmqAdapter>
               <queueName>BUSEVENT.DM21.QUEUE</queueName>
               <persistent>QUEUE_DEFAULT</persistent>
               <priority>-1</priority>
               <expiryTime>-1</expiryTime>
               <format>WBE</format>
            </wmqAdapter>
         </eventDispatcherAdapter>
      </eventDispatcher>

   </eventDispatcherSpecification>
```

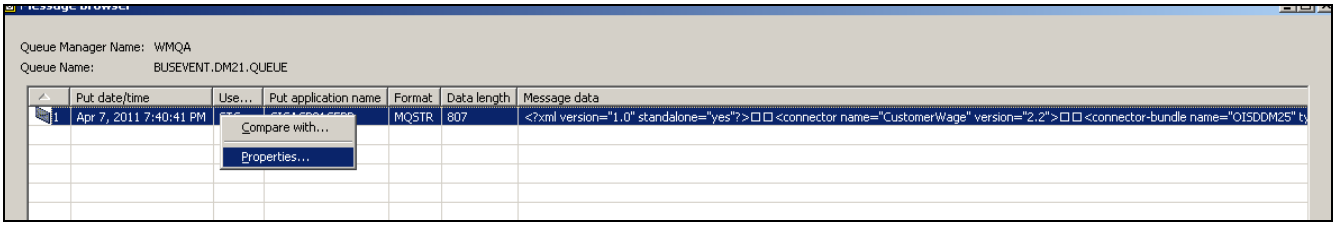
2. Discard the BUNDLE resource in CICS, “Exporting the bundle project”, “Testing the DT01 transaction”.

We use IBM WebShere MQ Explorer to show the message put to the WMQ queue by the adapter. We see one message put to the queue,

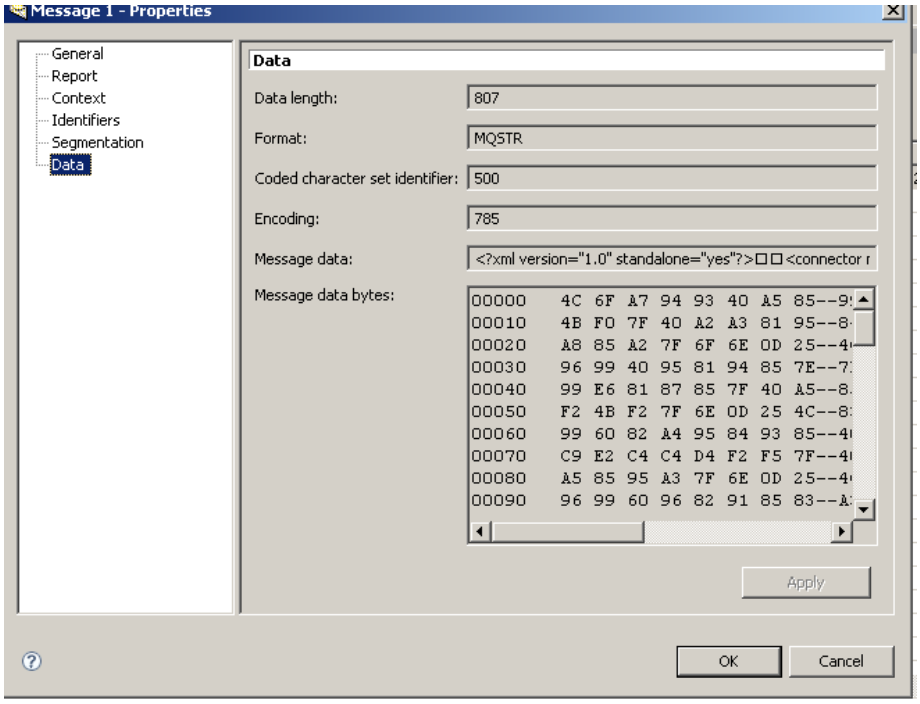
SHOPPING_EVENT_QUEUE



3. Right-click the line showing the message -and click Properties .



4. In the properties window, click Data. We see the message shown in character and hex format.



For the purpose of the book we mark Message data and copy it to.

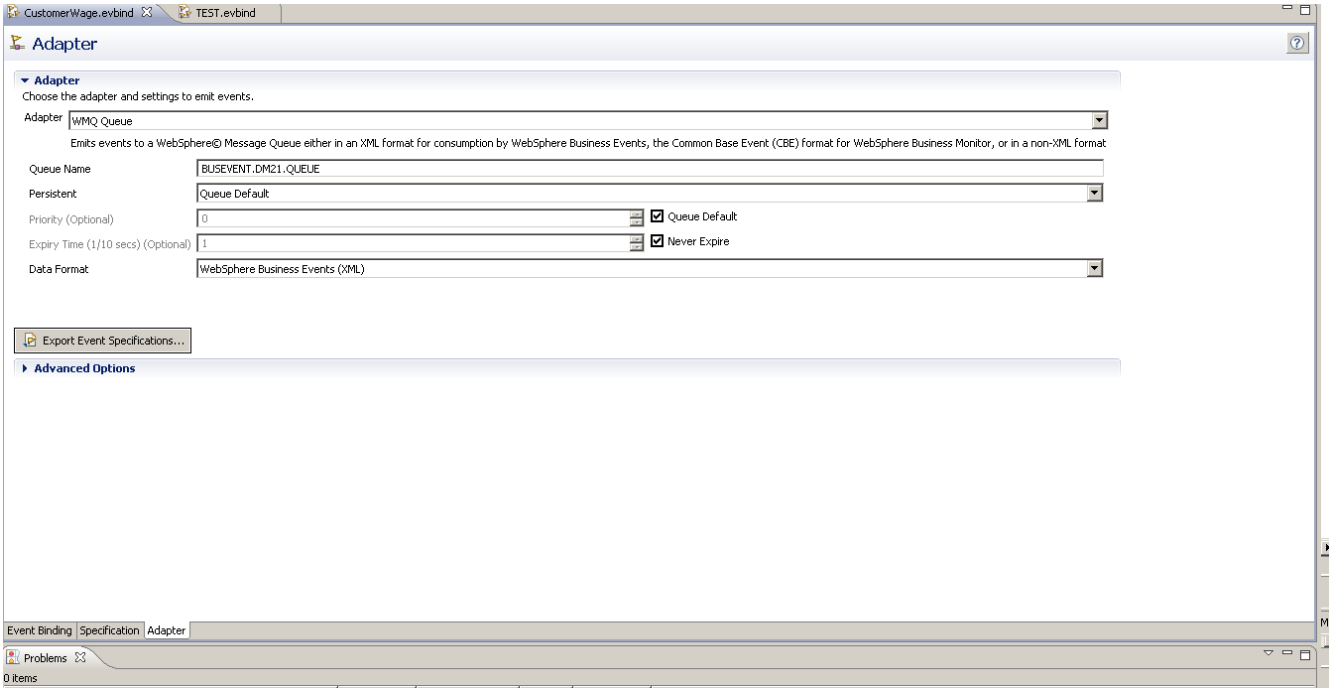
```
<?xml version="1.0" standalone="yes"?>
<connector name="CustomerWage" version="2.2">
<connector-bundle name="OISDDM25" type="Event">
<connector-object name="OISDDM25_Context">
<field name="Binding user tag"></field>
<field name="Network UOWID">170ED4D6D7E9E3F0F04BC9E2E9F0F4F1967FF59230B40001000100</field>
<field name="Capture Spec Name">Delete_Capture</field>
</connector-object>
<connector-object name="OISDDM25_Data">
<field name="Program_Name">OISA1010</field>
<field name="Record_Number">32</field>
<field name="Customer_Name">Bill Hamilton          </field>
<field name="Deleting_Userid">SUCI121 </field>
<field name="Wage">90000.00</field>
</connector-object>
</connector-bundle>
<system>MOPZT00.CICACB21</system>
 <timestamp>2011-04-07T17:48:13+00:00</timestamp>
</connector>
```

Having tested that the event in WebSphere Business Events format has been written to the WebSphere MQ queue, set up WebSphere Business Events to receive this event.

1.3.6.1 CICS parts: Event specs, export event schemas (WBE)

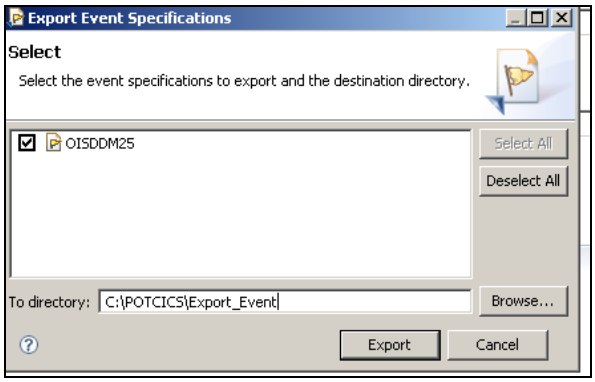
Once the CICS event specifications were defined in the Event Binding Editor within the CICS Explore, we exported them to xsd files so they could be imported in WebSphere Business Events Design Data, which is discussed later in other lab.

To export the event specifications from the Event Binding Editor, open the event binding file and select the adapter tab, as shown :



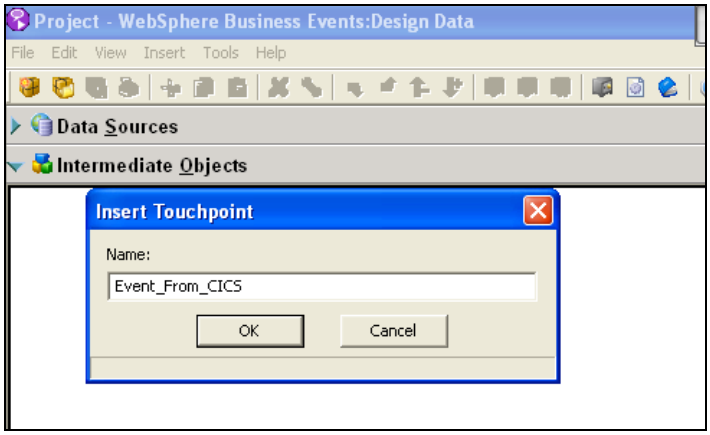
Notice that for export to WebSphere Business Events, the data format was set to WebSphere BusinesEvents (XML). The other choices are CICS Flattened Event (Binary) and Common Base Event (XML). The CBE event format was used when we sent events directly to WBM.

When we clicked on the **Export Event Specifications** button the Export Event Specifications dialog box was displayed.

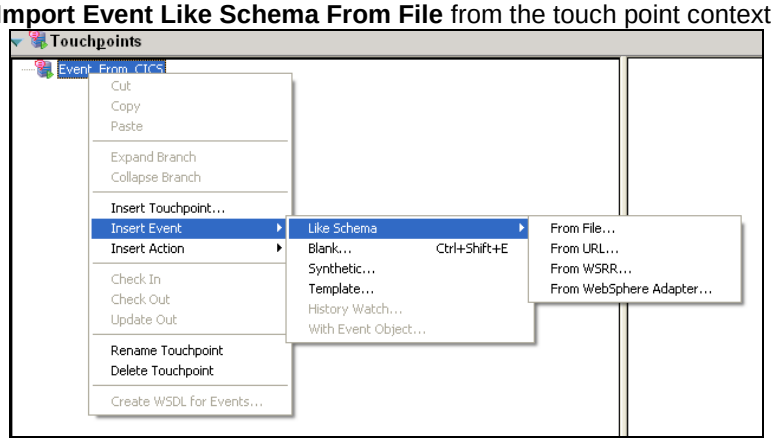


We chose to export the event specifications, and place them in the folder ***C:\POTCICS\Export_Event*** on the workstation.

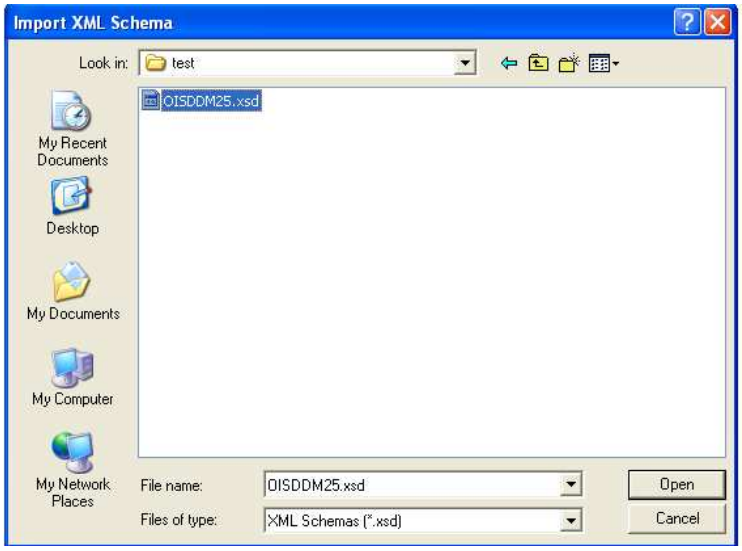
In WebSphere Business Events Design Data, having defined a new touch point named Events_From_CICS, files to import, as shown:



Select **Import Event Like Schema From File** from the touch point context menu,.

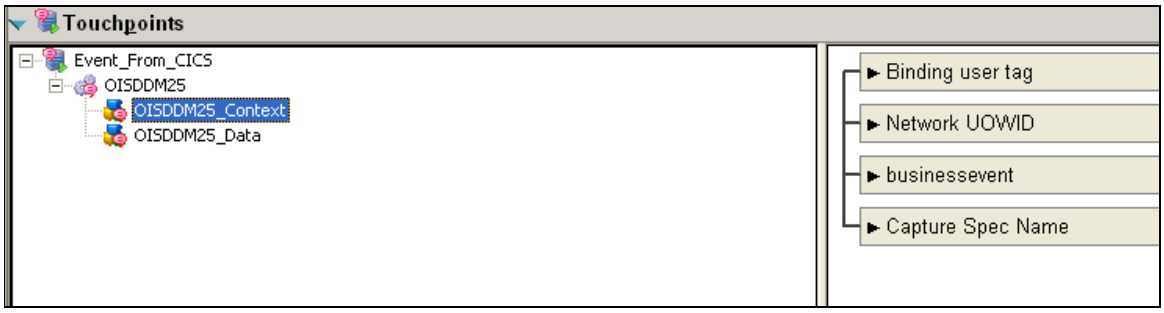


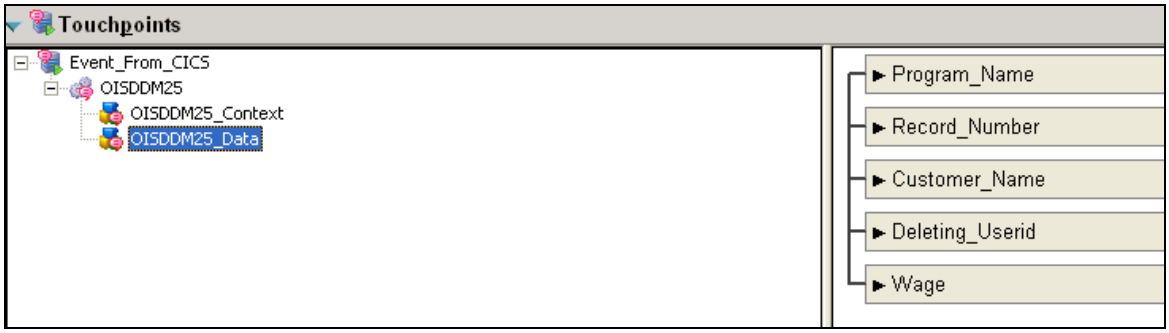
This allows us to select the event xsd



We selected our xsd file for import and clicked Open.

When the message box was closed (by clicking **OK**), the newly created events and events object tree, as shown, were created. In this panel, all the event specifications imported from the CICS Event Binding Editor are expanded



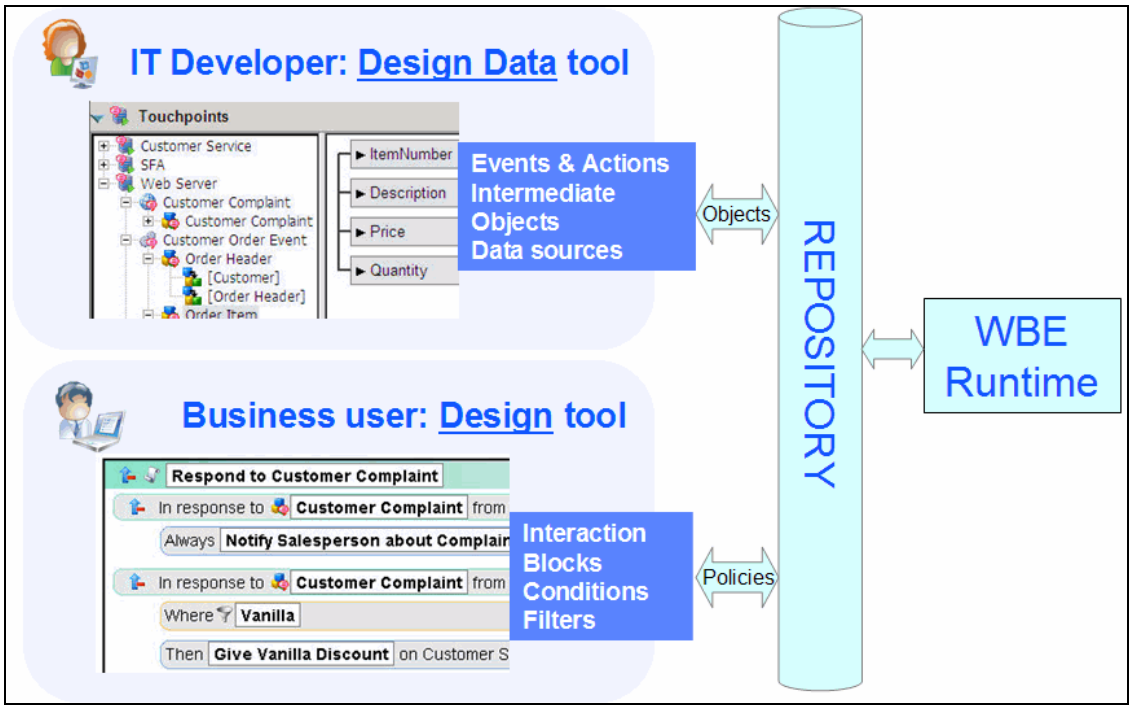


WebSphere Business Events development tooling WebSphere Business Events tooling was designed with two classes of users in mind, the IT developer and the business user.

IT developers use the Design Data tool to create events, actions, intermediate objects, and data source definitions. The skills required to achieve this are primarily technical in nature and require understanding of the protocols and formats of data. Once created, these objects are loaded into a common repository. The WebSphere Business Events Repository is part of the runtime 192 Implementing Event Processing with CICS database used by WebSphere Business Events. In our configuration we used DB2 running on the same z/OS LPAR for our relational database. When using z/OS for the WebSphere Business Events runtime, DB2 on z/OS is the only option for the runtime database. When using WebSphere Business Events on other platforms, other database engines can be used (such as Oracle or Microsoft SQL server).

Business users, through the design tool, retrieve objects from the repository to define the business conditions that link the events and actions (such as: if event A and event B happen with three days and this filter evaluates to true, then start this specific action).

The two WebSphere Business Events development tools, their usage, and how they relate to the WebSphere Business Events runtime is shown in this Figure.



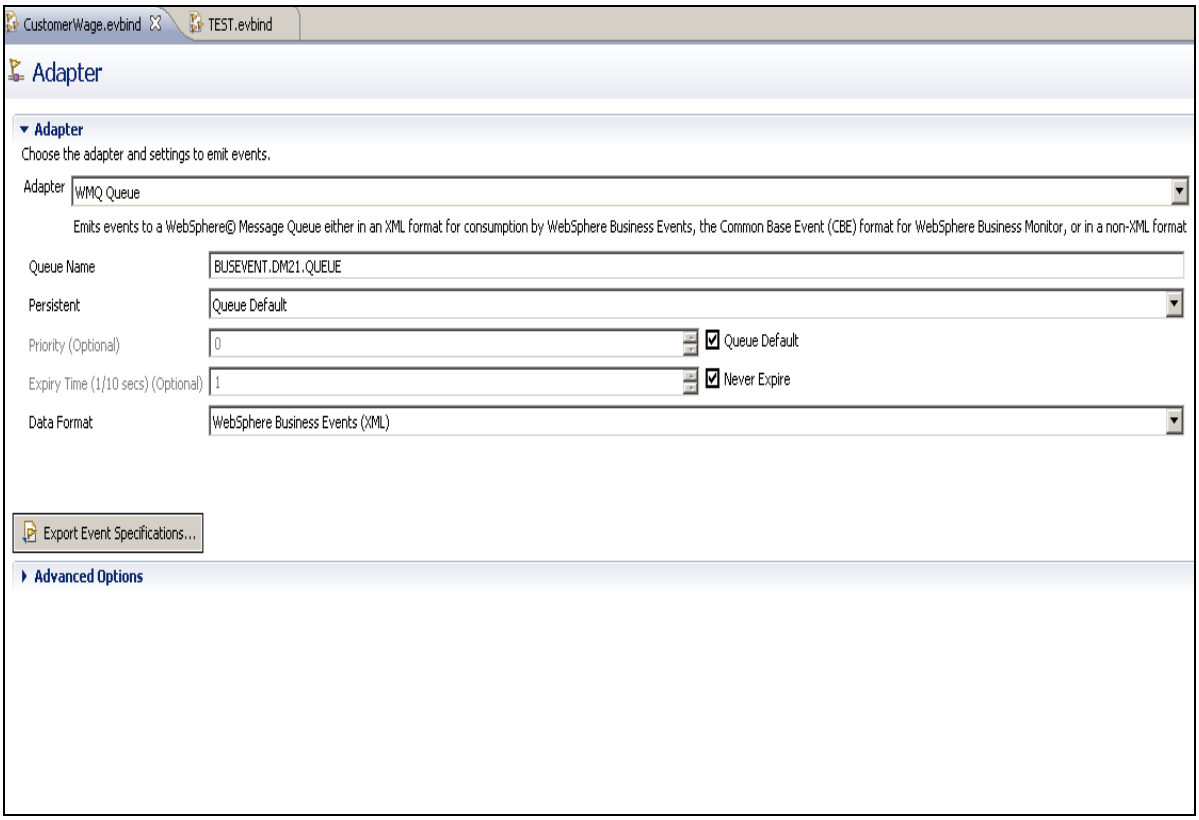
Both Design Data and Design use XML files to store projects. Once the project artifacts are ready to be shared amongst other users and made available to the WebSphere Business Events runtime, they are checked into the WebSphere Business Events repository

1.3.6.2 WMQ setup and WebSphere Business Events configuration for WMQ

CICS events for consumption by WebSphere Business Events are sent from CICS to WebSphere Business Events through a WMQ queue. WebSphere Business Events uses its message queue connection to read the message from WMQ and pass it on to the WebSphere Business Events runtime for processing.

Looking at the Adapter tab in the event binding shows that we defined the adapter as a WMQ adapter (the other choices are TS Queue, Transaction Start or Custom (User Written)), and the queue name as BUSEVENT.DM21.QUEUE.

Note that the WMQ queue manager name will be defined by the connection between CICS and WMQ. In our case,our CICS region connected to the queue manager WMQA at CICS start up.



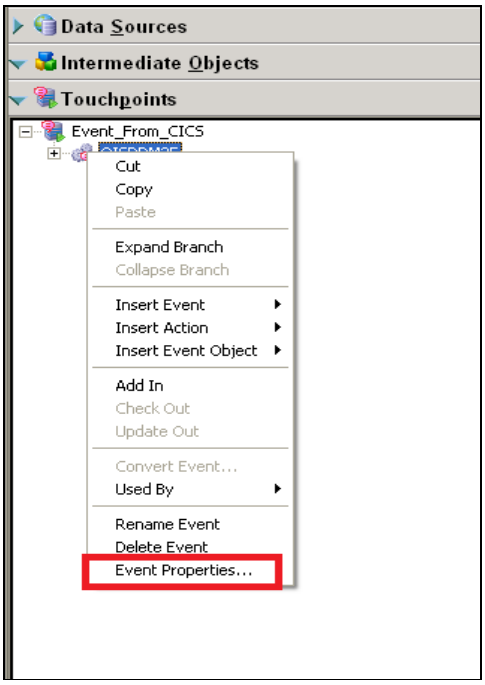
Once the WMQ queue was defined and before we did any configuration in WBE, we defined the JMS Queue Connection Factory and Queue in JNDI. We used the file system JNDI context and located it in the directory ***/u/carruth/mqjndi***. Two options exist for managing the JNDI repository:

_ Using the command line JMSAdmin tool supplied with WMQ

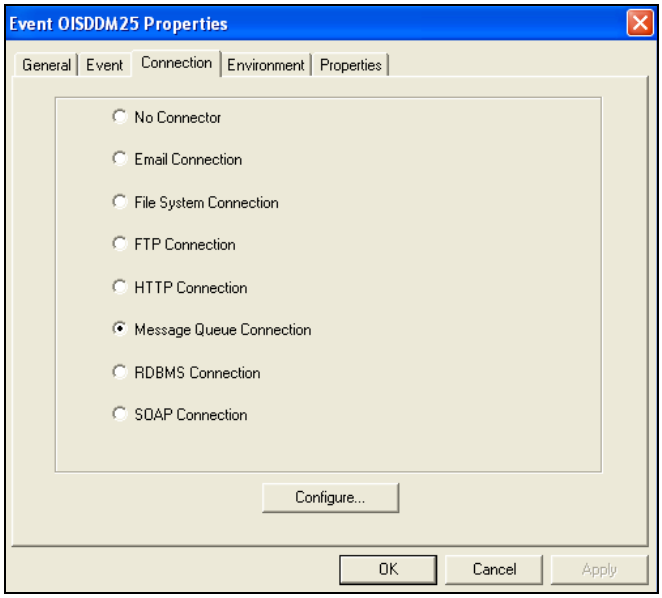
_ Using the JMS Administration facilities provided in WMQ explorer.

In WebSphere Business Events Design Data, we defined the connection to the WMQ queue through the event properties for the CICS OISDDM25 event. The event properties are assessed by selecting **Event Properties** from

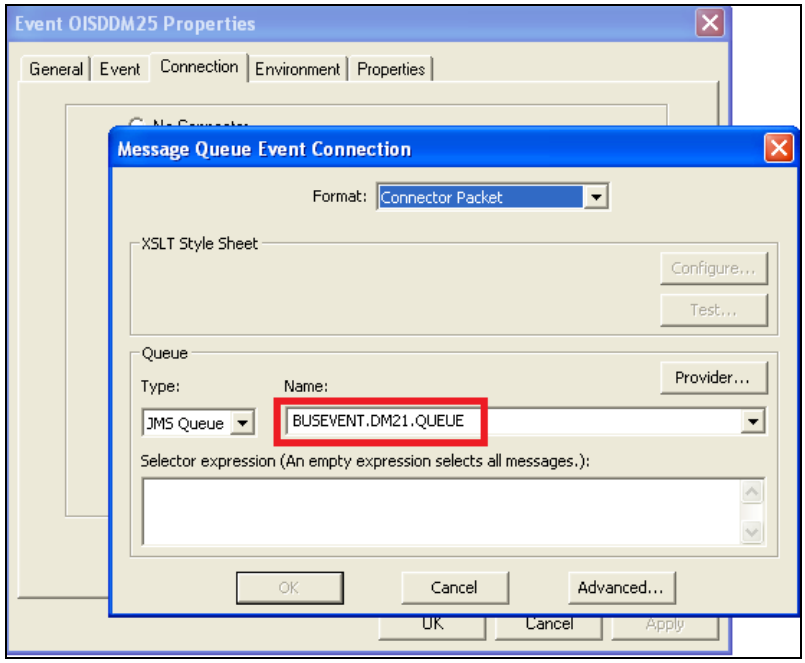
the event context menu.



On the resulting “Event Query Properties” dialog box, we selected the **Message Queue Connection** radio button and clicked **Configure**.



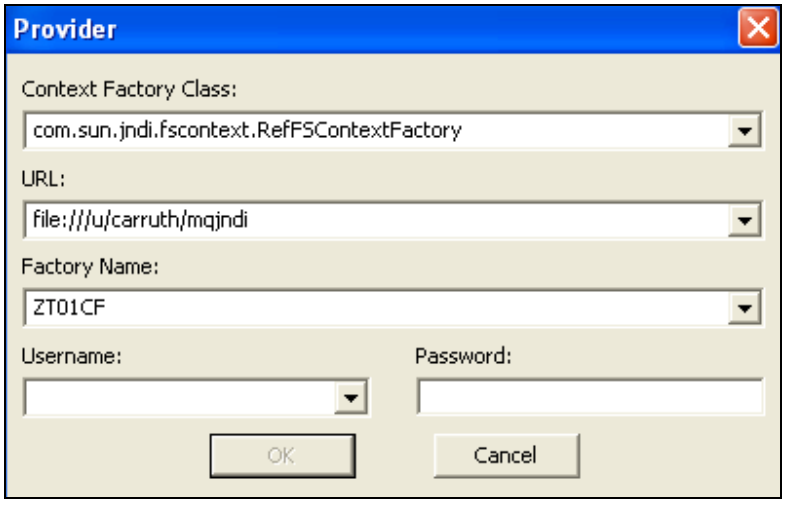
This gave us the “Message Queue Event Connection” dialog box, which allowed us to provide the JMS Queue name, BUSEVENT.DM21.QUEUE.



The **Provider** button is used to define the JNDI provider for the JMS resources.

Clicking this presented the “Provider” dialog box. We used the file system provider that we set up previously, and the ZT01CF Queue

Connection Factory created in that JNDI.



Notice that the user ID and password were not defined, as JNDI security was not enabled at this point.

1.4 Summary

Congratulations, you have just completed the Event Processing lab exercise.
You have now experienced:

- How to ascertain the conditions under which the event will be emitted, the data that the event should contain and how that event should be handled in CICS by an event processing adapter
- How to use the CICS Event Binding Editor to create an bundle containing an event binding that would implement an event based solution · How to move the bundle from the PC to the z/OS UNIX File system and then install into CICS
- How to verify that CICS event processing capability was started and confirm that the event binding was ready for use.
- How an verify that the event processing solution is working correctly by using an EP adapter that writes to a CICS TS queue

- How to select other event emission options

Lab 2 Implementing a Business Rules Using Rule Studio

2.1 Introduction

In this lab, you will work with ILOG Rule Studio to develop a business rule application that you will then deploy to the Rule Execution Server on z/OS. You will use various functions of ILOG Rule Studio.

The tasks you will execute are :

- Developments tasks
    - Designing the rule project
    - Orchestrating
    - Authoring in Rule Studio.

2.2 Design

We suppose the CICS 3270 green screen application has to be modernized in order to integrate these three business rules related to setup "customer loyalty for loan".

- Calculate customer level
- Calculate interest premium
- calculate interest rate

2.2.1 Calculate customer level

The customer level is identified based on the average balance, or wage amount maintained in the bank or the number of years of relationship with the bank, as shown in this table :

Customer since (in years)	Average balance or Wage amount (in $, €, £)	**Customer Level**
> 10	> 100k	**P (Platinum)**
> 5 <10	> 50k <100k	**G (Gold)**
> 3 <5	> 30k < 50k	**S (Silver)**
Other	Other	**R (Regular)**

2.2.2 Calculate interest premium

Depending on the customer level, the interest premium is defined, as shown in this table :

Customer Level	**Interest Premium (in %)**
P	**0**
G	**0.25**
S	**0.5**
R	**0.75**

2.2.3 Calculate interest rate

Depending on the customer credit rating, loan amount and duration of loan, the interest rate is added to the base interest;

Credit Rating	Principle (in $, €,£)	Terms (in years)	**Add interest rate (in %)**
> 540 < 700	< 430 k	15	**1.90**
		20	**2.00**
		30	**2.50**
	> 430	15	**1.95**
		20	**2.05**
		30	**3.00**
> 700 < 830	< 430	15	**1.00**
		20	**1.5**
		30	**1.8**
	> 430	15	**1.25**
		20	**1.75**
		30	**1.85**

2.3 Development tasks

2.3.1 Design the rule project

In this task, you use rule studio to create the vocabulary that will let business users write rules using terms that are meaningful to them. This vocabulary is created directly from the xsd model.

In order for business users to edit and write rules with terms familiar to them, as the rule project developer, you first create the business vocabulary.

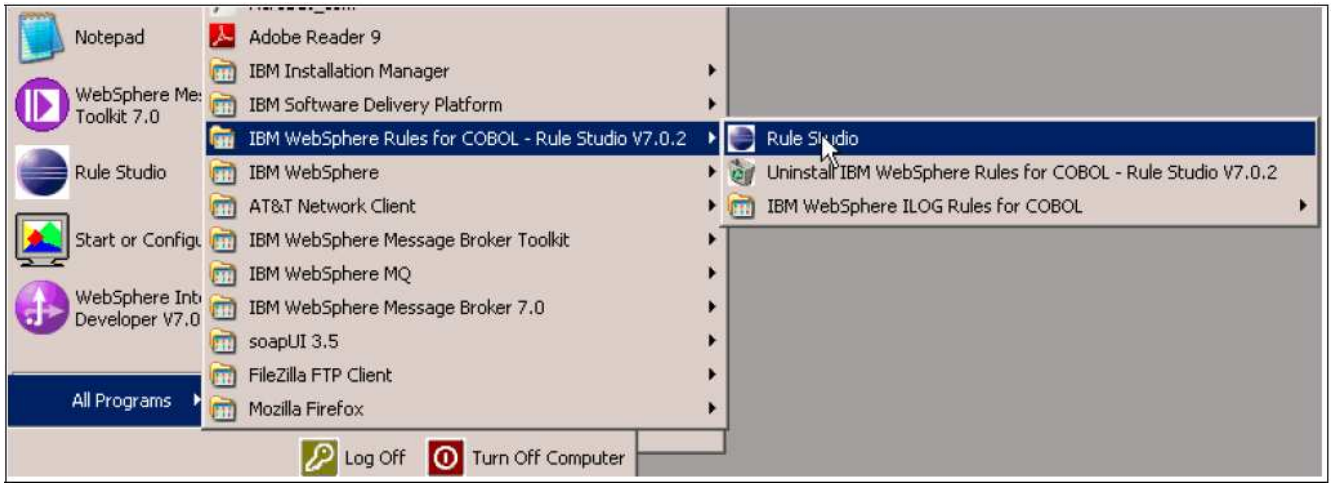
The process of creating this business rule vocabulary is referred to as verbalization. You will create a Business Object Model (BOM) based on a an project model defined in a java project. The class and members of the BOM map to the terms and phrases familiar to the business user, as follows:

1. Starting Rule Studio
2. Creating a Rule project
3. Defining a dynamic XOM
4. Creating the BOM
5. Declaring ruleset parameters

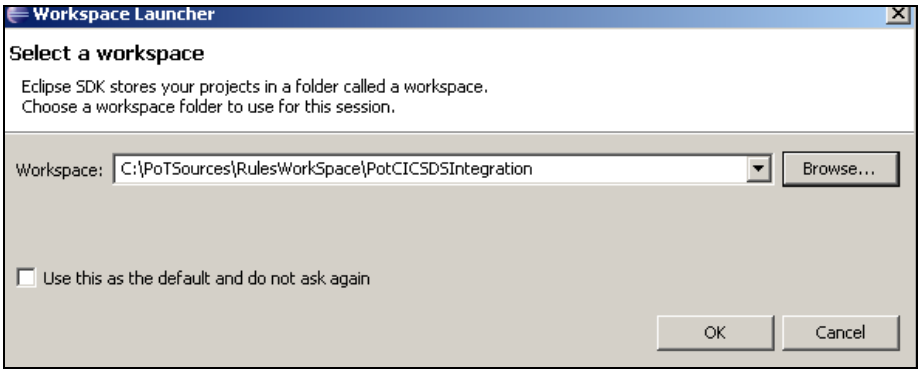
2.3.1.1 Starting the rule project

Rule Studio is the development environment for business rule applications. Developers can take advantage of the integration with Eclipse to develop their projects along with rule projects. Much of the work in this lab is done in Rule Studio.

1 - Launch Rule Studio from the **Start** menu : click **All Programs** --> **IBM WebSphere JRules** (or **IBM WebSphere Rules for COBOL**) --> **Rule Studio**



2 - The WorkSpace launcher dialog shows your default workspace. Change the name of your workspace to **C:\PoTSources\RulesWorkSpace\PotCICSDSIntegration**



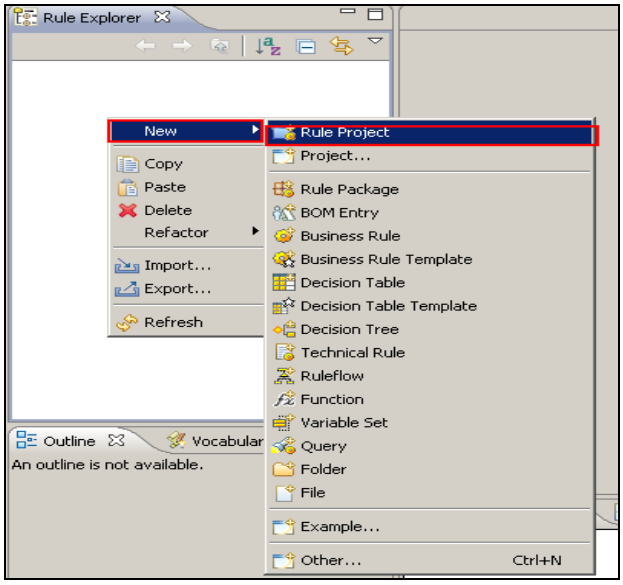
3 - Rule Studio opens in the Rule perspective. Close the Welcome view if you see this pane.

Error! Objects cannot be created from editing field codes.

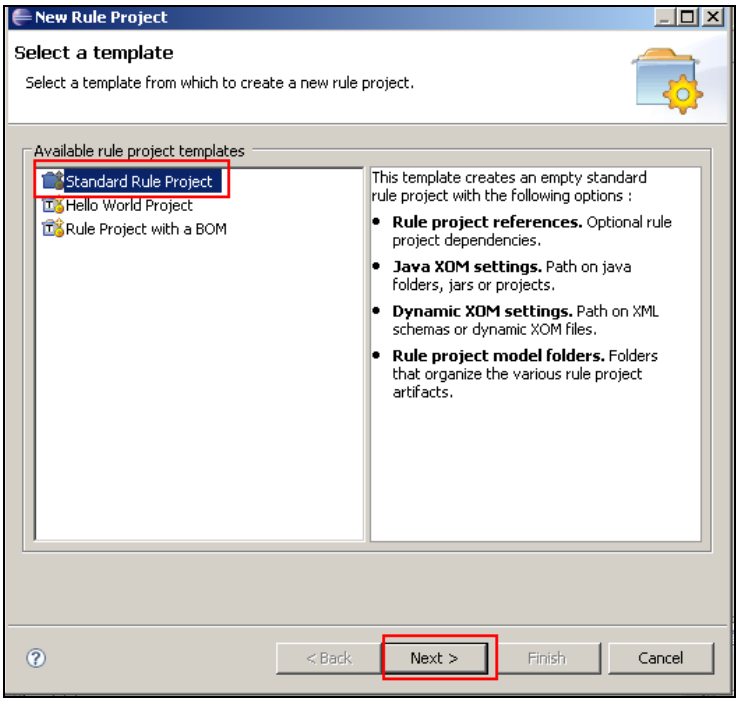
2.3.1.2 Creating a Rule Project

In Rule Studio, you store the business logic of your application in a Rule Project. A Rule project enables you to manage, build, and debug the items that comprise the business logic of your application. To create the rule project.

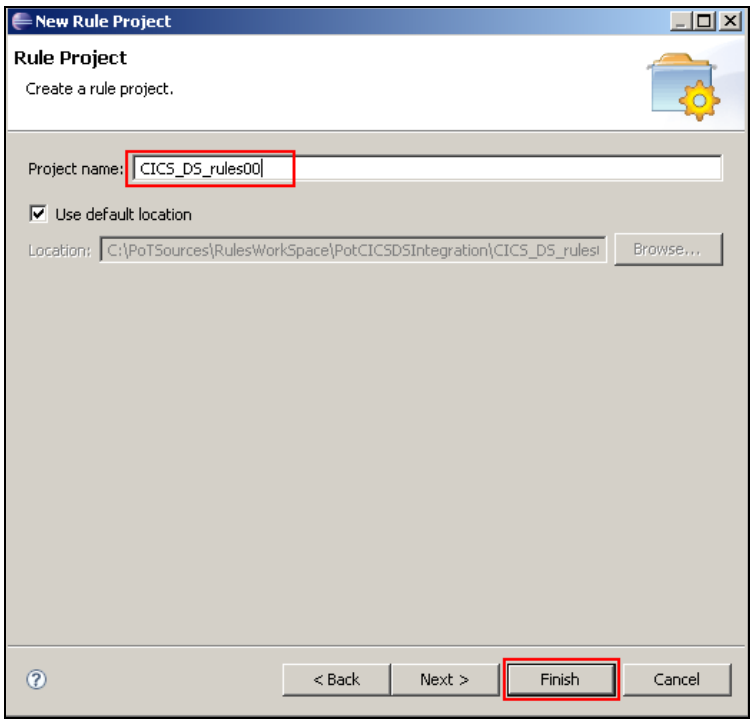
1 - Click File --> New --> Rule Project



2 - In the Select a template pane , select Standard Rule Project and click Next.



3 - In the Rule project pane, in the project name field, type ***CICDS-rulesxx*** (where you need to substitute "xx" with your team number)



4 - Click Finish

The CICSDS-rulesxx project appears in the Rule Explorer, as show in the figure below.

Fo now, the Rule project only contains empty folders. During the lab you will make use of the rules and bom folders to store your rules and your BOM.

__1. Defining a dynamic XOM

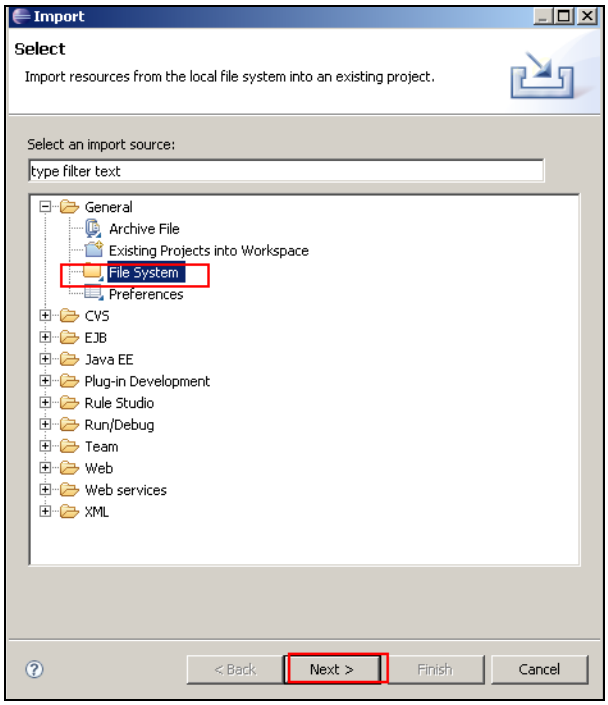
Now that you have an empty Rule project, you can use the Rule Project Map to guide you through the steps of building the project.

You can define a dynamic XOM either in the New Rule Project wizard when you create the project or later, using the Rule Project Properties dialog. The procedure is the same in each case.

Importing the XOM into your rule project

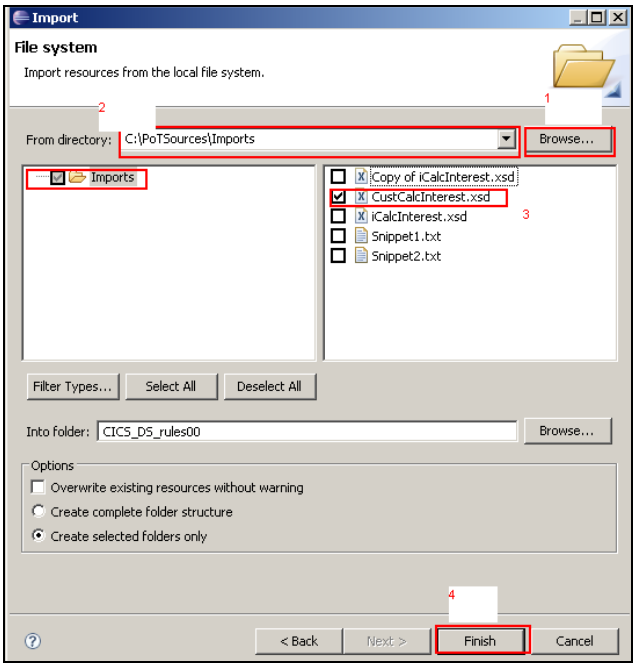
1 - You first need to add the XSD that is held in your file system to the workspace. Select ***CICDS-rulesxx*** (substitute "xx" with your team number) project and select File --> Import.

In the Import dialog that follows, select General --> File System, as shown in the figure below.



2 - Click **Next** . In the next panel, click "**Browse**" and navigate to ***c:\PoTsource\Imports***

Click "OK"

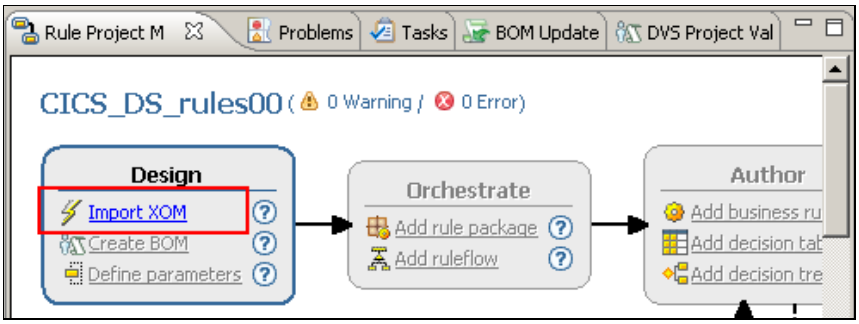


3 - In the File System panel, check Imports and make sure that *CustCalcInterest.xsd* is checked and click "**Finish**"

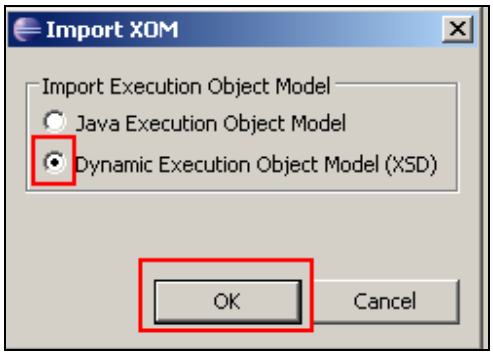
4 - In the Rule Explorer, again make sure that ***CICDS-rulesxx***(substitute "xx" with your team number) project is highligted.

The Rule project Map display the steps to follow to design your rule project.

5 - In the diagram part of the Rule Project Map, Click "**Import XOM**"



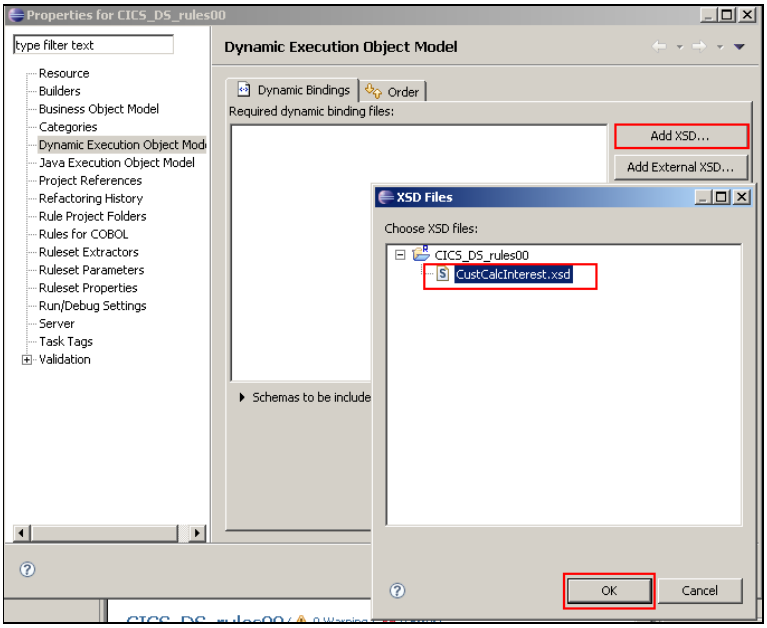
6 - In the Import XOM dialog, select **Dynamic Execution Object Model** (**XSD**) and click **OK**



7 - In the left pane of the properties dialog, click " **Dynamic Execution Object Model"**

8 - In the right pane of the properties dialog, under **Dynamic Execution Object Model,** click on the **Dynamic Bindings** tab.

9 - Click **Add XSD**. Expand ***CICDS-rulesxx***(substitute "xx" with your team number) and select *CustCalcInterest.xsd* from the XSD files dialog then click **OK**



For each schema namespace , a package name is defined in wich the XOM classes will be stored. You can expand each list entry to see the XML namespace and the name of the package.

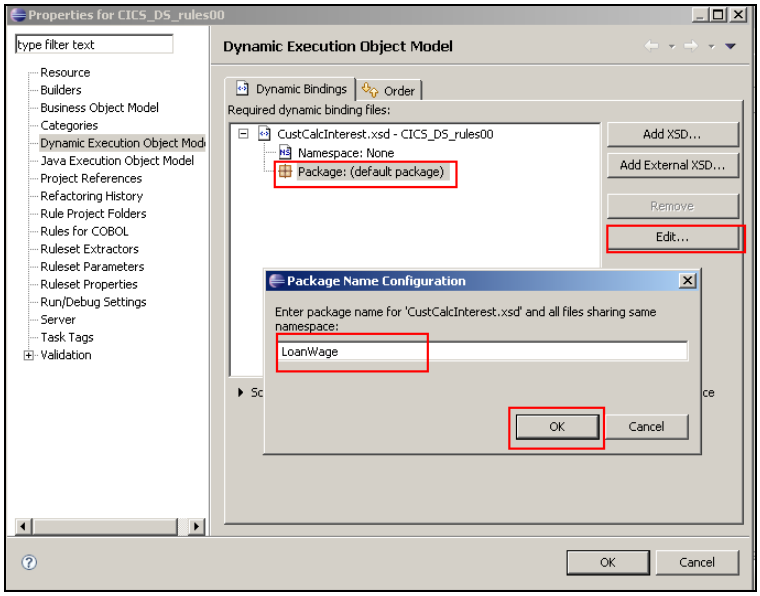
10 - Expand the XSD and you will see the "Package : (Default package)", so without a name basically. rename this package as follow:

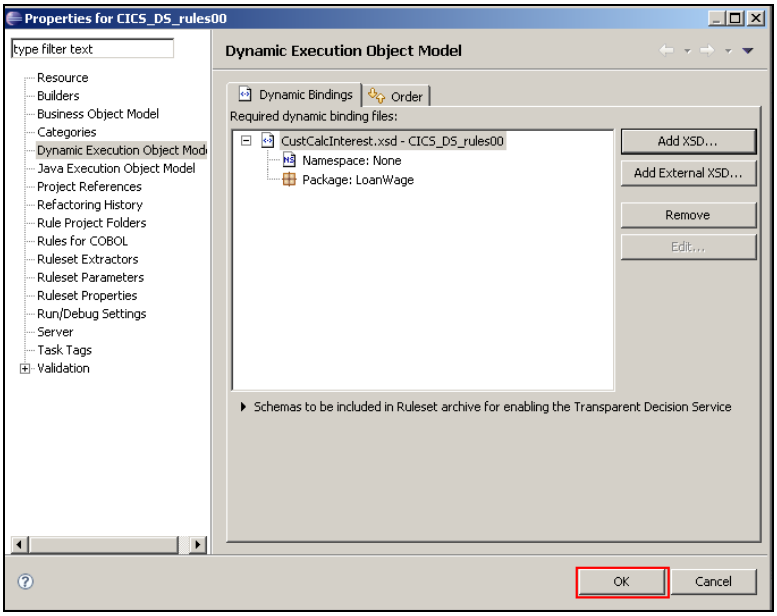
- select the default package and click "Edit"

- Type the new name (ex: Loan, or CICSEP, ...)

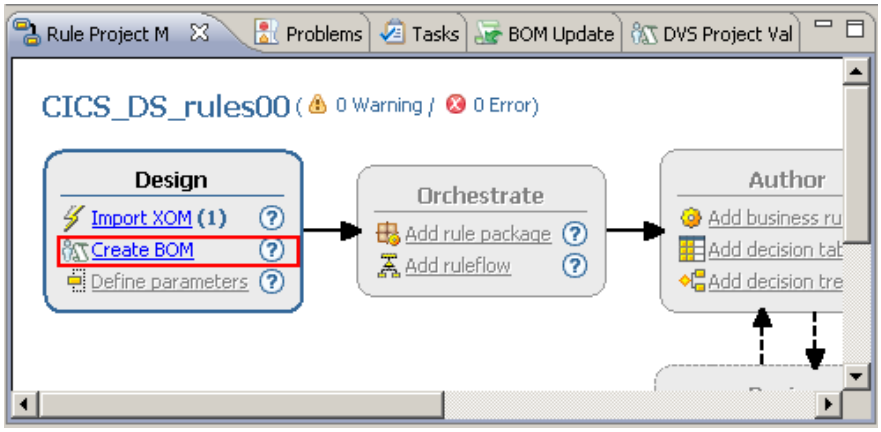
11 - Click **OK** to close.

Figures below show the steps.

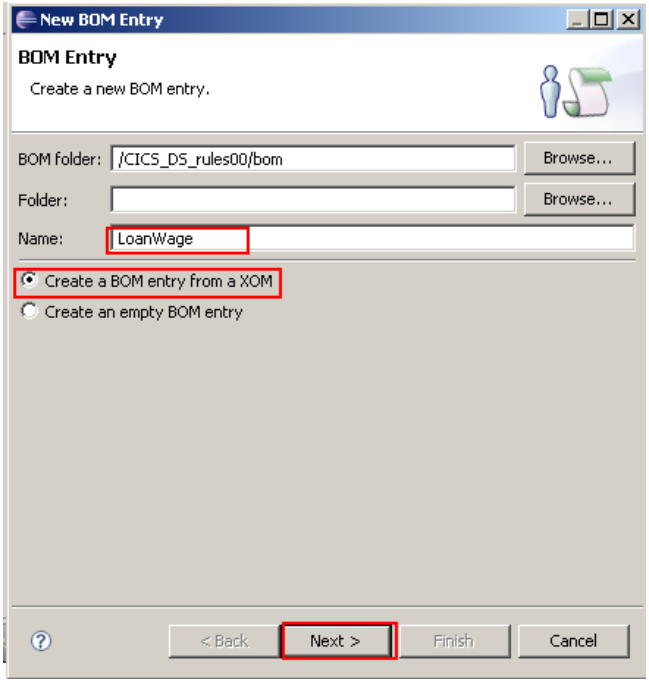

Properties for CICS_DS_rules00
type filter text
Resource
Builders
Business Object Model
Categories
Dynamic Execution Object Mod
Java Execution Object Model
Project References
Refactoring History
Rule Project Folders
Rules for COBOL
Ruleset Extractors
Ruleset Parameters
Ruleset Properties
Run/Debug Settings
Server
Task Tags
Validation
Dynamic Execution Object Model
Dynamic Bindings
Order
Required dynamic binding files:
CustCalcInterest.xsd - CICS_DS_rules00
Namespace: None
Package: (default package)
Add XSD...
Add External XSD...
Remove
Edit...
Package Name Configuration
Enter package name for 'CustCalcInterest.xsd' and all files sharing same namespace:
LoanWage
OK
Cancel
OK
Cancel


Properties for CICS_DS_rules00
type filter text
Resource
Builders
Business Object Model
Categories
Dynamic Execution Object Mod
Java Execution Object Model
Project References
Refactoring History
Rule Project Folders
Rules for COBOL
Ruleset Extractors
Ruleset Parameters
Ruleset Properties
Run/Debug Settings
Server
Task Tags
Validation
Dynamic Execution Object Model
Dynamic Bindings
Order
Required dynamic binding files:
CustCalcInterest.xsd - CICS_DS_rules00
Namespace: None
Package: LoanWage
Add XSD...
Add External XSD...
Remove
Edit...
Schemas to be included in Ruleset archive for enabling the Transparent Decision Service
OK
Cancel

2.3.1.3 Creating the BOM

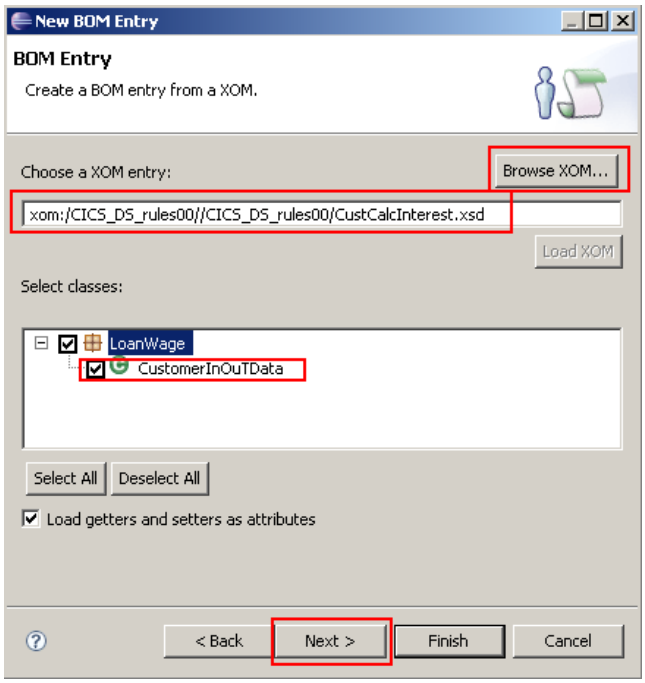


1 - In the Design part of the Rule Project Map, click "**Create BOM**"



2 - In the Create a new BOM entry pane:

- In the name field, type a name (ex : LoanWage)
- Make sure Create a **BOM entry from XOM** is selected and **Folder** is empty
- Click "Next".



3 - In the Create a BOM entry from a XOM pane:

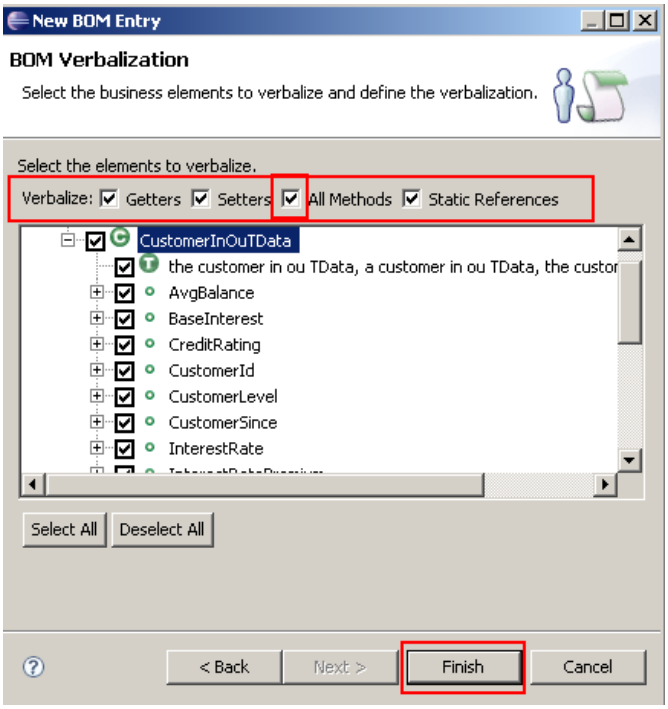
- In the "**Choose a XOM entry**" field, click **Browse XOM**..., select the XSD and then click "OK".

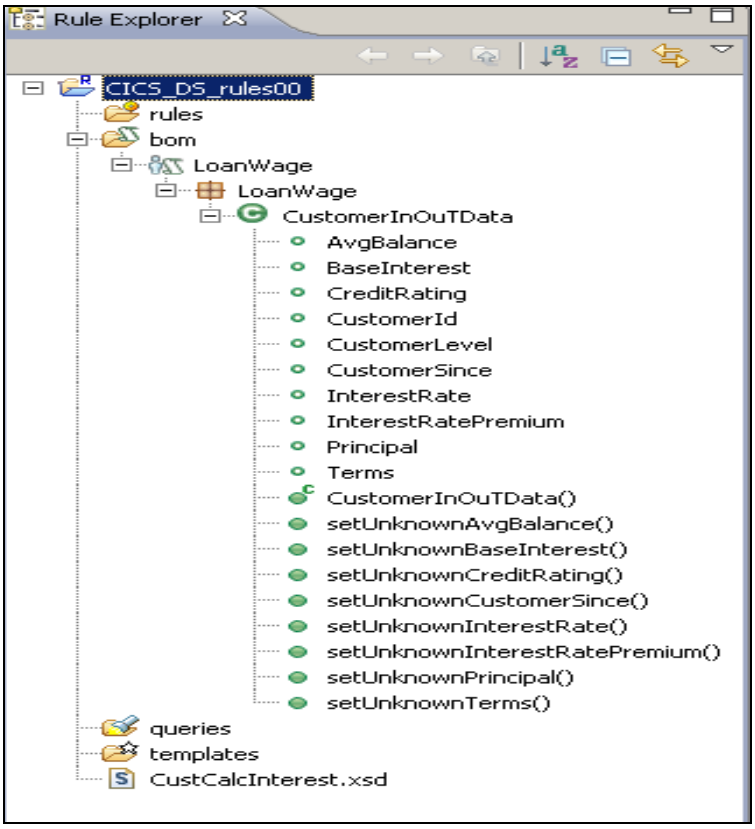
- Under Select classes, select the package name. Selecting the package selects all the classes in the package automatically.

4 - Click Next

5 - In the BOM Verbalization pane, you must select the "**All methods** " check box . This ensures that all methods are verbalized in addition to the elements already selected.

6 - Click **Finish**.

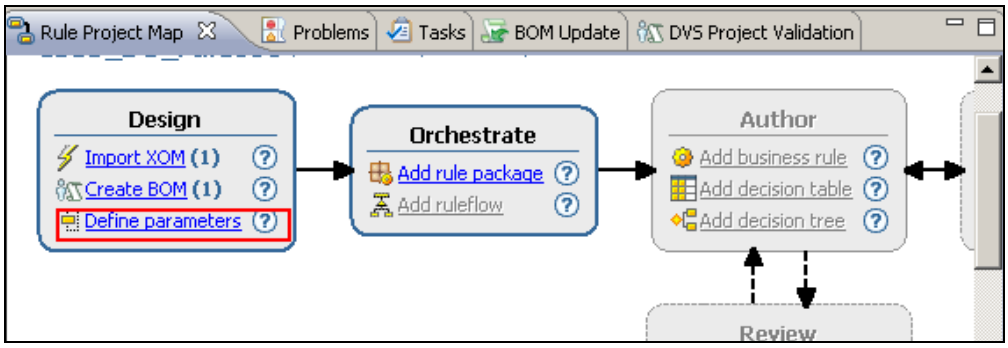

New BOM Entry
BOM Verbalization
Select the business elements to verbalize and define the verbalization.
Select the elements to verbalize.
Verbalize: Getters Setters All Methods Static References
CustomerInOuTData
the customer in ou TData, a customer in ou TData, the custor
AvgBalance
BaseInterest
CreditRating
CustomerId
CustomerLevel
CustomerSince
InterestRate
Select All
Deselect All
< Back
Next >
Finish
Cancel


Rule Explorer
CICS_DS_rules00
rules
bom
LoanWage
LoanWage
CustomerInOuTData
AvgBalance
BaseInterest
CreditRating
CustomerId
CustomerLevel
CustomerSince
InterestRate
InterestRatePremium
Principal
Terms
CustomerInOuTData()
setUnknownAvgBalance()
setUnknownBaseInterest()
setUnknownCreditRating()
setUnknownCustomerSince()
setUnknownInterestRate()
setUnknownInterestRatePremium()
setUnknownPrincipal()
setUnknownTerms()
queries
templates
CustCalcInterest.xsd

2.3.1.4 Declaring ruleset parameters

Ruleset parameters are part of the design of the project because they define the data that is sent to the application and the type of information that can be retrieved.

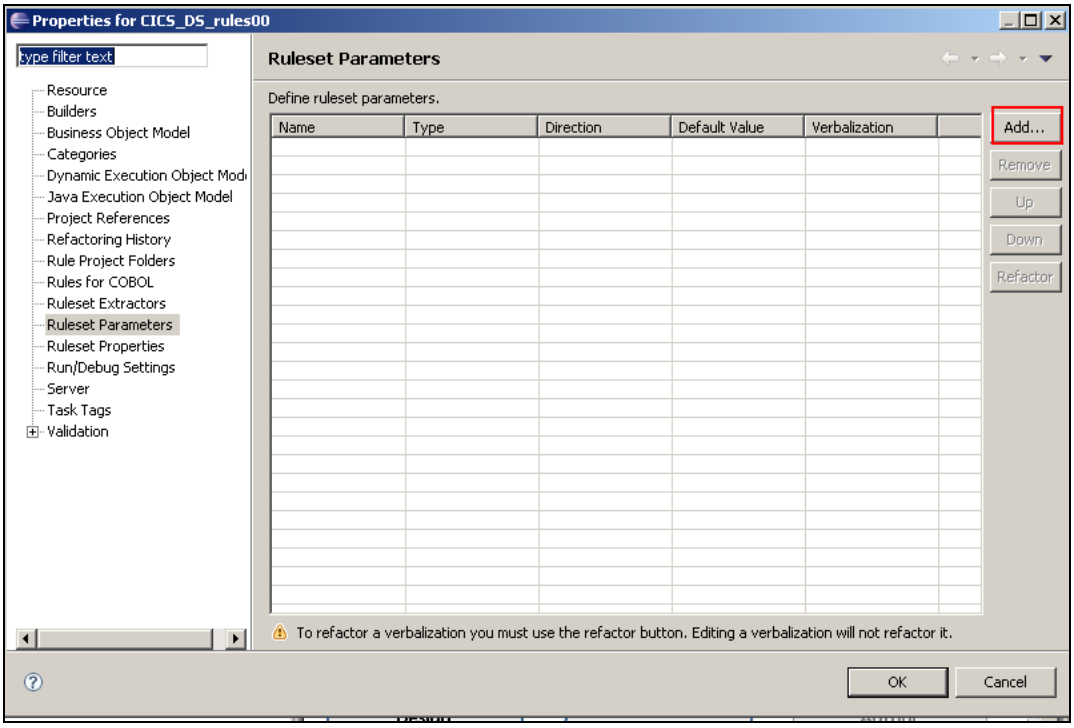
Rule can then use hese parameters to manipulate objects passed to the rule engine.



Perform the following steps:

1 - In the design part of the Rule Project Map, click **Define parameters**

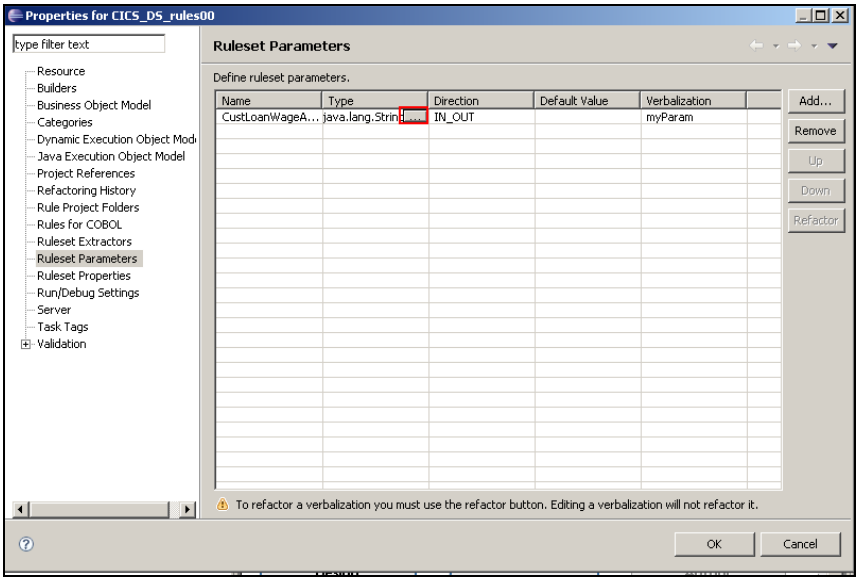
This will open again the standard Properties pane, but positioned at **Ruleset Parameters**.



2 - To define the parameter, click Add.

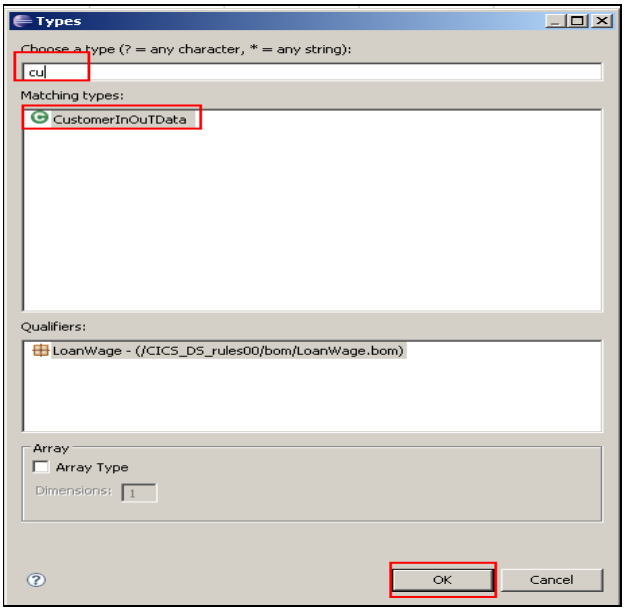
3 - A new row appears with default values. Change the values as follows and as shown in the figure

below.



- In the name column type a Name (ex : **CustLoanWageArgReq**)

- In the Type column click the ... button to display the Types dialog, and double click the **CustomerInOuTData** type in the matching types box.



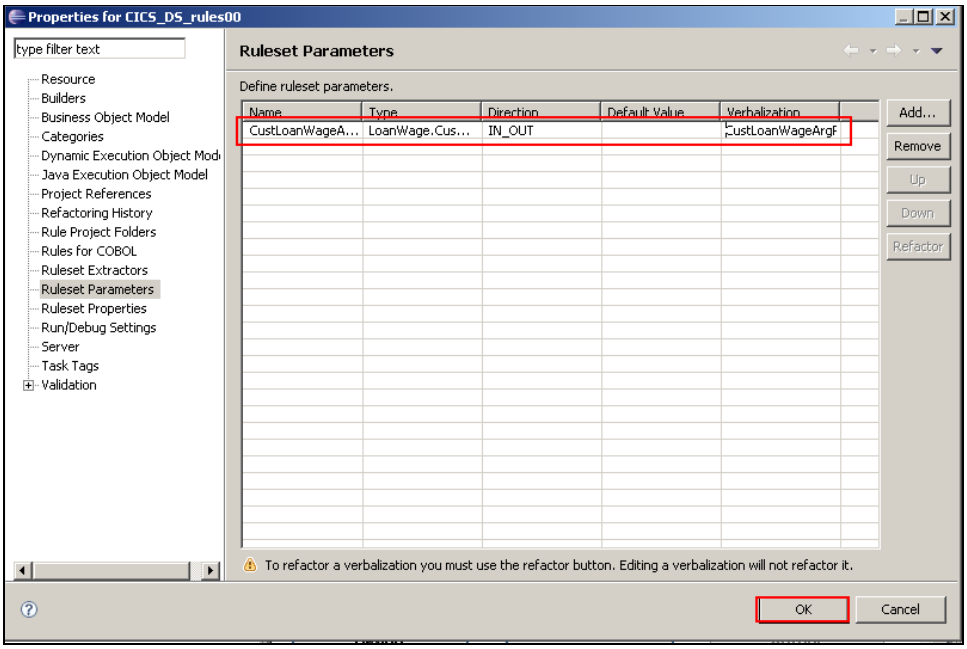
- In the Direction column, select the IN_OUT direction.

- In the Verbalization column, type **CustomerInOuTData**

5 - Click **Ok** to save

With a Rule Project , a vocabulary, and ruleset parameters, you have completed the design part of your project.

Before writing the actual Rules in the Rule studio, you will orchestrate how your Rules will be executed. You do this with a rule flow in the next task



Your ruleset parameters appears as show in fugures below.

2.3.2 Orchestrating

In this task, you use Rule Studio to orchestrate rule execution using Ruleflow.

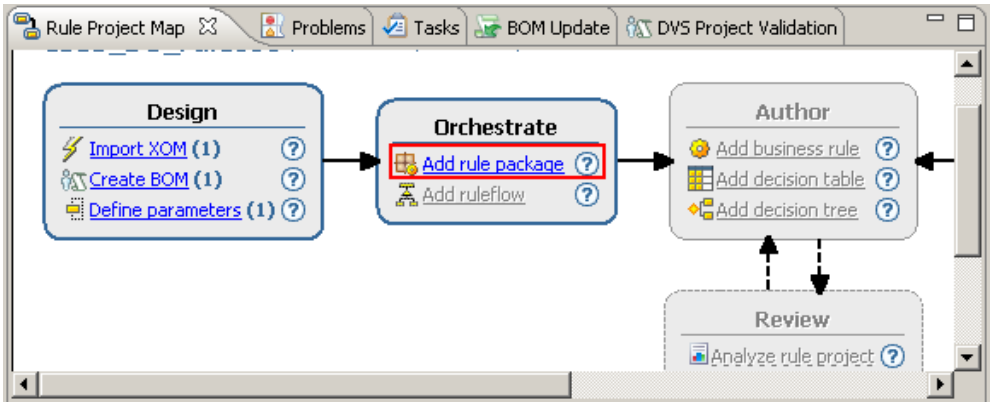
To specify the order in which rules are executed, you must create a ruleflow. A ruleflow is a way to organize the sequence in which rules are processed by the rule engine.

The steps are :

1 - Creating Rule packages

2 - Creating the ruelflow diagram

3 - Editing the ruleflow


Rule Project Map
Problems
Tasks
BOM Update
DVS Project Validation
Design
Import XOM (1)
Create BOM (1)
Define parameters (1)
Orchestrate
Add rule package
Add ruleflow
Author
Add business rule
Add decision table
Add decision tree
Review
Analyze rule project

2.3.2.1 Creating Rule packages

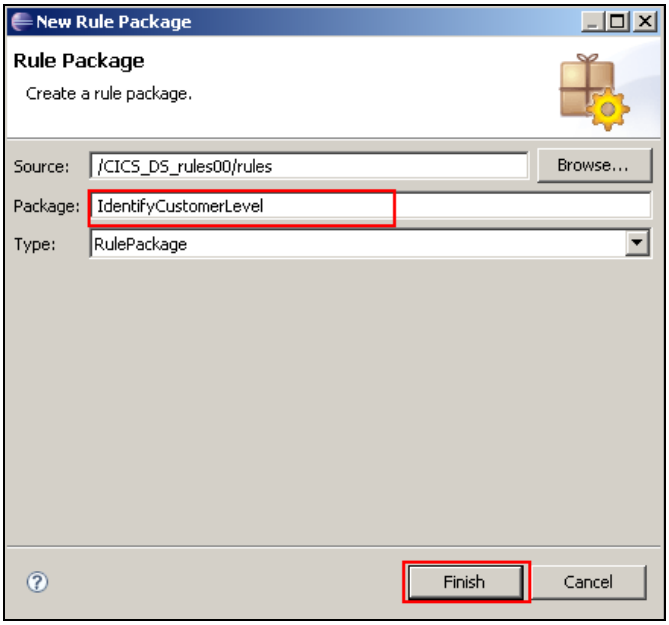
In this case , you have a package of rules into packages that contain related to *IdentfyCustomerLevel*, *SetInterestPremium* and *setInterestRate*.

And also a compute node to calculate the final interest rate for customer. You then treat these Rule packages as tasks in the ruleflow.

Creating a Rule package.

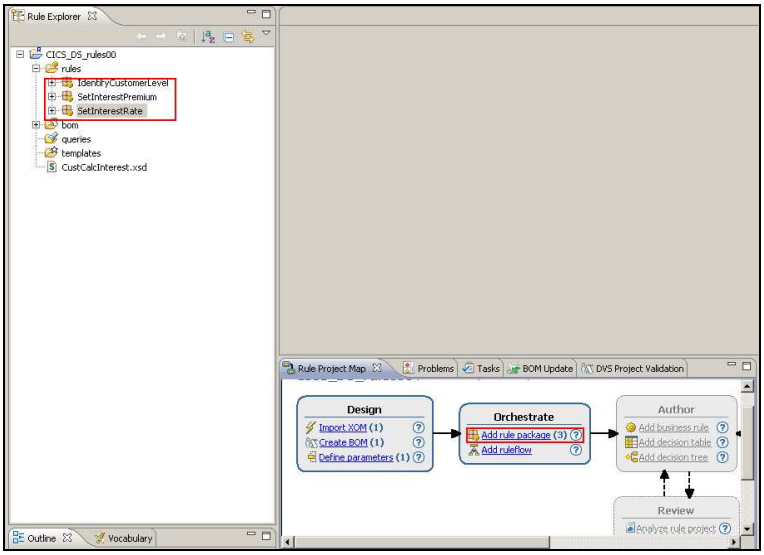
Perform the following steps :

1 - In Rule studio , in the Orchestrate part of the Rule Project Map, click **Add rule package**



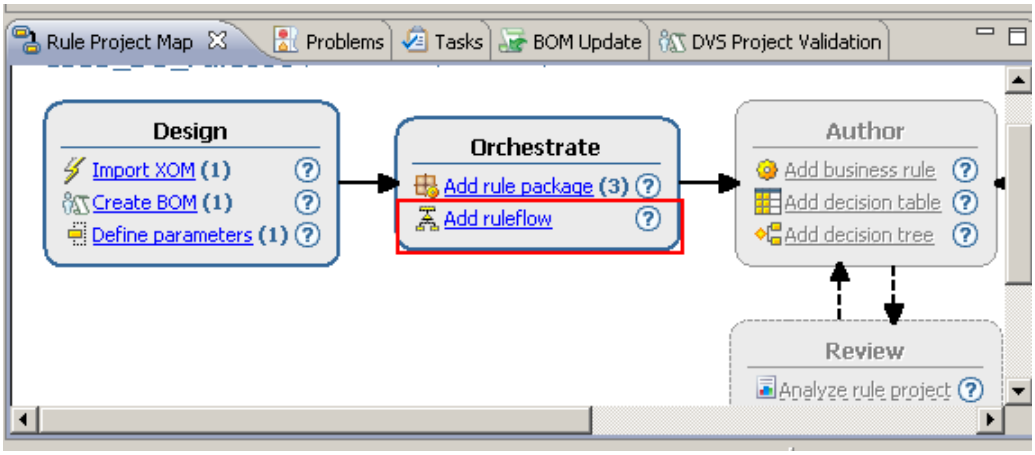
In the Create a rule package wizard, name the package **IdentifyCustomerLevel** and then click **Finish**

3 - Create two more packages named *SetInterestPremium* and *SetInterestRate*,


Rule Explorer
CICS_DS_rules00
rules
IdentifyCustomerLevel
SetInterestPremium
SetInterestRate
bom
queries
templates
CustCalcInterest.xsd
Outline
Vocabulary
Rule Project Map
Problems
Tasks
BOM Update
DVS Project Validation
Design
Import XOM (1)
Create BOM (1)
Define parameters (1)
Orchestrate
Add rule package (3)
Add ruleflow
Author
Add business rule
Add decision table
Add decision tree
Review
Analyze rule project

*

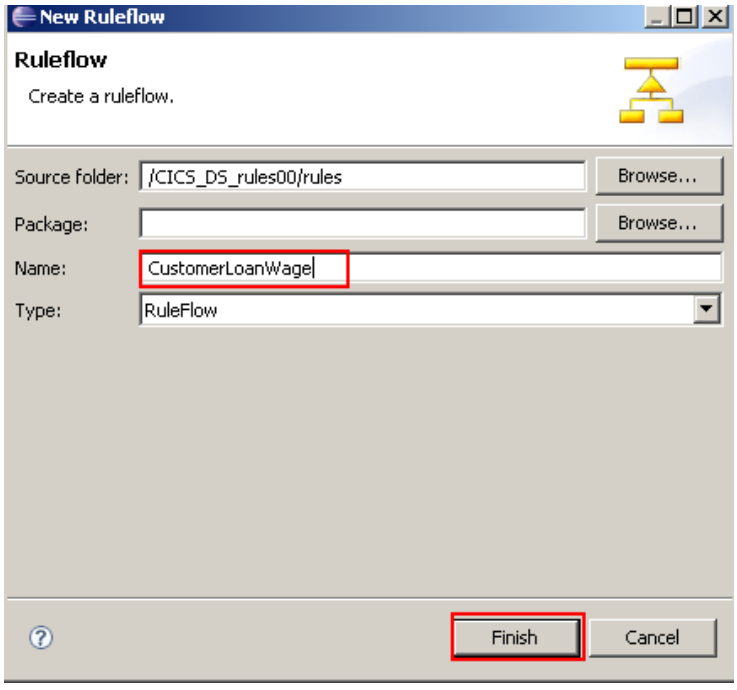
2.3.2.2 Creating Ruleflow diagram



Perform the following steps:

1 - In the orchestrate part of the Rule project Map, click **Add ruleFlow**

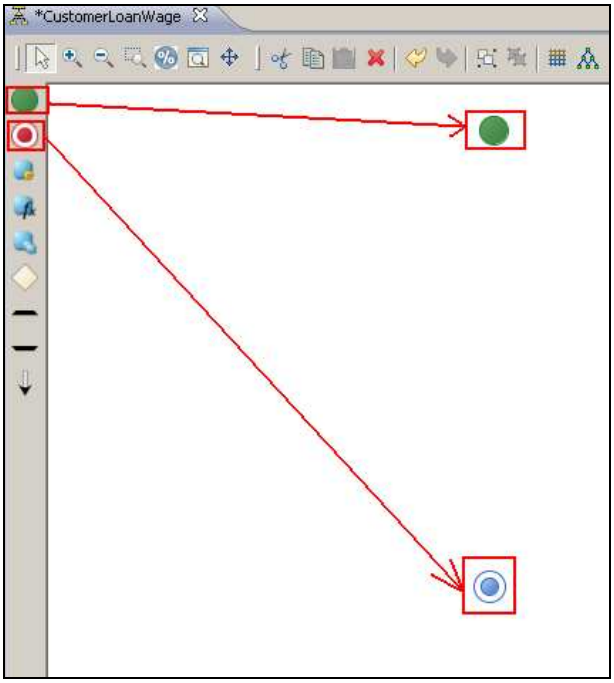
2 - in the Name field, type for example : *CustomerLoanWage*



3 - Click **Finish**

The Ruleflow Editor opens, and enables you to construct the flow of the tasks graphically.

You specify how tasks are chained togheter ; how, when, and under what conditions they will be executed.

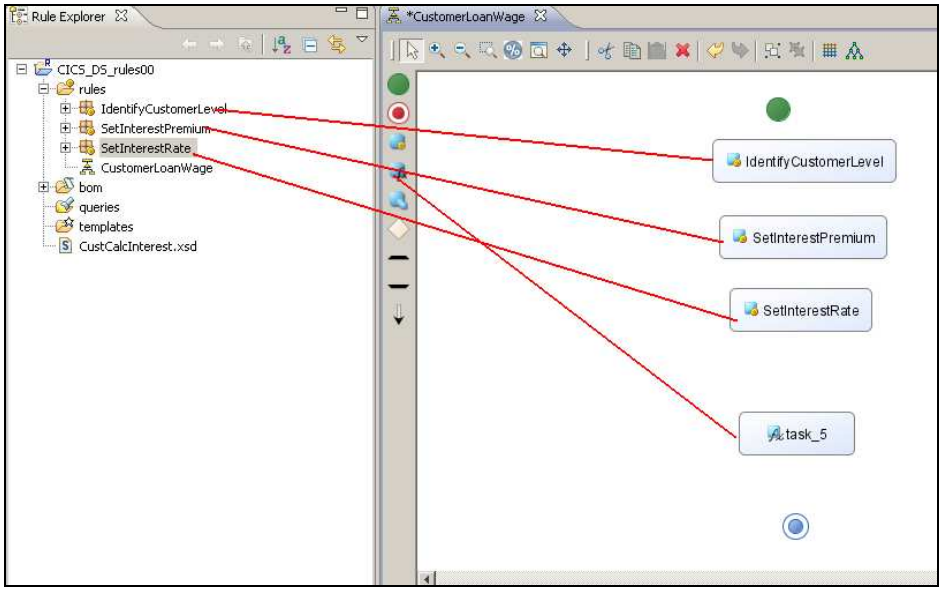


4 - Click **Create Start Point** and and then click somewhere in the upper center part of the ruleflow Editor.

5 - Click **Create End Point** and then click somewhere in the lower center part of the Ruleflow Editor.

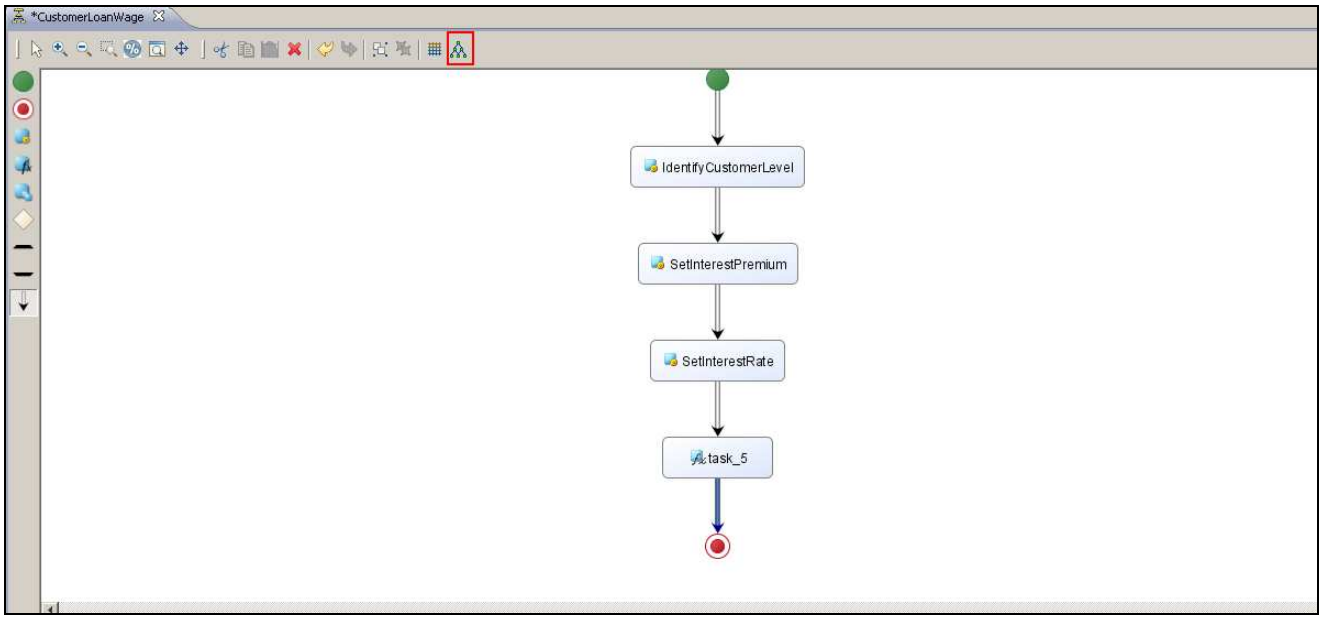
6 - Drag the *IdentifyCustomerLevel* rule package from the Rule explorer and drop it into the ruleflow editor, just below the Start Point.

7 - Drag the *SetInterestPremium* and *SetInterestRate* rule package from the Rule explorer and drop it into the ruleflow editor, just below the *IdentifyCustomerLevel*.



8 - Click **Function Task** and then click in the Rule Editor, just below the *SetInterestRate*.

The default label is "task_5"



9 - Click **Create Transition** and create the following transitions(shown as arrows) by clicking the first and then clicking the second

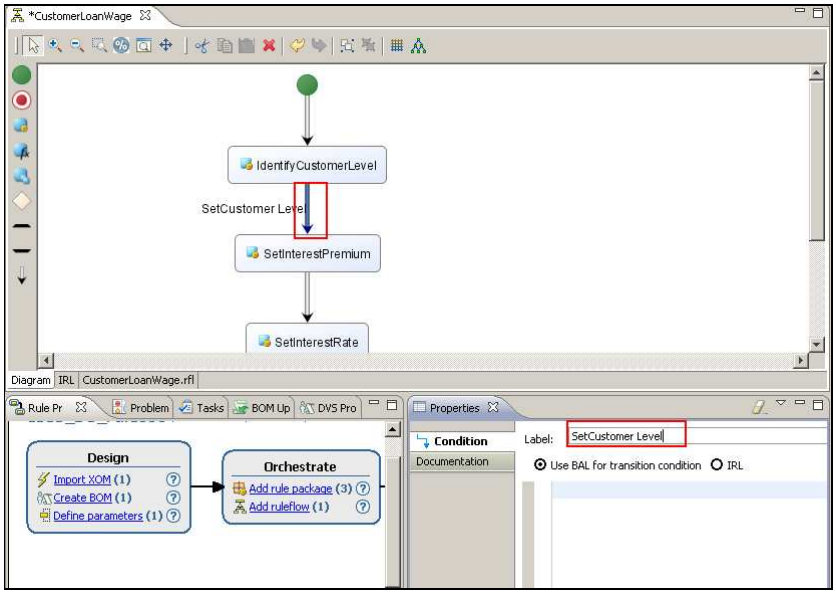
- The Start Point and the **IdentifyCustomerLevel** task

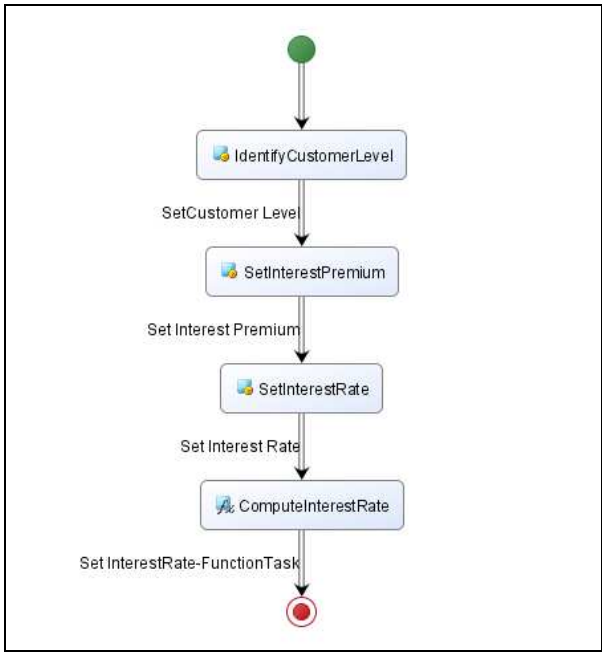
- The **IdentifyCustomerPremium** task and the **setInterestPremium** task

- The **setInterestPremium** task and the **SetInterestRate** task

- The **setInterestRate** task the **Function Task** task

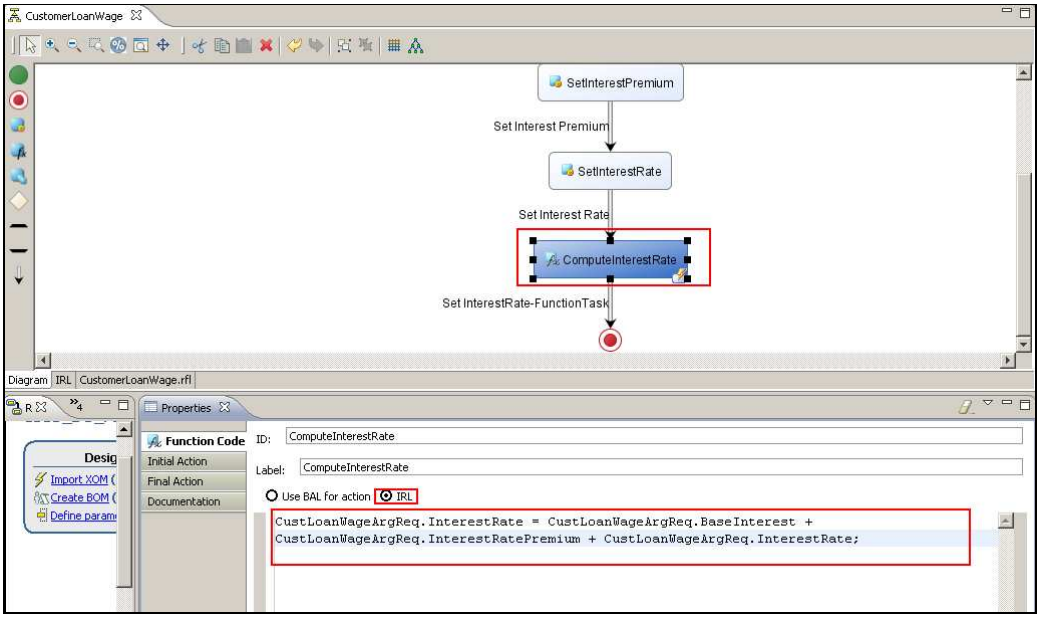
- The **Function Task** and the **End Point.**





10 - Click **Create Transition** to deselect the transition tool

11 - Click **Layout All Nodes**



2.3.2.3 Editing the ruleflow

- Double Click the transition from *IdentifyCustomerLevel* to *SetInterestPremium*

- Click Windows ==> Show View ==> Properties to open the properties view and set the condition for this transition.

- In the Label field, type Set Customer Level

- Similar set the label field to *set Interest Premium* and *Set Interest Rate* in the transitions *setInterestPremium* , *setinterestRate* and *setInterestRate-FunctionTask* respectively.

- click on the Function Tasl (task_5)

- In the Properties pane, set the Label field to **ComputeInterestRate**

- select the **IRL** option, which stands for "**ILOG Rule language**".

- Type the following code as a single line.

```
CustLoanWageArgReq.InterestRate = CustLoanWageArgReq.BaseInterest +
CustLoanWageArgReq.InterestRatePremium + CustLoanWageArgReq.InterestRate;
```

Note : In case this code is not correct, select the argument *CustLoanWageArgReq* and CTRL+Space and select the proper argument in the list.

- Save the changes

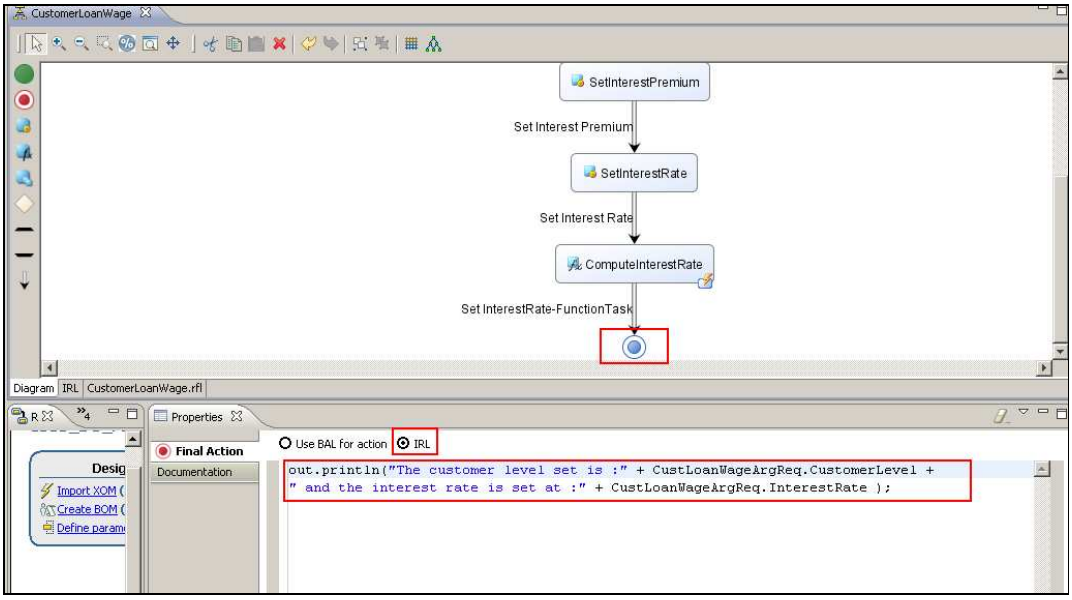
- Double-click on the **End Point**.

The properties view lets you enter the final action.

In the final Action section, select IRL , and type in the following

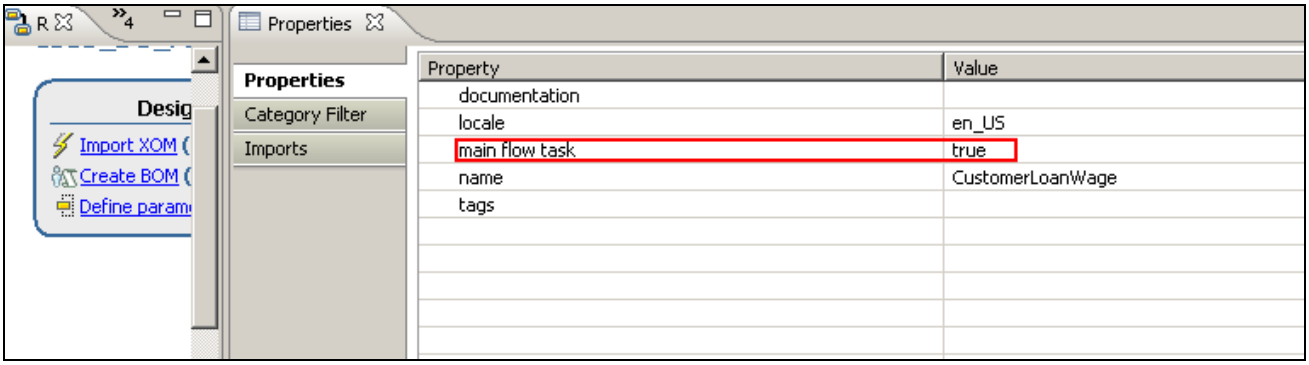
```
out.println("The customer level set is :" + CustLoanWageArgReq.CustomerLevel + " and the interest rate is set at :" + CustLoanWageArgReq.InterestRate );
```

Note : In case this code is not correct, select the argument *CustLoanWageArgReq* and CTRL+Space and select the proper argument in the list.



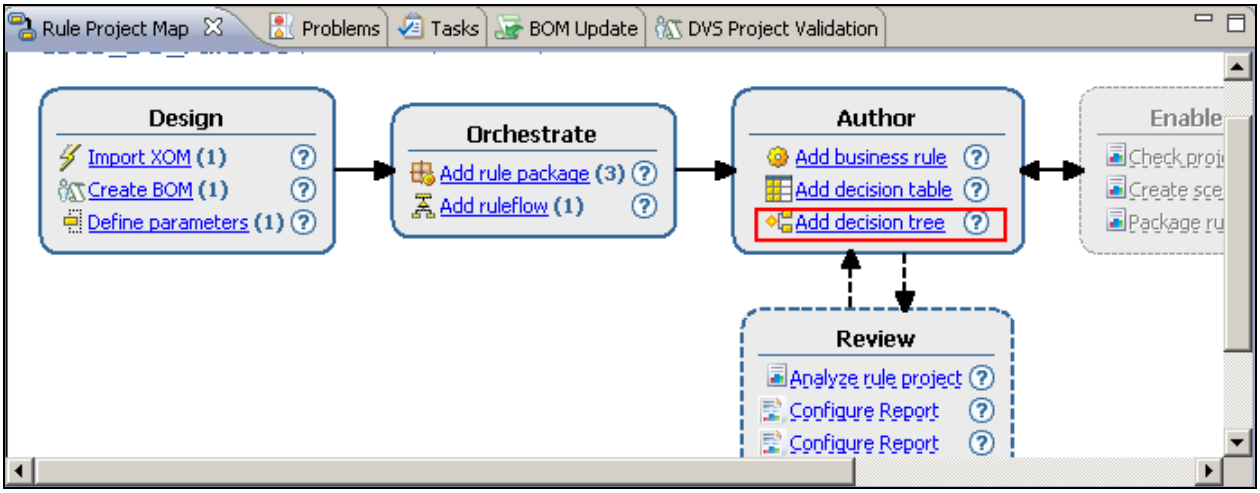
At the execution time, this final action will display a message in the Console indicating the status of the loan at the end of the rule execution.

- in the ruleflow editor, click outside the diagram, and in the Properties view, make sure that the main flow task is set to **true**.



___ 1. Authoring

In this task, you use Rule studio to author the rule in an it-then layout using the vocabulary that you created.



- Creating a business rule - IdentifyCustomerLevel using a Decision Tree

- Creating a business rule - SetInterestPremium using a Decision Table

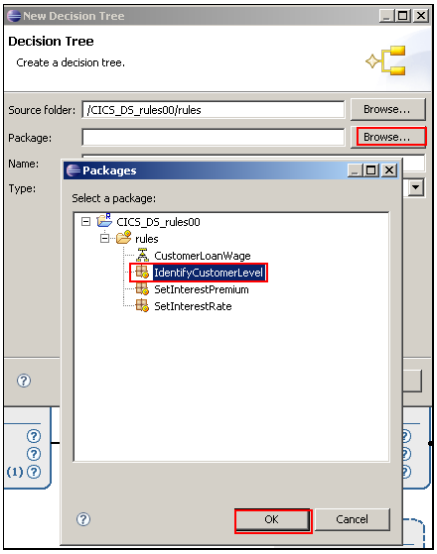
- Creating a business rule - SetInterestRate using a Decision Table

___ 1. - Creating a business rule - IdentifyCustomerLevel using a Decision Tree

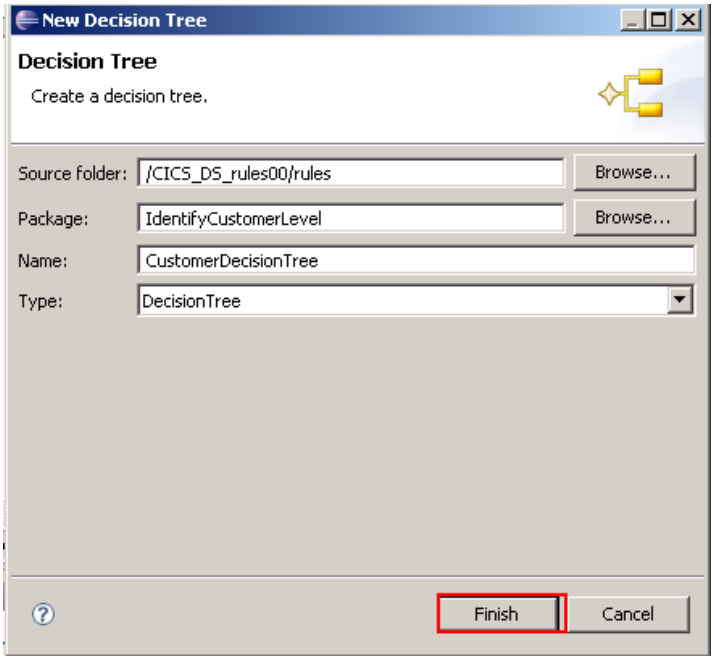
Perform the following steps:

- Select Rule project and in the file menu click New--> Decision Tree

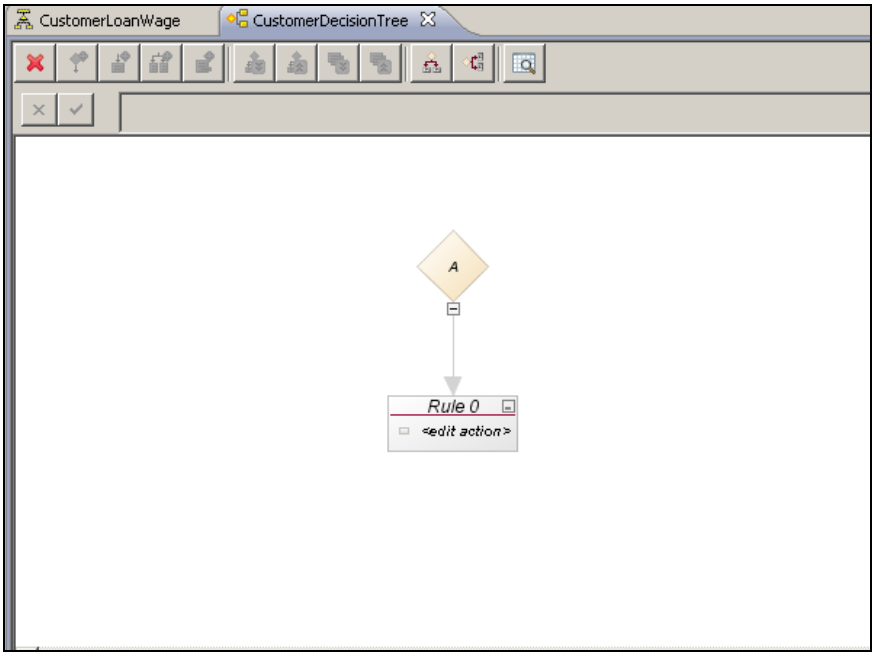
- Browse next to the Package field and navigate to Rule package *IdentifyCustomerLevel*. Click OK



Type *CustomerDecisionTree* in the name field

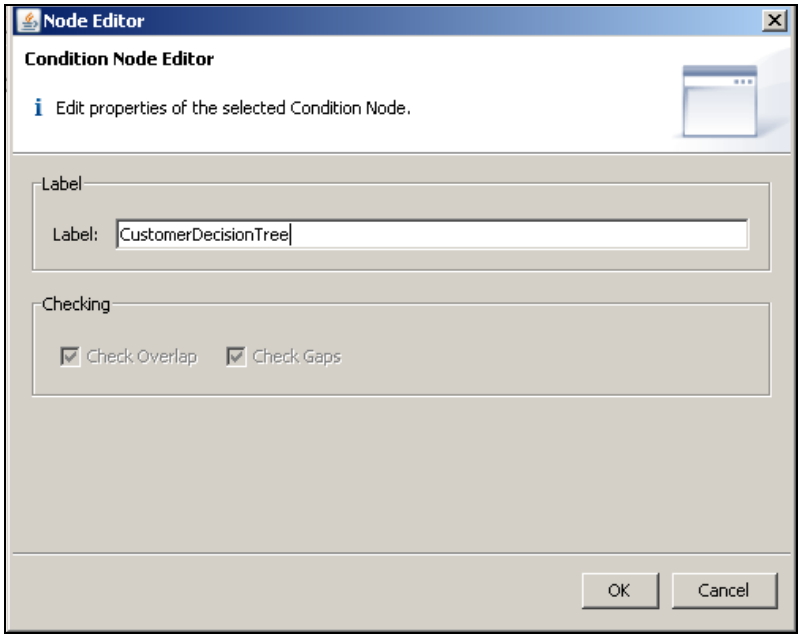


- Click Finish



The new decision tree appears in the Rule Explorer view and the Decision Tree Editor opens.

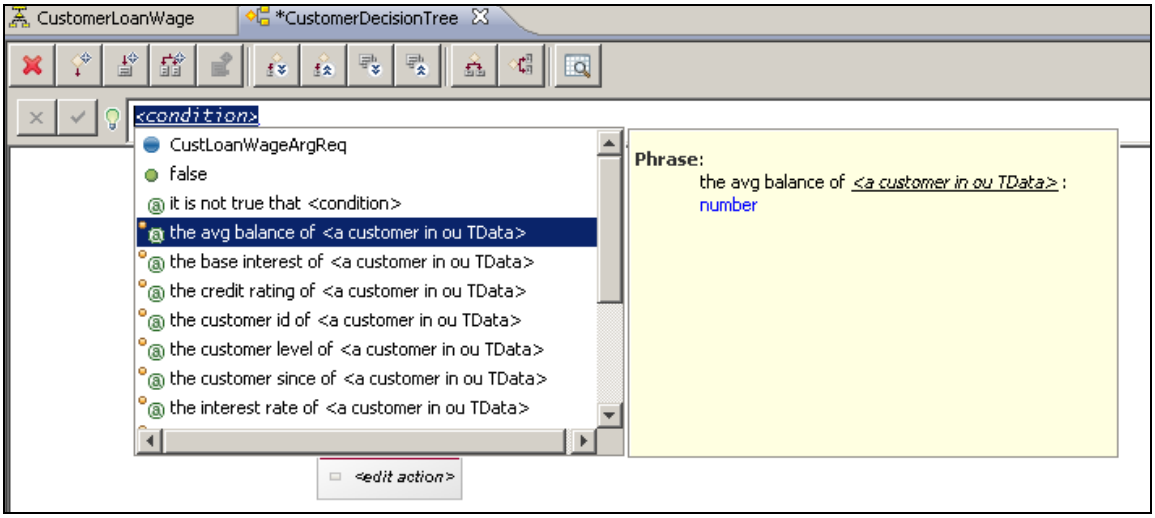
Label a condition node



In the Decision Tree Editor, double - click on the condition node.

Double click the on the Condition node

In the Label field type ***CustomerDecisionTree***

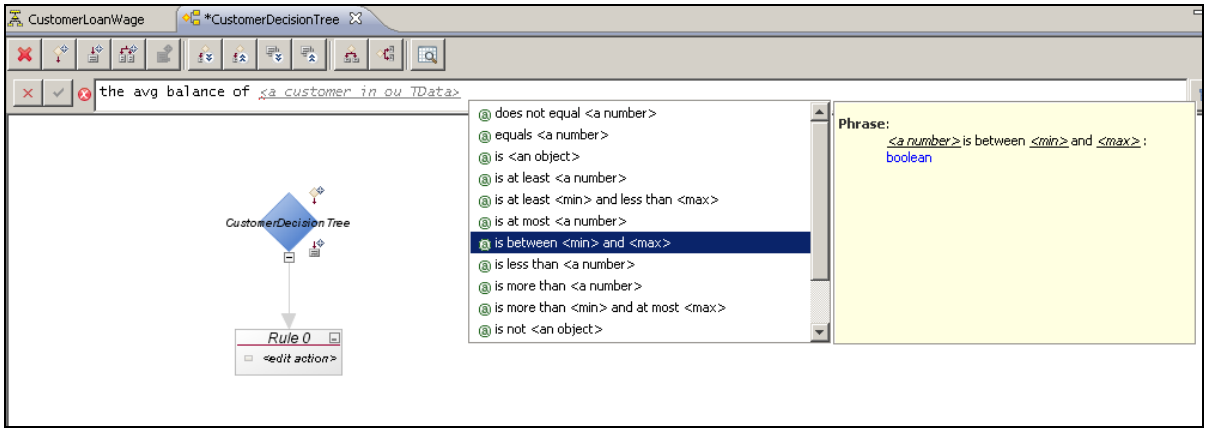


In the edit bar, you will now see the word <**condition**> . click on it.

Select from the drop down list **the avg balance of < a customer in ou T Data >** and double click. The first part of the condition statement is now visible in the **edit** bar.

Press **Ctrl + space**, and select from the drop-down list **is between <min> and <max>**

Double-click.

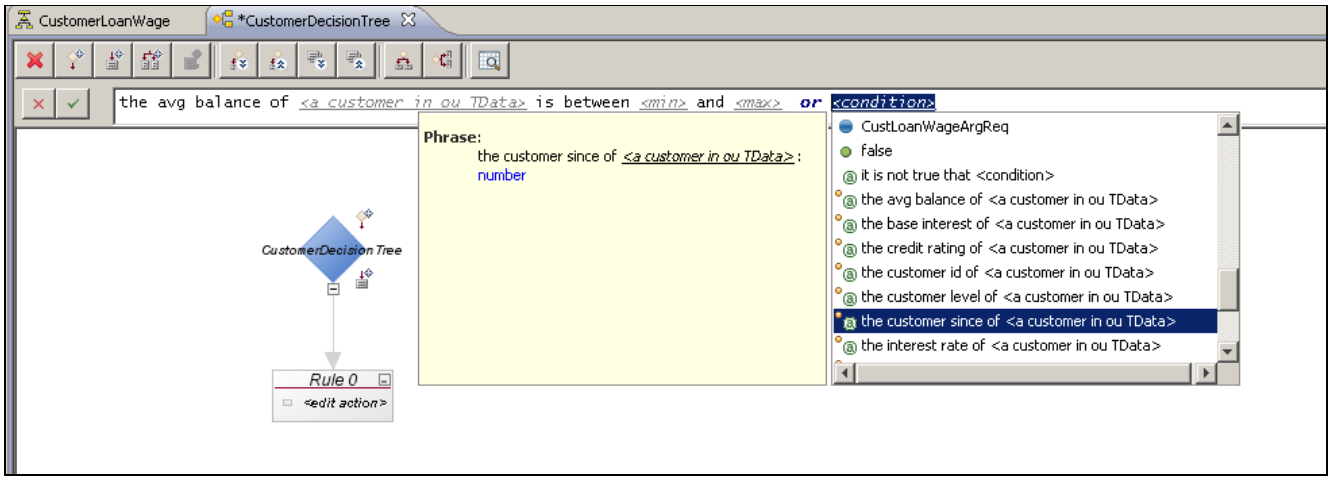


You will now see both parts of the condition statement added in the edit bar

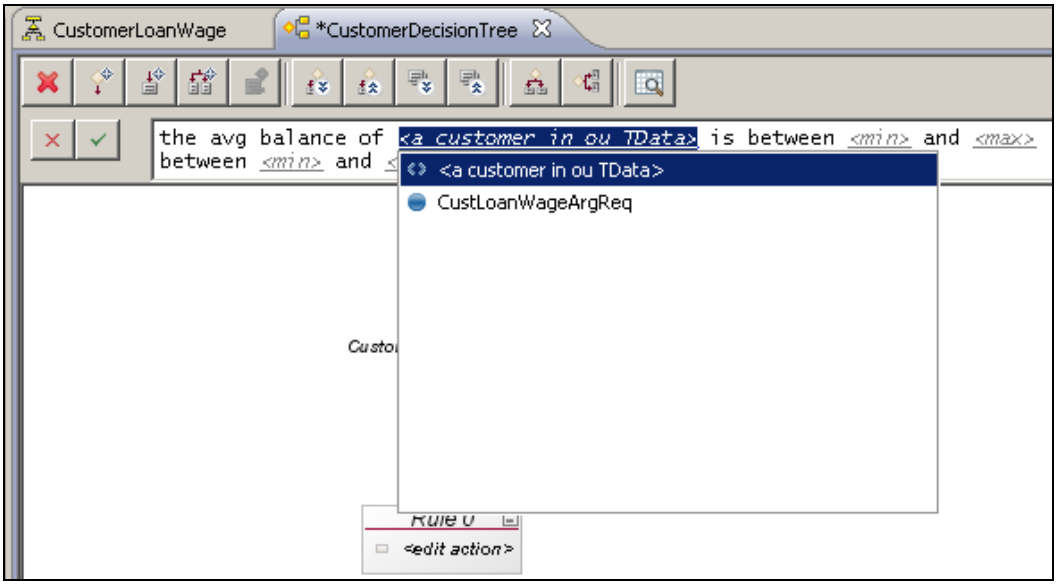
In the **Edit** bar, at the end of the condition statement, type or followed by a **Space**.

At the end of the condition statement, press CTRL + Space, and select from the drop - down list **the customer since of < a customer in ou T Data>** and double click**.**

Press **CTRL + Space** , and select from the drop-down **is between <min> and <max>** and double click.



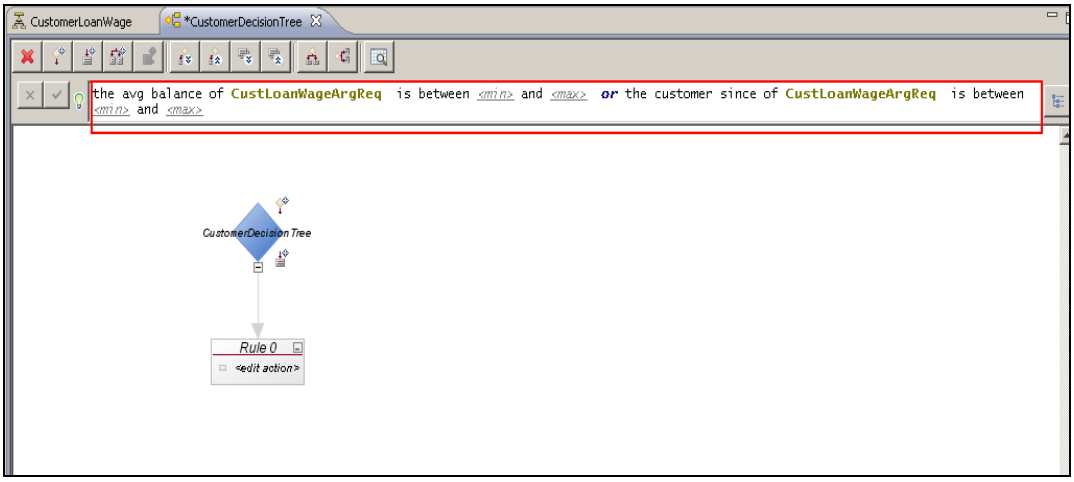
Now click on the first occurrence of **< a customer in ou T Data>** and select **CustLoanWageArgReq** from the pop-up menu.



Double -click

Repeat the same procedure for the second occrrence of **< a customer in ou T Data>**

Finally , the condition looks like this.



To save the definition, click the **Check mark** buton to the left of the edit bar.

The condition node is now defined.

Insert a new branch.

In the Decision Tree Editor , right click the condition node, and cick Add-> New -> Branch.

Add one more branch

Insert an Otherwise branch

In the decision tree editor, right click the condition node and click Add--> Otherwise.

Defining and labeling branches

To define a branch perform the following steps :

- In the Decision Tree Editor, select the left most first branch. After selecting it should turn blue.

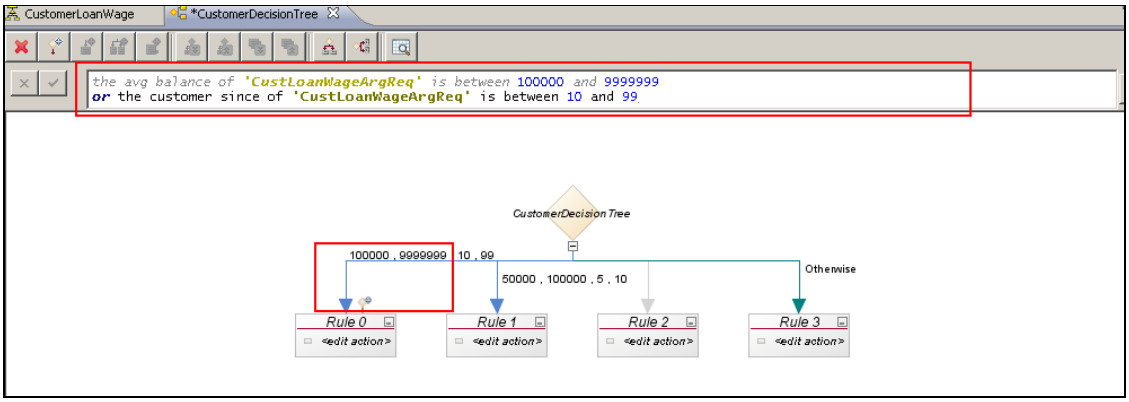
- In the edit bar, edit the operator and values of the condition statements. Click the first occurrence of **<a number>** and select from the drop-down list <number> and replace **0 with 100000**.

Now repeat this procedure for the other occurrences of **<a number>** and replace the values with **9999999, 10, and 99**

Click **Check mark** to save the definition

You obtain this code :

the avg balance of 'CustLoanWageArgReq' is between 100000 and 9999999 or the customer since of 'CustLoanWageArgReq' is between 10 and 99

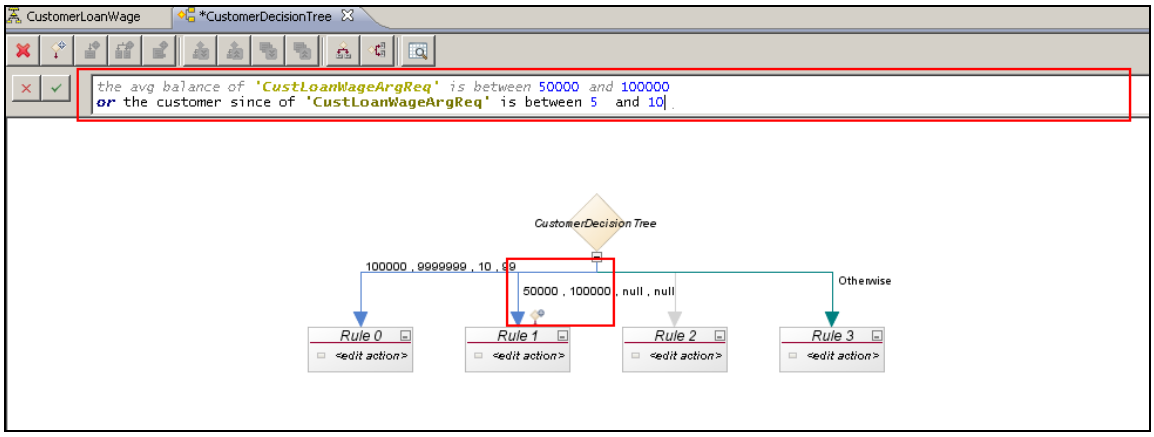


Repeat the above steps for defining the branch for the next two branches using the following data respectively :

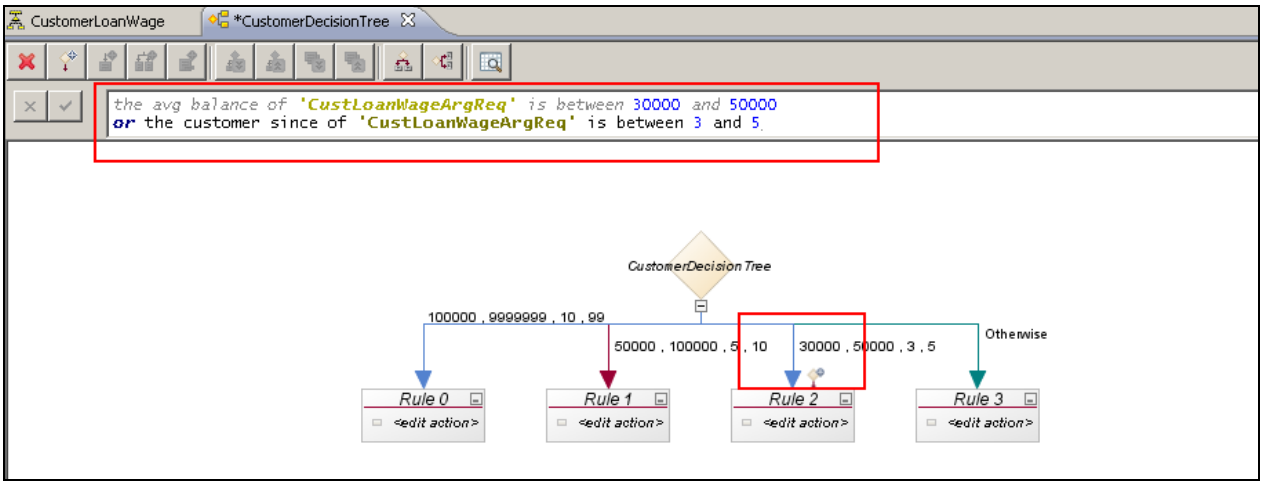
50000 ,100000, 5,10

30000, 50000, 3, 5

the avg balance of 'CustLoanWageArgReq' is between 50000 and 100000 or the customer since of 'CustLoanWageArgReq' is between 5 and 10



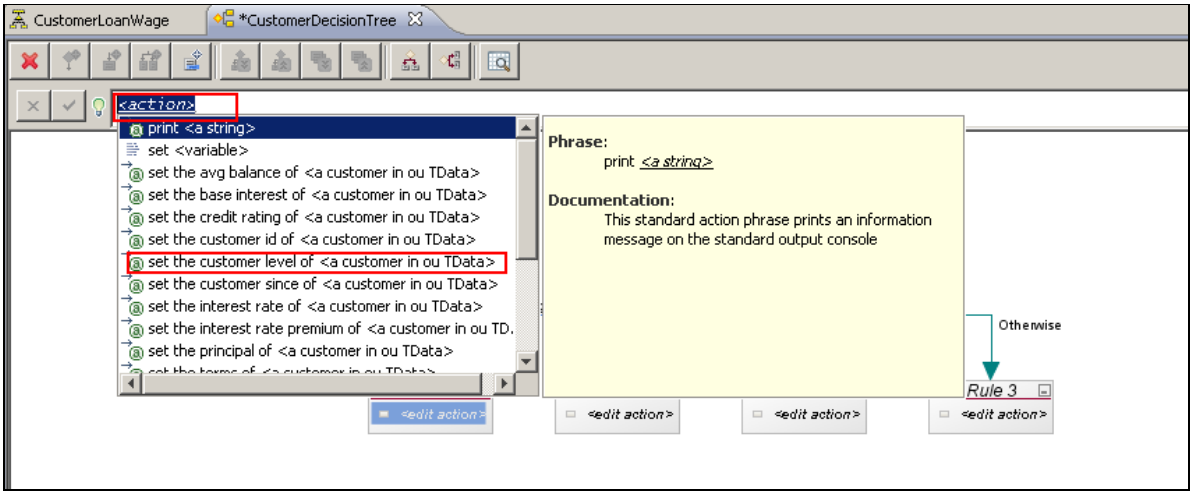
the avg balance of 'CustLoanWageArgReq' is between 30000 and 50000 or the customer since of 'CustLoanWageArgReq' is between 3 and 5



Labeling and defining actions

To define an action, perform the following steps :

- In the Decision Tree Editor, click <Edit action> of rule 0



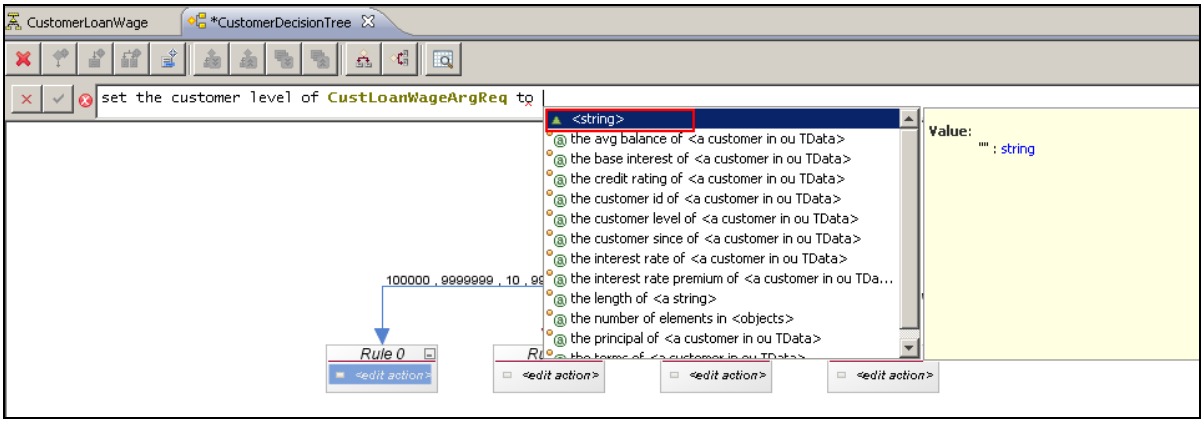
In the edit bar , click <**action**> to define the action

Select set the customer level of **< a customer in ou TData >** from the drop-down list and double click .

Press **CTRL + Space** and select **CustLoanWageArgReq** from the drop-down list and double-click.

At the end of the action, press **CTRL + Space** and select **to <a string>** from the drop-down list and double-click .

Press **CTRL + Space** and select **<string>**

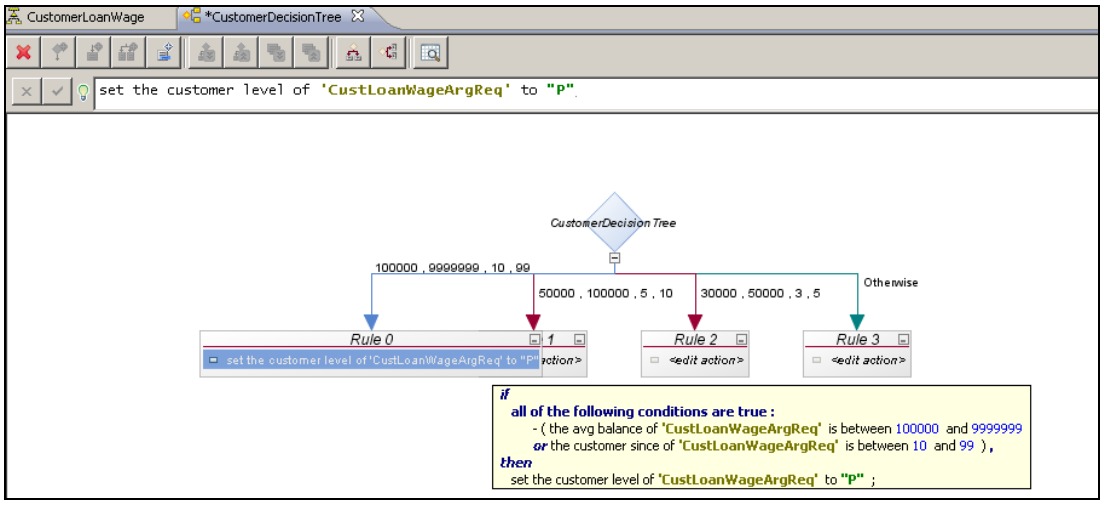


Replace "" with "P" in the edit bar.

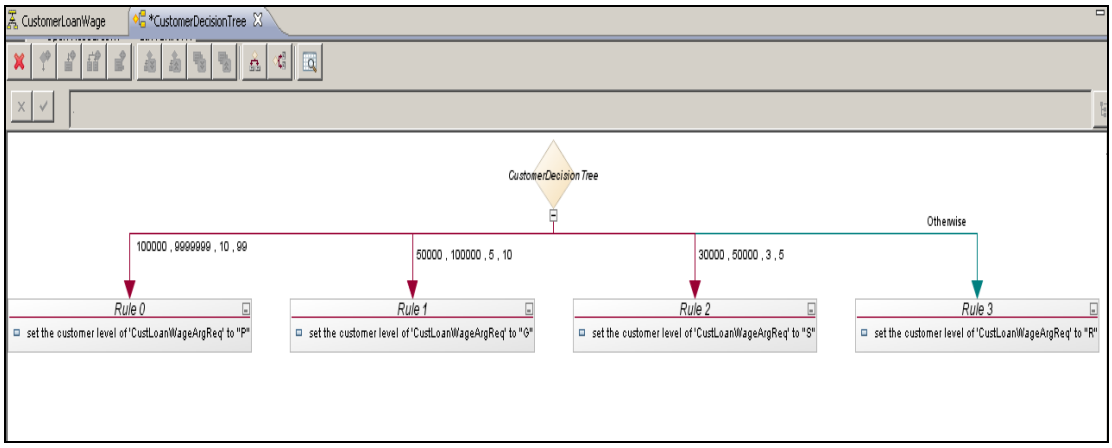
Click the **Check Mark** button

Repeat the above steps for defining an action for the other branches with jsut replacing the customer level string as follows:

- Rule 1 : set the customer level of **CustLoanWageArgReq** to "G"
- Rule 2 : set the customer level of **CustLoanWageArgReq** to "S"
- Rule 3 : set the customer level of **CustLoanWageArgReq** to "R"



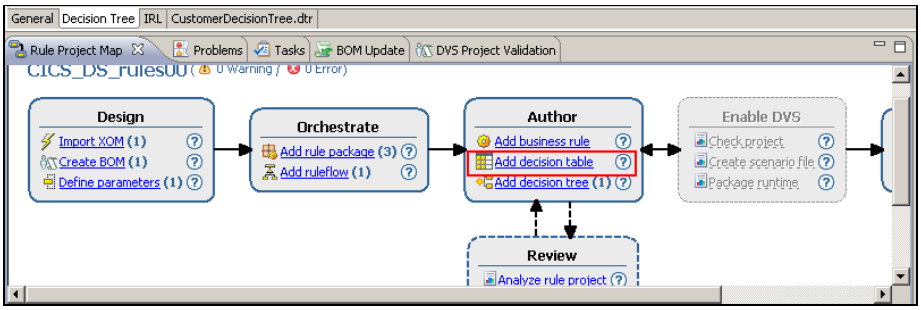
Now the Decision Tree should look like this :



Save your work and close the decision tree editor

___ 2. - Creating a business rule - SetInterestPremium using a Decision Table

For this business rule you will use the decision table technique to define the rule.



Creating a decision table.

Decision tables are rules comprising rows and columns. They are used to represent in tabular form all possible situations that a business decision may encounter, and to specify which action to take in each of these situations.

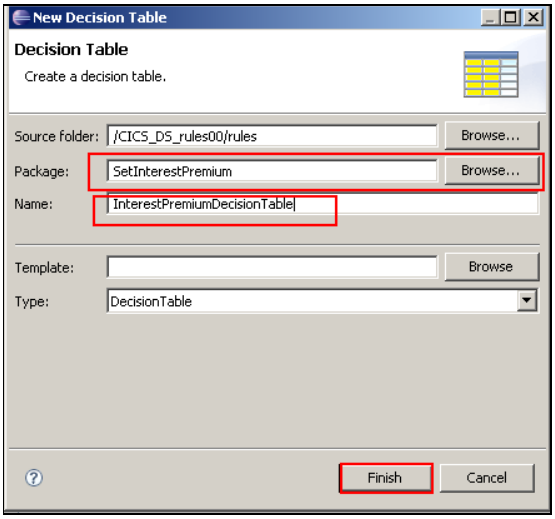
Perform the following steps :

- Select Rule project and in the file menu click New--> Decision Table

- Browse next to the Package field and navigate to Rule package *SetInterestPremium*. Click OK

- Type ***InterestPremiumDecisionTable*** as a name of the Decision table in the *Name* field.

- Click **Finish**

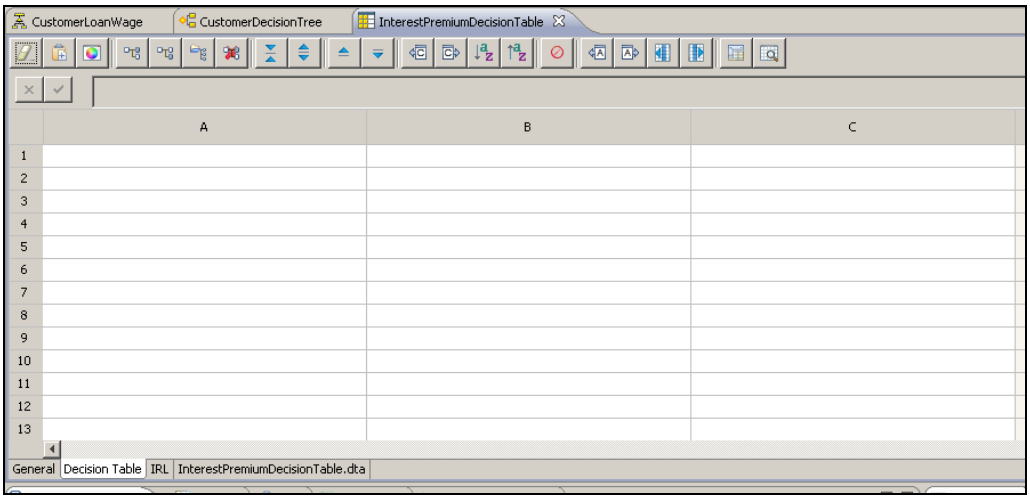


The new decision table appears in the Rule Explorer view and the Decision Table.

The default decision table has tree condition columns and one action column.

Action columns have a shaded background.

Defining a condition column using the Decision Table editor.



The new decision table appears in the Rule Explorer view and the Decision Table.

The default decision table has tree condition columns and one action column.

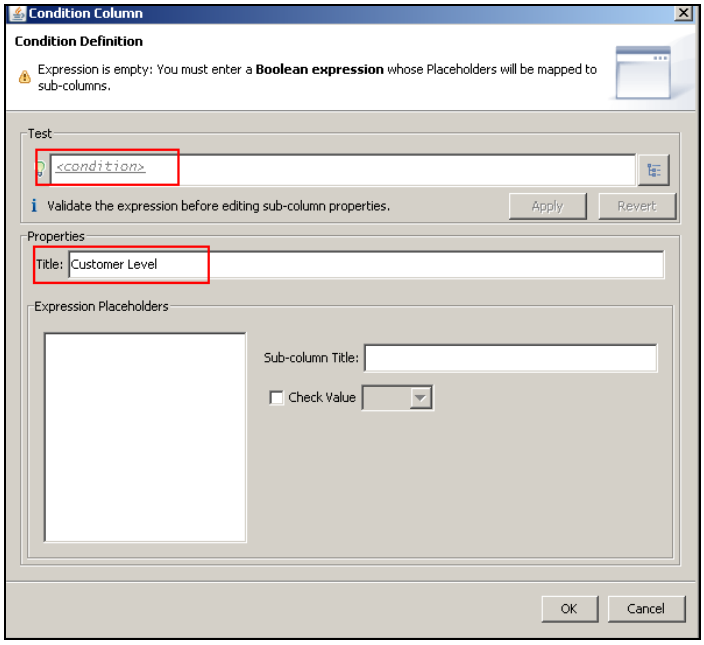
Action columns have a shaded background.

Defining a condition column using the Decision Table editor.

Perform the following steps:

- In the decision Table editor, double-click on the first **condition** column labeld **A**.

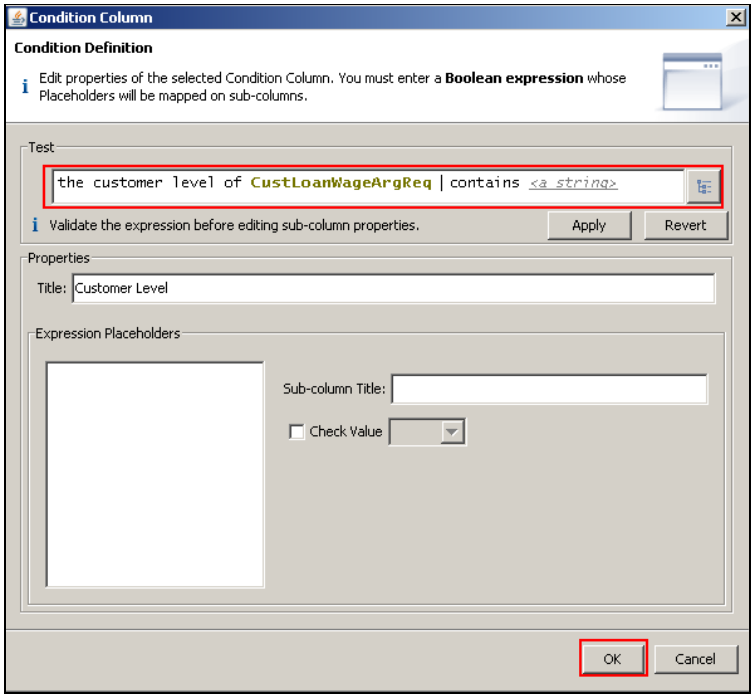
- In the Condition Definition dialog, type the **Customer Level** in the *Title* field.

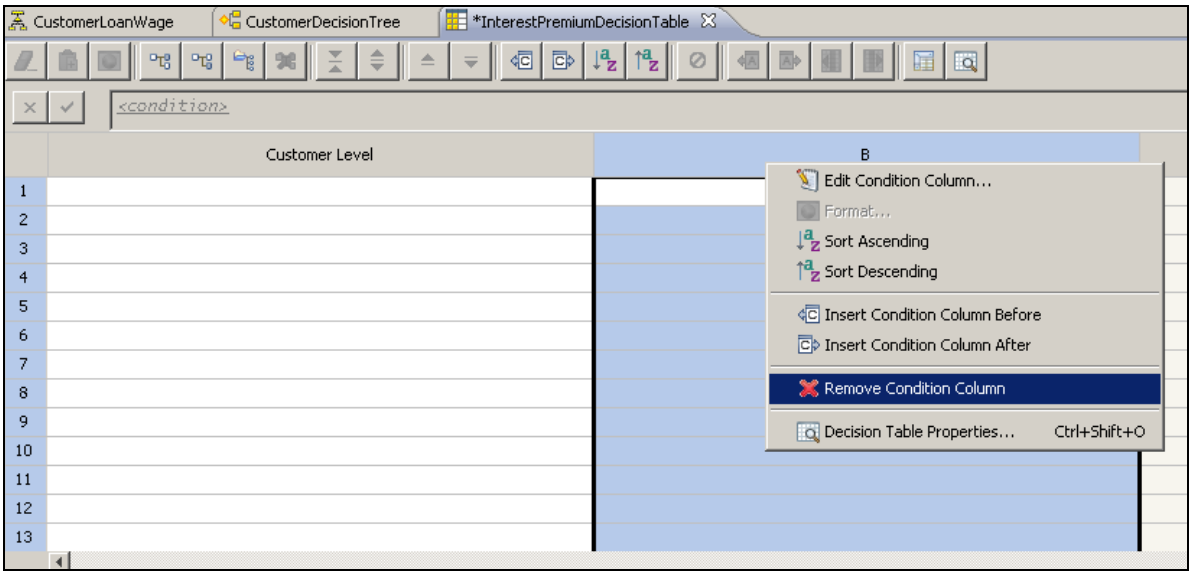


- Click <**condition**> and select the customer level of **< a customer in ou TData >** from the drop-down list and double-click.

- Press **Ctrl + Space** and select **contains < a string>** from the drop-down list and double-click.

Click on **< a customer in ou TData >** and select **CustLoanWageArgReq**, and double-click.





Removing columns

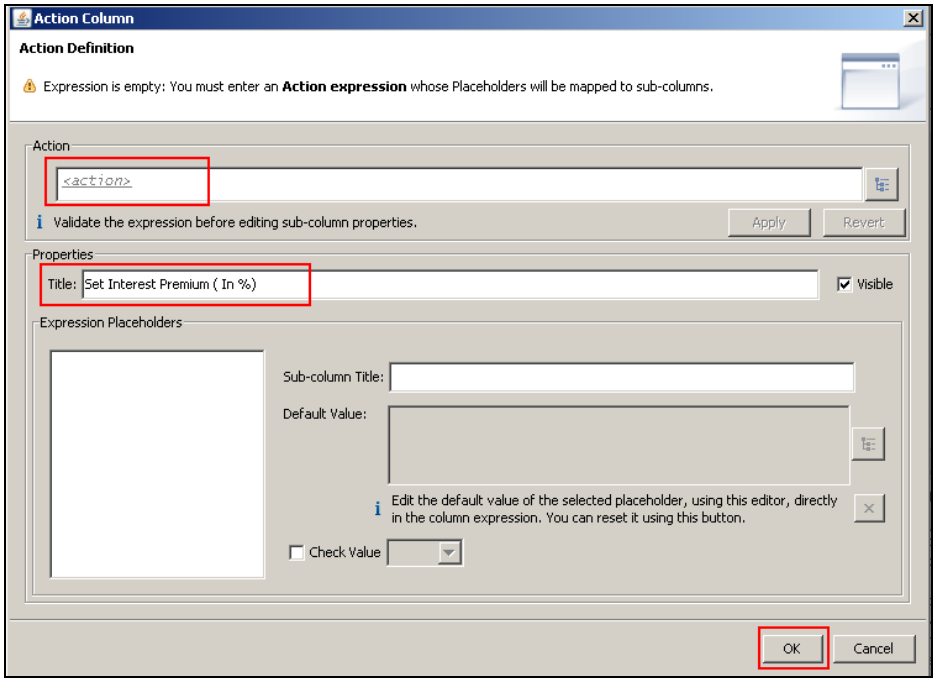
Condition or action columns can be removed from decision table.

You now need to remove the two other condition columns from the decision table.

- In the Decision Table editor , right click **column B**. Click **Remove Condition Column**.

- Again, right click **column B**. Click **Remove Condition Column**.

The column and any dependent cells are removed from the decision table



Defining action columns

Action columns are defined through the Decision Table editor.

To define an action column :

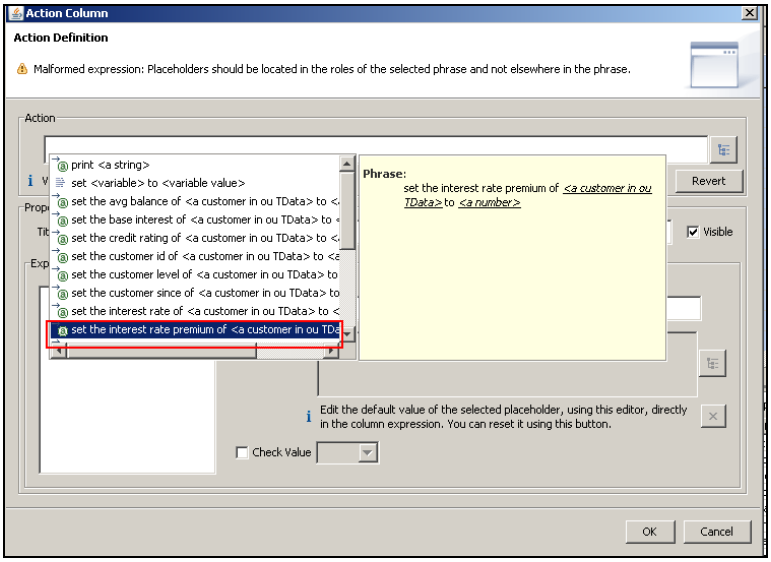
- In the Decision Table editor, double - click an **action** column (**column B**).

  The Action Definition pane opens

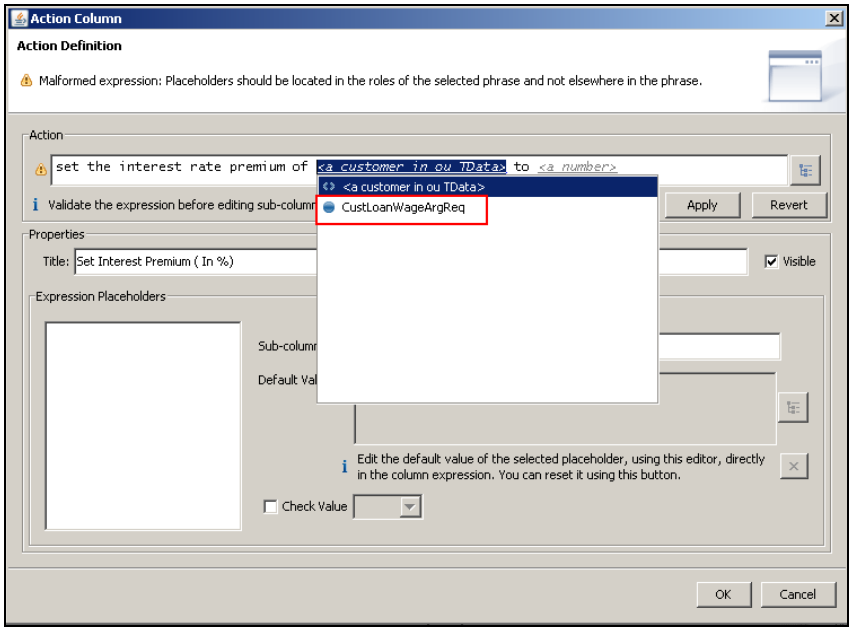
- In the Action definition dialog, type **Set Interest Premium (in %)** in the *Title* field.

- Click <**action**> and construct your action statement in the Action Expression Editor using the available elements from the vocabulary.

- Select from the drop-down list **set the interest rate premium of < a customer in ou TData >** to **< a number>** and double-click.



- click on of **< a customer in ou TData >** and select **CustLoanWageArgReq** and double-click.

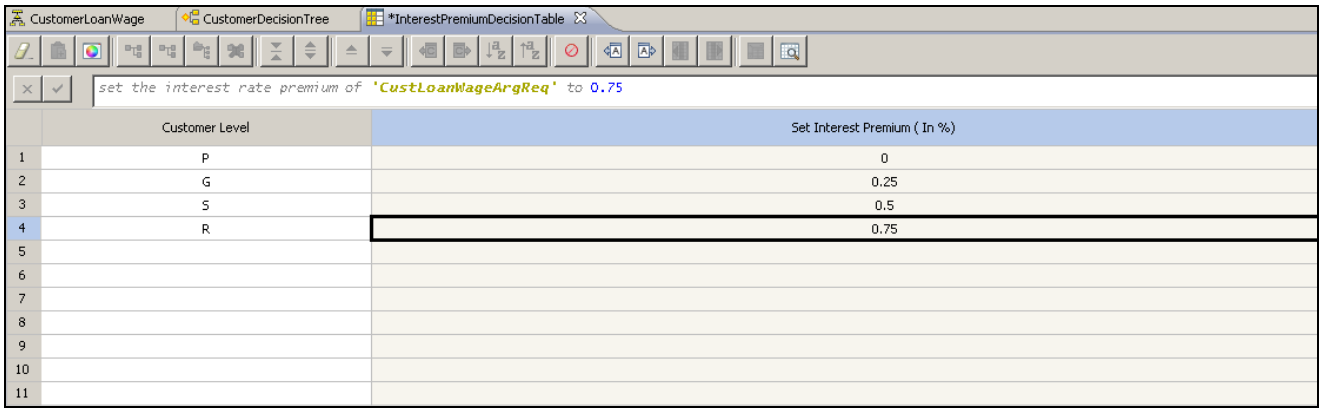


- Click **OK**.

The Action column is now set up.

Populating data for the table

- Insert the following data into the table columns.



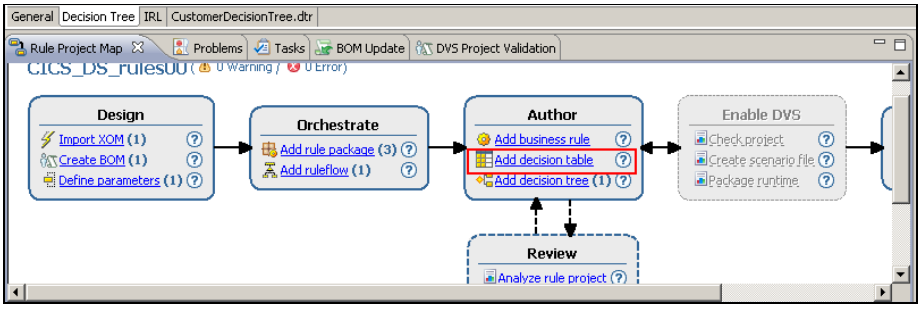
	Customer Level	Set Interest Premium (In %)
1	P	0
2	G	0.25
3	S	0.5
4	R	0.75
5		
6		
7		
8		
9		
10		
11		

Now the **InterestPremiumDecsionTable** decision table looks as shown in the figure above.

- Save your work and close the Decision Table Editor.

___ 3. - Creating a business rule - SetInterestRate using a Decision Table

Again, for this business rule you will use the decision table technique.



Creating a decision table.

Decision tables are rules comprising rows and columns. They are used to represent in tabular form all possible situations that a business decision may encounter, and to specify which action to take in each of these situations.

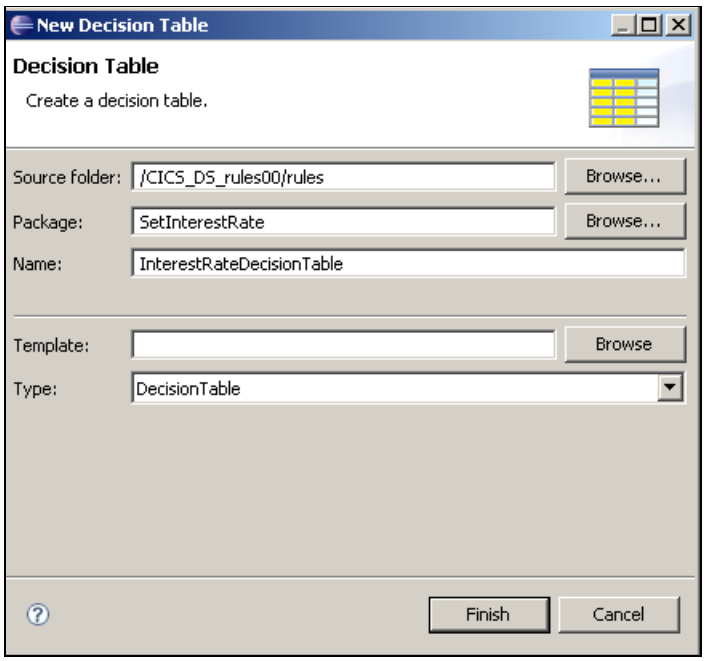
Perform the following steps :

- Select Rule project and in the file menu click New--> Decision Table

- Browse next to the Package field and navigate to Rule package *SetInterestRate*. Click OK

- Type ***InterestRateDecisionTable*** as a name of the Decision table in the *Name* field.

- Click **Finish**



The new decision table appears in the Rule Explorer view and the Decision Table.

The default decision table has tree condition columns and one action column.

Action columns have a shaded background.

Defining columns.

A decision table must have at least one condition column and one action column. You can define condition and action columns using the decision table editor. if a condition statement has two placeholders, the required sub columns are automatically created.

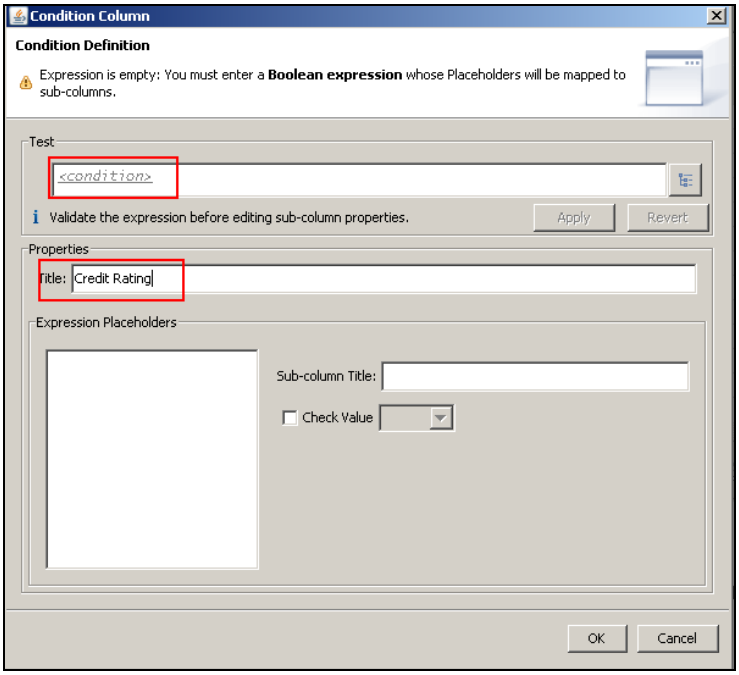
Once you have defined at leat one condition column and an action column, you can begin to specify values for the placeholders in the condition and action definitions.

TO define a condition column using the decision table Editor :

- In the decision Table editor, double-click on the **A Condition** column.

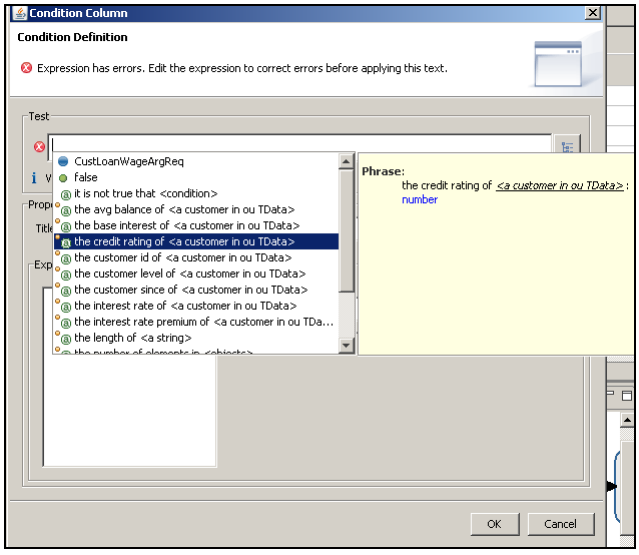
The column Definition pane opens.

- In the Condition Definition dialog, type **Credit Rating** in the Title field.



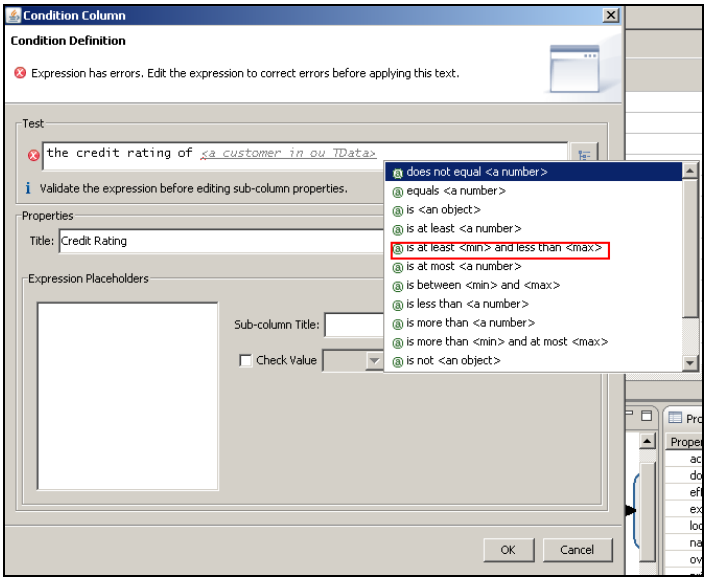
- Click < condition > and select the credit rating of a **CustLoanWageArgReq is at least <min> and less than <max>**

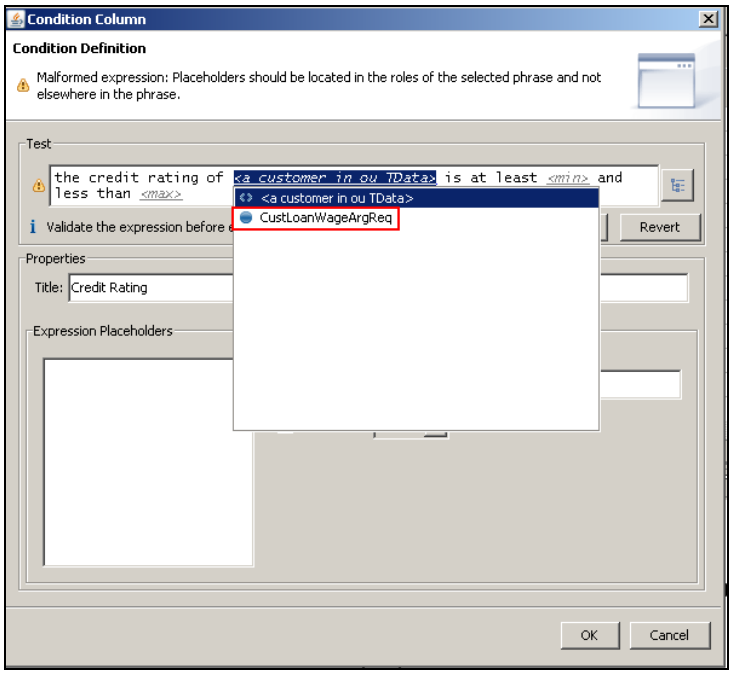
- Click **OK**



.

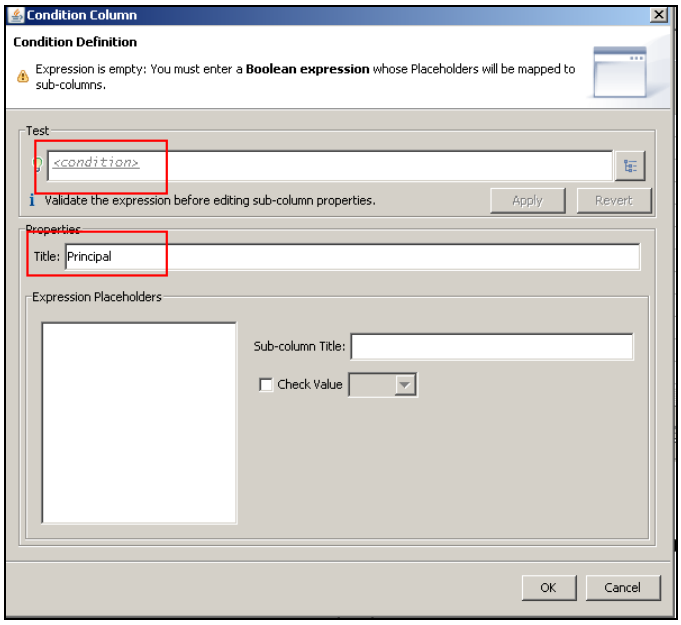
-


Condition Column
Condition Definition
Expression has errors. Edit the expression to correct errors before applying this text.
Test
the credit rating of <a customer in ou TData>
Validate the expression before editing sub-column properties.
Properties
Title: Credit Rating
Expression Placeholders
Sub-column Title:
Check Value
does not equal <a number>
equals <a number>
is <an object>
is at least <a number>
is at least <min> and less than <max>
is at most <a number>
is between <min> and <max>
is less than <a number>
is more than <a number>
is more than <min> and at most <max>
is not <an object>
OK
Cancel


Condition Column
Condition Definition
Malformed expression: Placeholders should be located in the roles of the selected phrase and not elsewhere in the phrase.
Test
the credit rating of <a customer in ou TData> is at least <min> and less than <max>
<a customer in ou TData>
CustLoanWageArgReq
Validate the expression before
Revert
Properties
Title: Credit Rating
Expression Placeholders
OK
Cancel

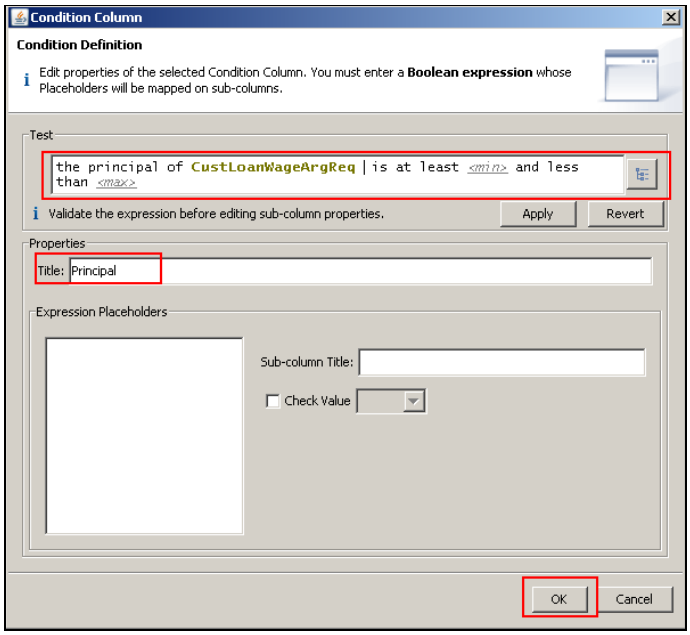
- Now double-click on the **B condition** column dialog

Again, the Column Definition pane opens



- In the Condition Definition dialog, type **Principal** in the *Title* field

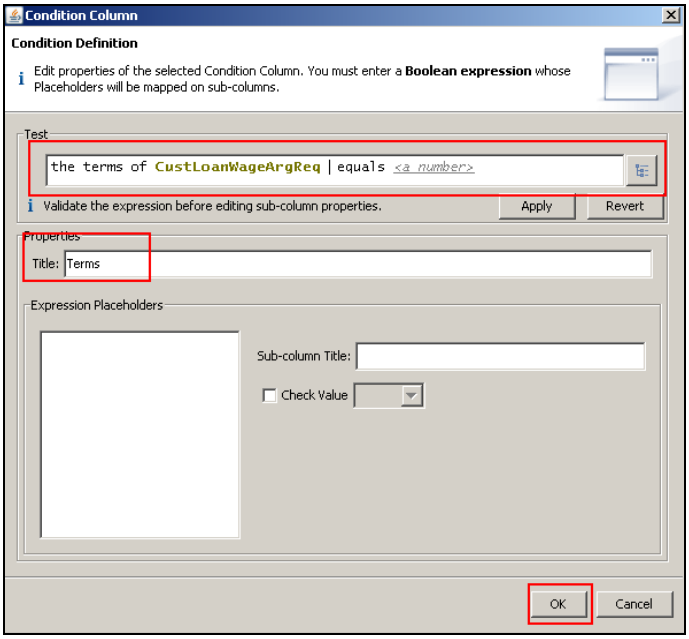
- Click <**condition**> and select the principal of CustLoanWageArgReq is at least <min> and less than <max>.



Click **OK**.

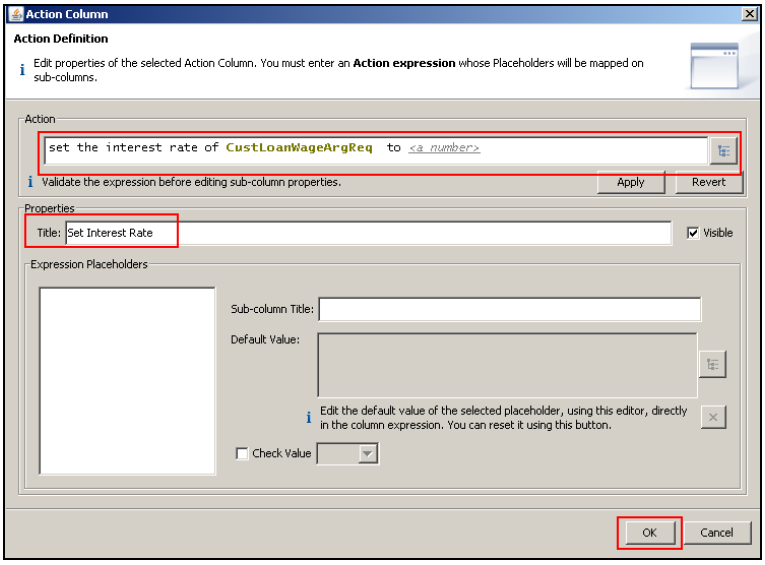
- In the Decision Table Editor, double-click on the C **Condition** column.

- In the Condition definition dialog, type **Terms** in the *Title* field.



- Click <**condition**> and select the terms of **CustLoanWageArgReq equals < a number>.**

- Click **OK**.



Defining action columns

Action columns are defined through the Decision Table Editor.

To define an action column :

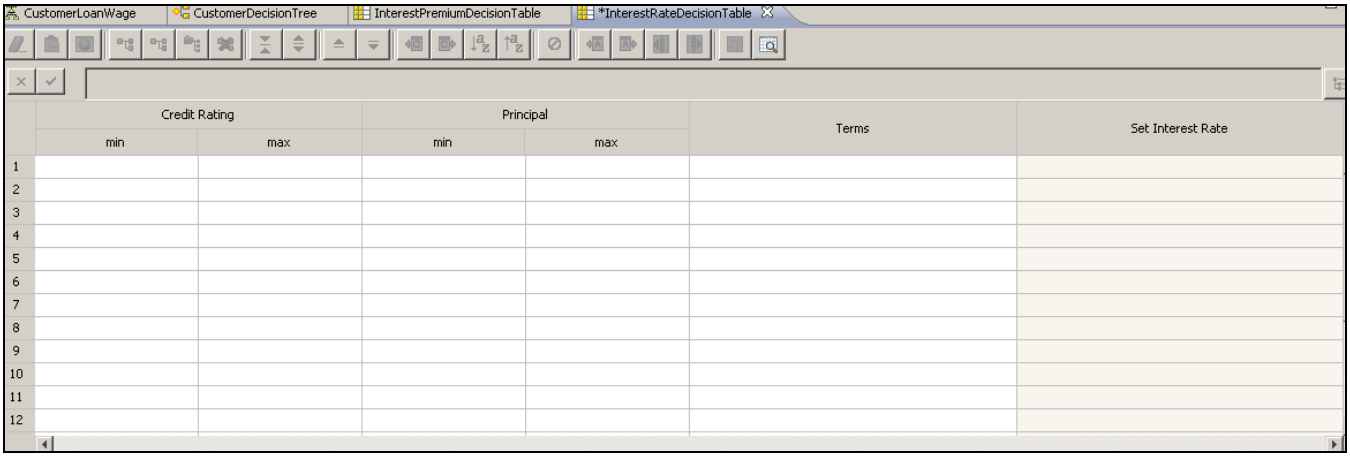
- In the Decision Table Editor, double - click on the D **action** column dialog

- In the Action Definition dialog, type **Set Interest Rate** in the *Title* field.

- Click <action> and select **set the intresest rate of CustLoanWageArgReq to <a number>.**

- Click **OK**.

Now the **InterestRateDecisionTable** decision table looks like this :



Populate Data for table.

Perform the following steps :

- In the decision table editor, select first the column Credit Rating.

- Insert data in the min and max columns as show in figure below.

	Credit Rating	
	min	max
1	540	700
2		
3		
4		
5		
6		
7	700	830
8		
9		
10		
11		
12		

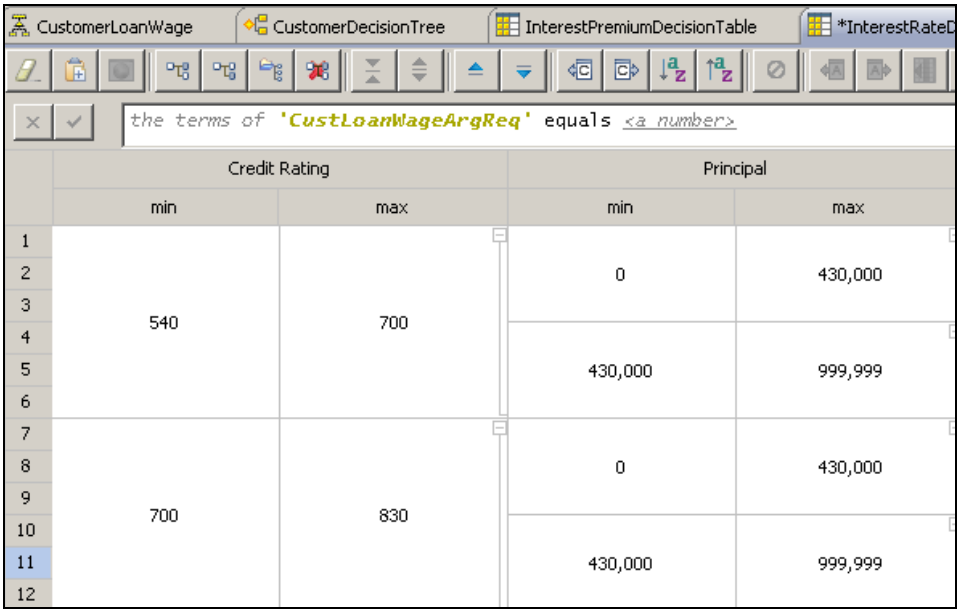
- Click the second condition column **Principal**

- Right click a first row cell in the **Principal** Column.

- Right click and in the pop-pup menu clcik **Add--> Insert -New Row Before or Insert New Row After**

- Add one more row after

- Insert data in the min and max columns as shown in the figure below:



	Credit Rating		Principal	
	min	max	min	max
1	540	700	0	430,000
2				
3				
4			430,000	999,999
5				
6				
7	700	830	0	430,000
8				
9				
10			430,000	999,999
11				
12				

- Click on the third condition column **Terms**.

- Right-Click a first row cell in the **Terms** column.

- Add 2 rows by using **Add--> Insert -New Row Before or Insert New Row After**

- Right-Click a fourth row cell in the **Terms** column.

- Add 2 rows by using **Add--> Insert -New Row Before or Insert New Row After**

- Right-Click a seventh row cell in the **Terms** column.

- Add 2 rows by using **Add--> Insert -New Row Before or Insert New Row After**

- Right-Click a tenth row cell in the **Terms** column.

- Add 2 rows by using **Add--> Insert -New Row Before or Insert New Row After**

- Insert data in the columns as shown in the figure below:

	Credit Rating		Principal		Terms
	min	max	min	max	
1	540	700	0	430,000	15
2					20
3					30
4			430,000	999,999	15
5					20
6					30
7	700	830	0	430,000	15
8					20
9					30
10			430,000	999,999	15
11					20
12					30

- Click the fourth action column **Set Interest Rate**

- Insert data in the min and max columns as shown in the figure below:

	Credit Rating		Principal		Terms	Set Interest Rate
	min	max	min	max		
1	540	700	0	430,000	15	1.9
2					20	2
3					30	2.5
4			430,000	999,999	15	1.95
5					20	2.05
6					30	3
7	700	830	0	430,000	15	1
8					20	1.5
9					30	1.8
10			430,000	999,999	15	1
11					20	1.5
12					30	1.8

Now the **InterestRateDecisionTable** decision table looks like the figure above.

- Save your work and close the decision table editor.

You have now defined all business rules in the decision tree and decision tables form.

Lab 3 Deploying the Business Rule

3.1 Introduction

In this lab, you will work with ILOG Rule Studio to develop a business rule application that you will then deploy to the Rule Execution Server on z/OS. You will use various functions of ILOG Rule Studio.

The tasks you will execute are :

- Deployment to Rule Execution Server
- Testing
- Publishing to an exploring in Rule Team Server

3.2 Deployment to Rule Execution Server

In this task, you deploy your ruleset to Rule Execution Server on z/OS.

The *RuleApp* is the format expected by Rule Execution Server. It contains the ruleset. In the same way that java classes are packaged in Jar file and contains all that is necessary for execution (rules, ruleflow, and so on).

You will perform the following steps :

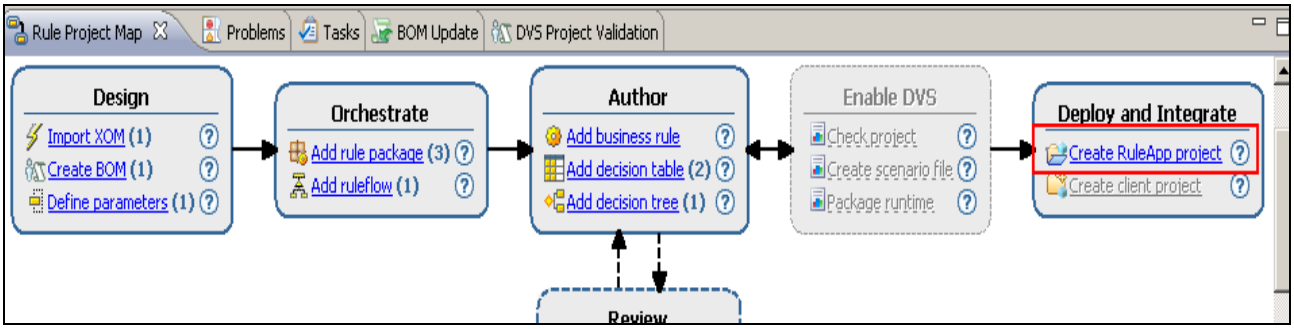
- Deploying from Rule Studio
- Viewing the deployed RuleApp
- Generating a Web Service fo ILOG JRules.

3.2.1 Deploying from Rule Studio

Creating a RuleApp project and deploy the RuleApp from Rule Studio

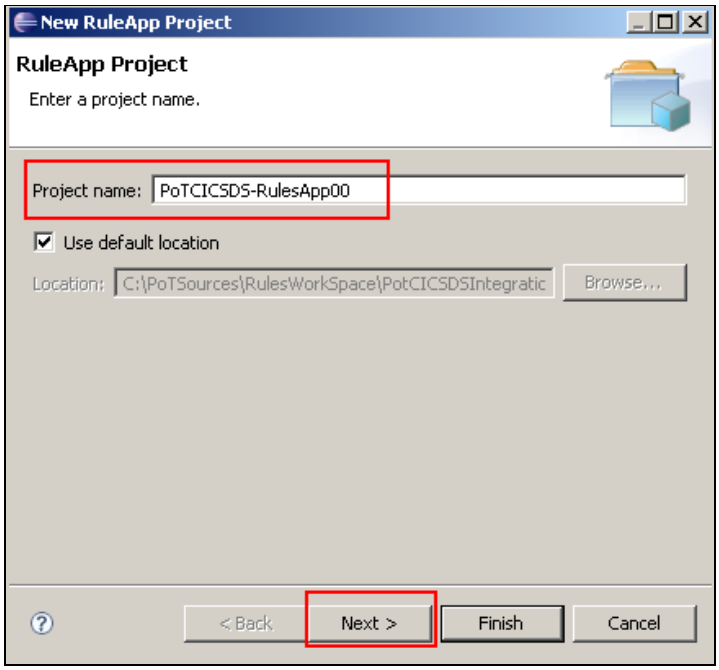
Perform the following steps .

- In Rule Studio, in the Deploy and Integrate part of the Rule project Map, click **Create RuleApp project**

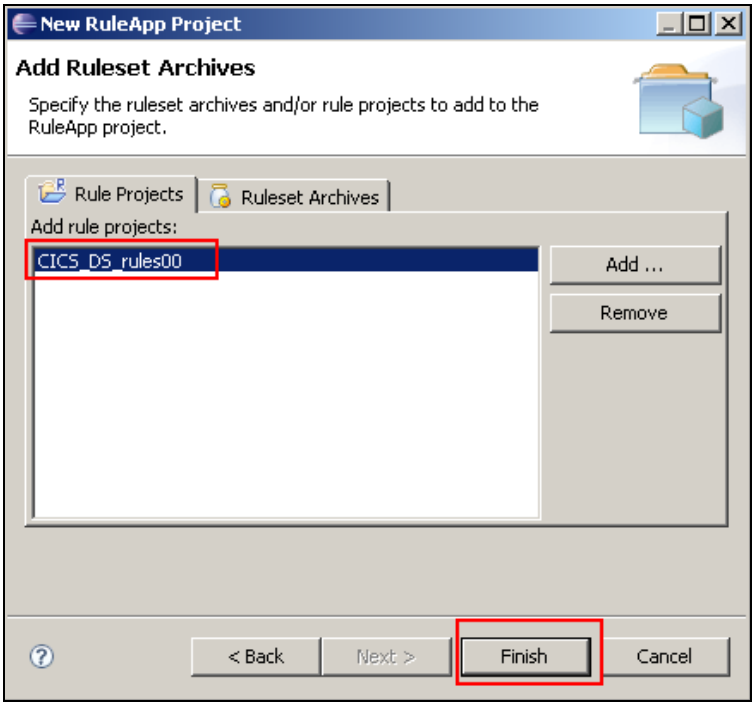


- In the New RuleApp Project Wizard, in the Project name field, type **PoTCICSDS-RuleAppxx** (where you substitute "**xx**" with your team number)

- Click **Next**



The CICS-DS_rulesxx (where you substitute "**xx**" with your team number) Rule project is displayed on the Add RuleSet Archive page, meaning that you will be creating a ruleset archive from this project as showed in the figure below.

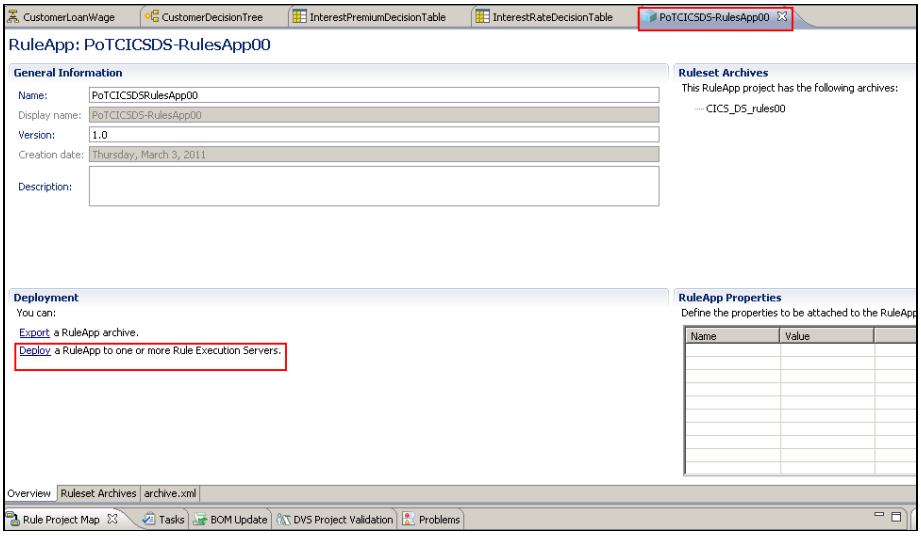


If it is not displayed, click **Add** and select it.

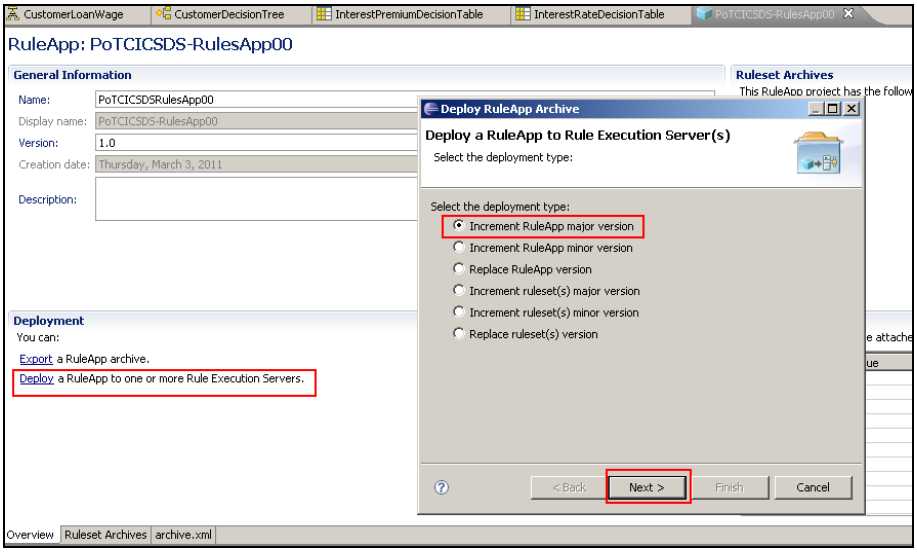
- Click **Finish** .

The RuleApp Editor opens

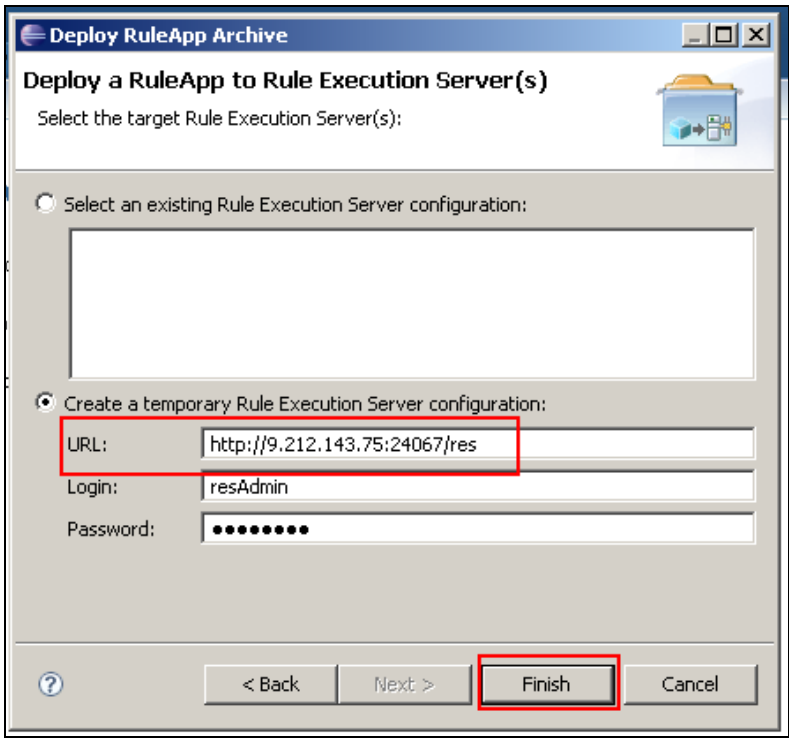
- In the Deployment section, click **Deploy**



- in the Deploy RuleApp to Rule Execution Server(s) wizard, keep Increment RuleApp major version selected, and click Next.



- On the next wizard page, make sure that Create a temporary Rule Execution Server Configuration is selected, and enter the following configuration details :



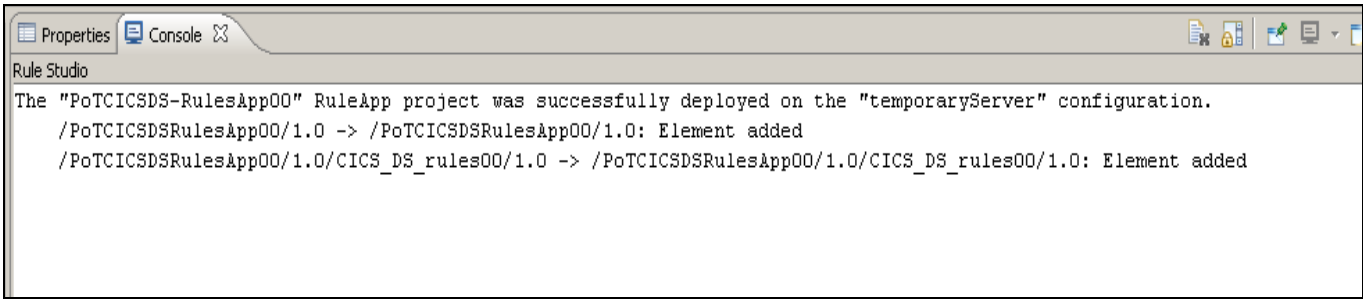
Set the URL of RES : http://9.212.143.75:24067/res

RES Admin : resAdmin

and RES password : resAdmin

- Click Finish.

The Console displays the following message, indicating that the 1.0 version of the RuleApp has been deployed.



3.2.2 Viewing the deployed RuleApp

You will now view the deployed RuleApp in the Rule Execution Server, which is an execution environment for Rules (Java SE and Java EE) that interacts with the rule engine.

Rule Execution Server handles the management, performance, security and logging capabilities associated with the execution of your rules.

From your application, you access Rule Execution Server using either Web Services, Enterprise JavaBeans (EJB), or pure Java objects (POJO).

Viewing the deployed RuleApp.

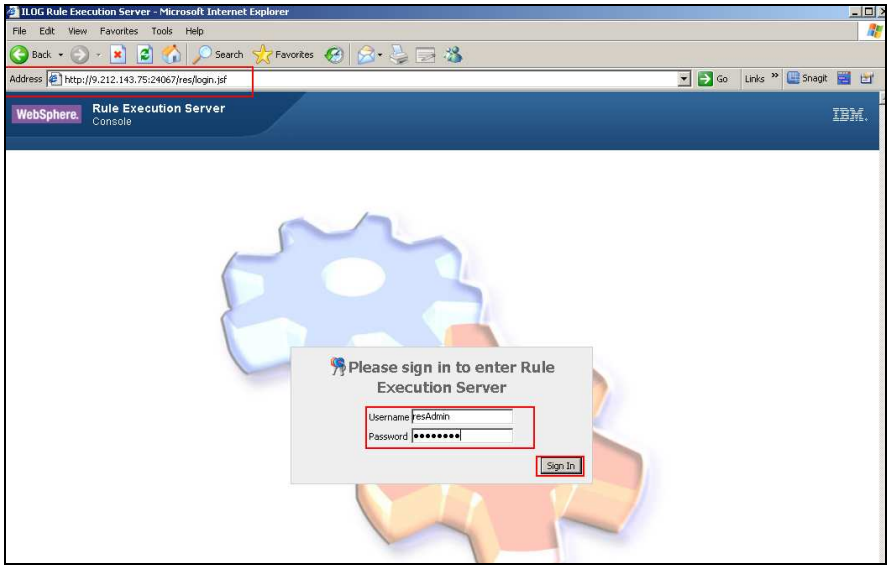
Perform the following steps

- In a browser, access the server at URL *http://9.212.143.75:24067/res*

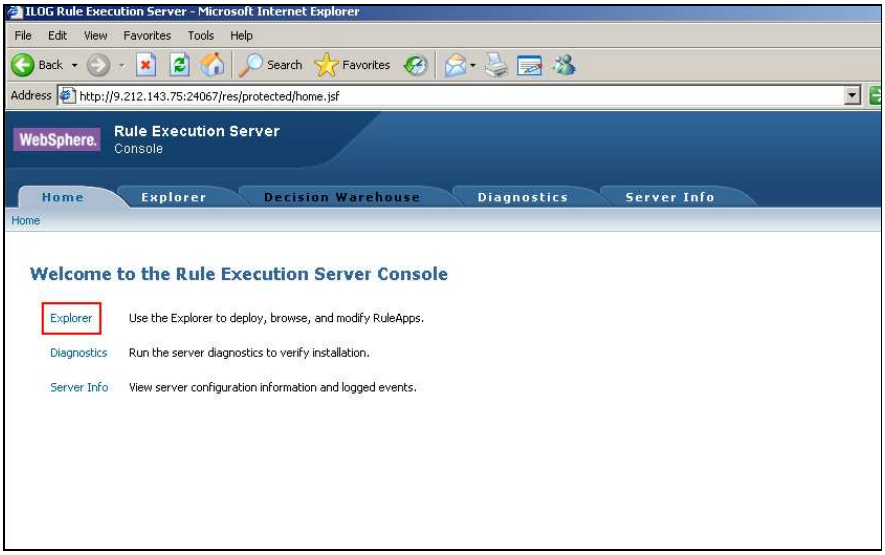
- Sign in to the Rule Execution Console using the following details :

Username : < Rule Execution Server Userid>

Password : < Rule Execution Server Password>

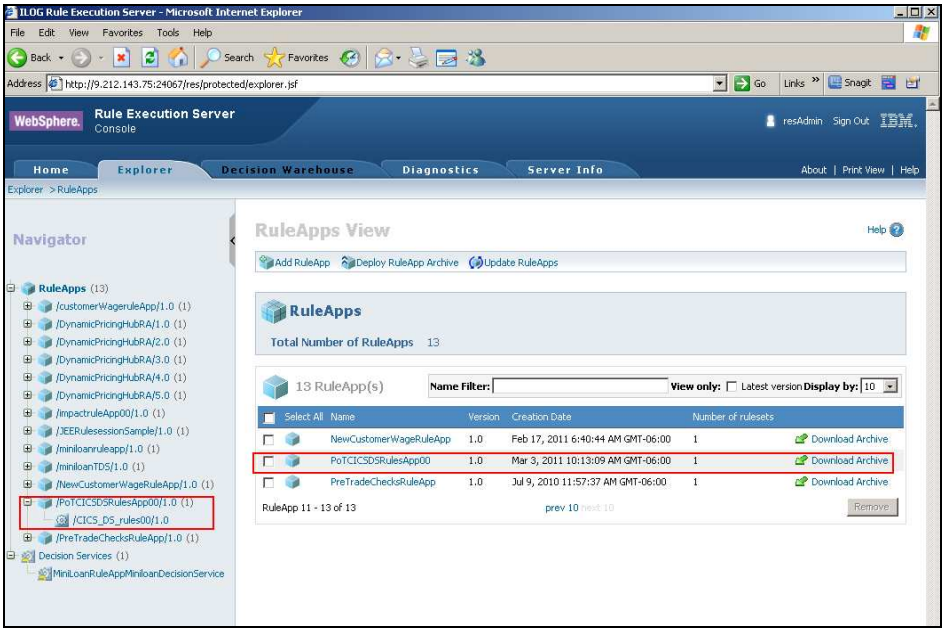


- click the **Explorer** tab

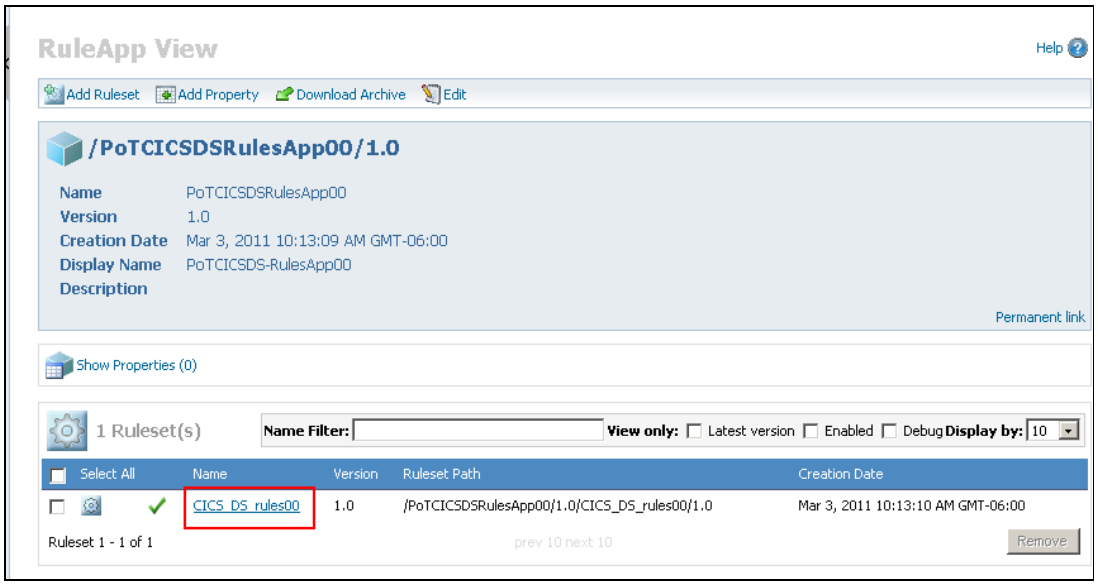


- Expand **RuleApp** in the **Navigator**

You see that Rule Execution Server contains your 1.0 version of **PoTCICSDSRuleAppxx** which contains the 1.0 version of the CICS-DS_rulesxx ruleset as expected.



- Click on **/PoTCICSDSRulesAppxx/1.0** where "xx is your team number.

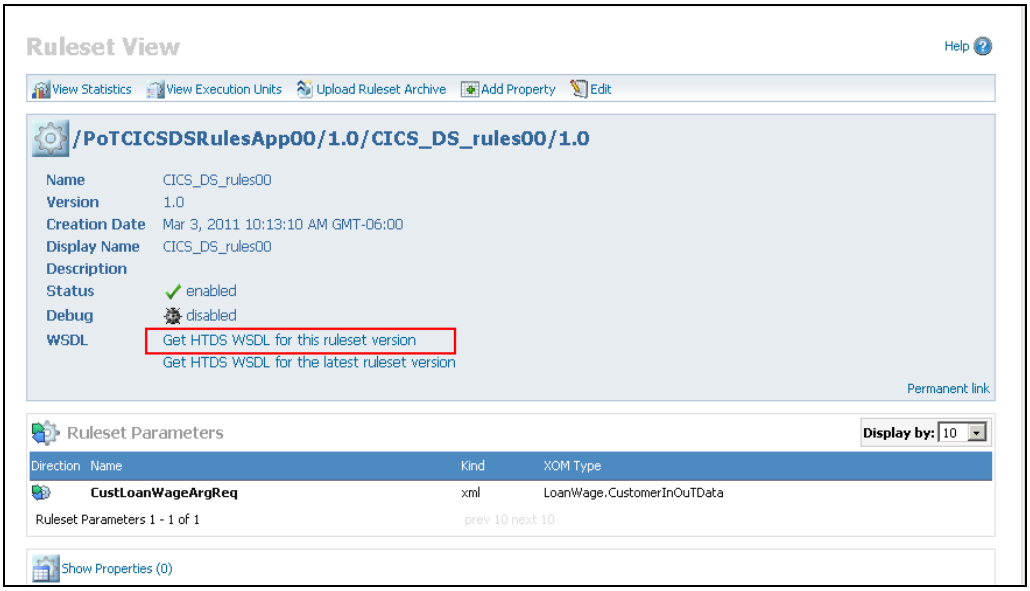


You have now deployed your ruleset to Rule execution Server on z/OS.

3.2.3 Generating a Web Service for ILOG JRules

To be able to access the Rule application, you can generate a Web Service as follows :

- In the RuleApp view, click on CICS_DS_rulexx, where "xx" is your team number.

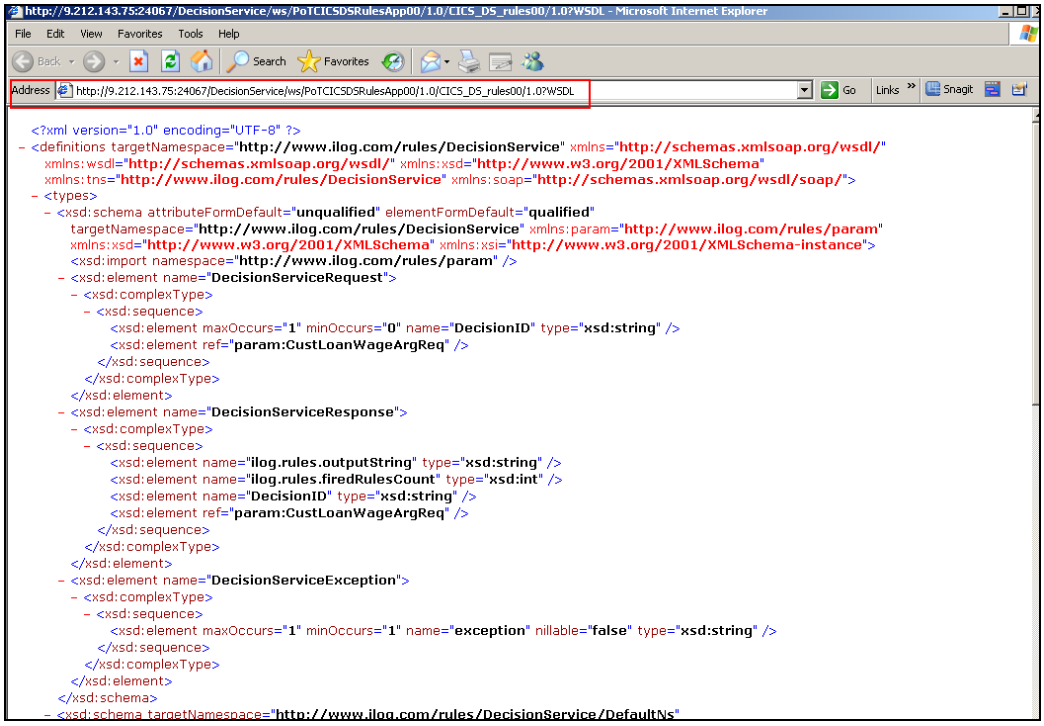


Notice that the status of the ruleset is **enable**, indicating that it can be executed.

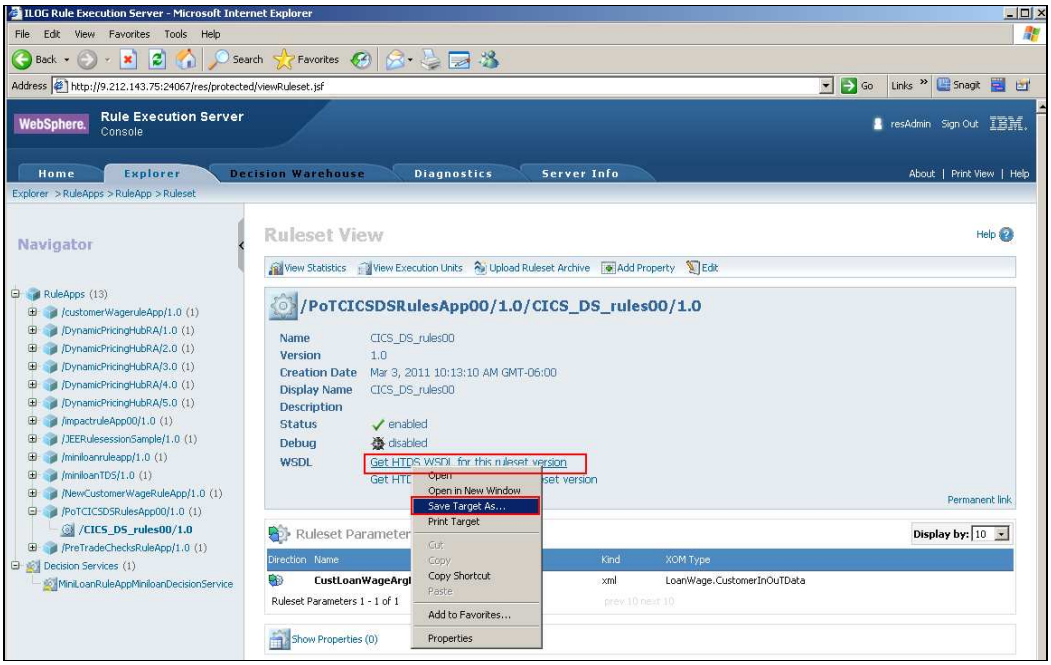
- Click on **Get HTDS WSDL for this ruleset version**

This generated a Web service for the RuleApp you created.

- A new browser windows opens displaying the WSDL for this Rules application, as show in figure below :



- Save the WSDL to ***c:\PoTsource\exports\CICS_DS_Rulesxx.wsdl*** *(Where **xx** is your team number)*



- Close the WSDL browser windows.

- Sign out ot the Rule Execution Server Console

3.2.4 Testing

WSDL documents allow developers to expose their applications as network-accessible services on the Internet. Through UDDI and WSDL, others applications can discover WSDL documents and bind with them to execute transactions or perform other business processes.

The Web Services Explorer allows you to explore, import, and test WSDL documents.

Pre requesites for using the WSDL Explorer are :

- You must have a valid URL for WSDL document
- You must launch the Web Services Explorer first.

Opening the prosper Perspective

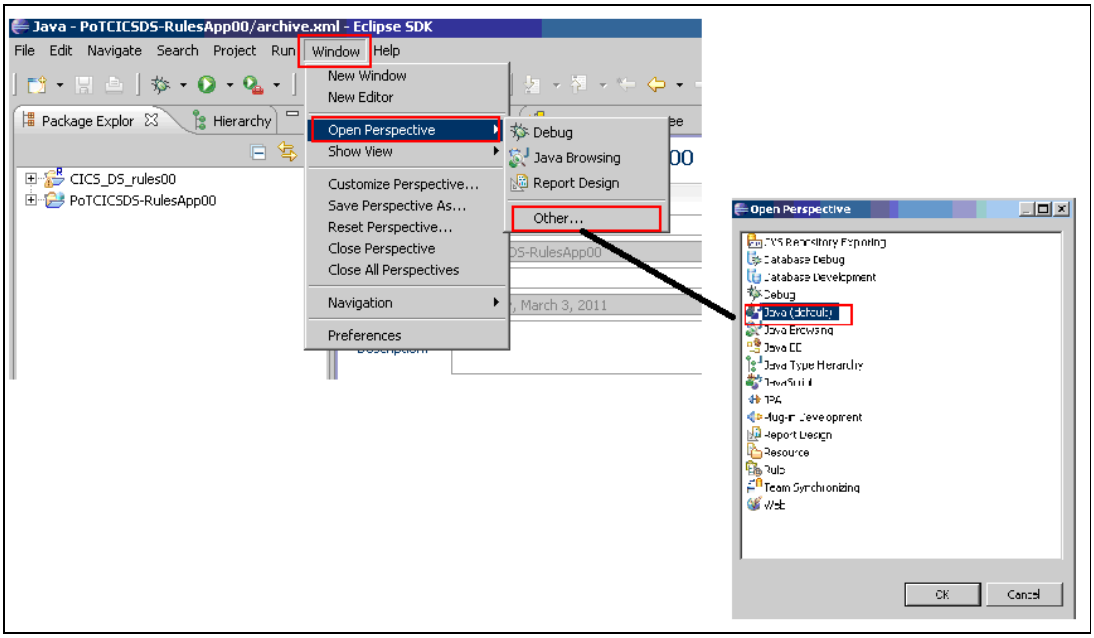
Perform the following steps

- Select **Windows --> Open Perspective --> Others...**

- Select **Java** (default)

- The Java Perspective opens with two projects from JRules

This would suffice to open Web Services Explorer and do the testing.



Importing a WSDL

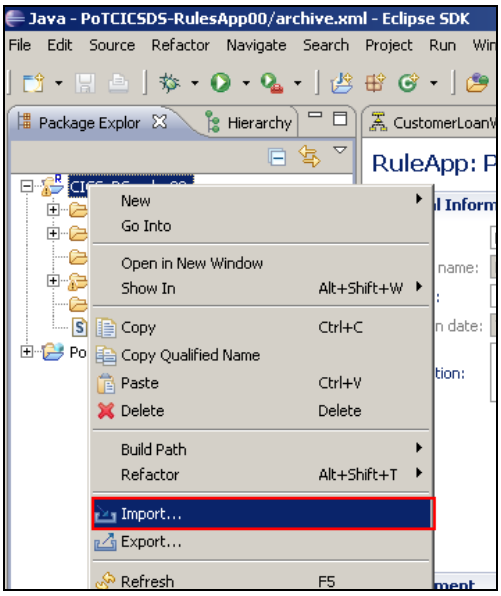
You can import WSDL files into the workbench and open them in the WSDL editor. Opening them in the WSDL editor provides you with an opportunity to have a structured viewing of the WSDL file.

To import a WSDL file to your project, complete, complete the following steps :

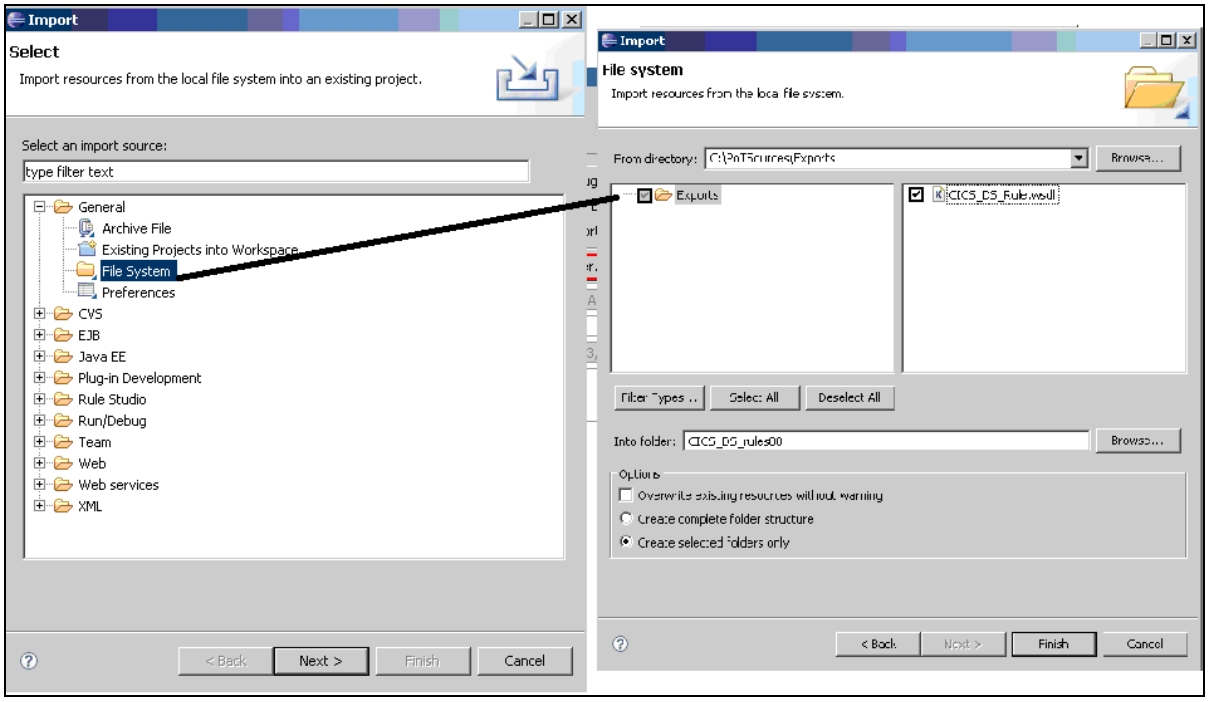
- Select Rule project in the **Package Explorer**, and from the **File** menu, select **Import**.

- Select **General --> File System** and click **Next**.

- Click **Browse ...** on the next page of the wizard to select the directories from which you would like to add WSDL. Navigate to ***c:\PoTsource\exports***



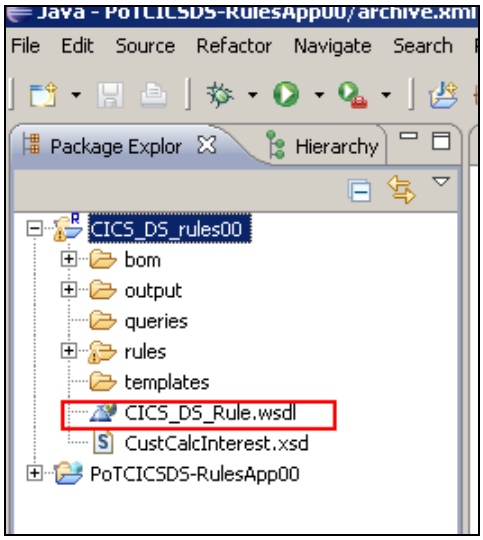
- Select only **CICS_DS_Rulesxx.wsdl**



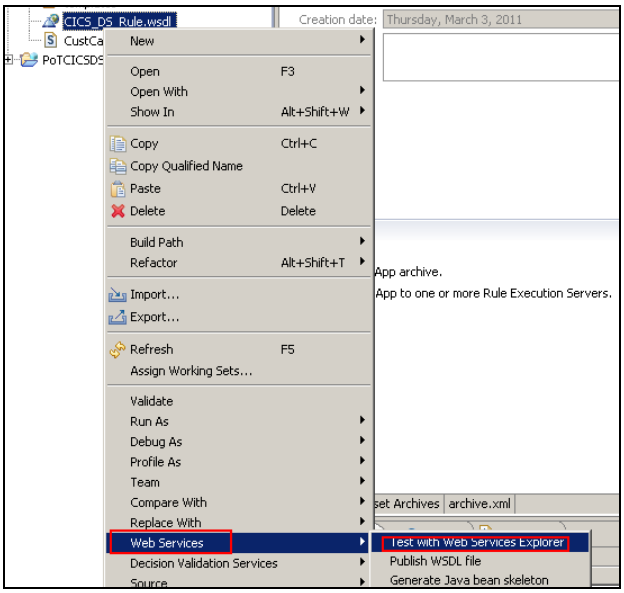
- Click **Finish**

Testing the Web service

Launch the Web Services Explorer from a WSDL file, as follows:



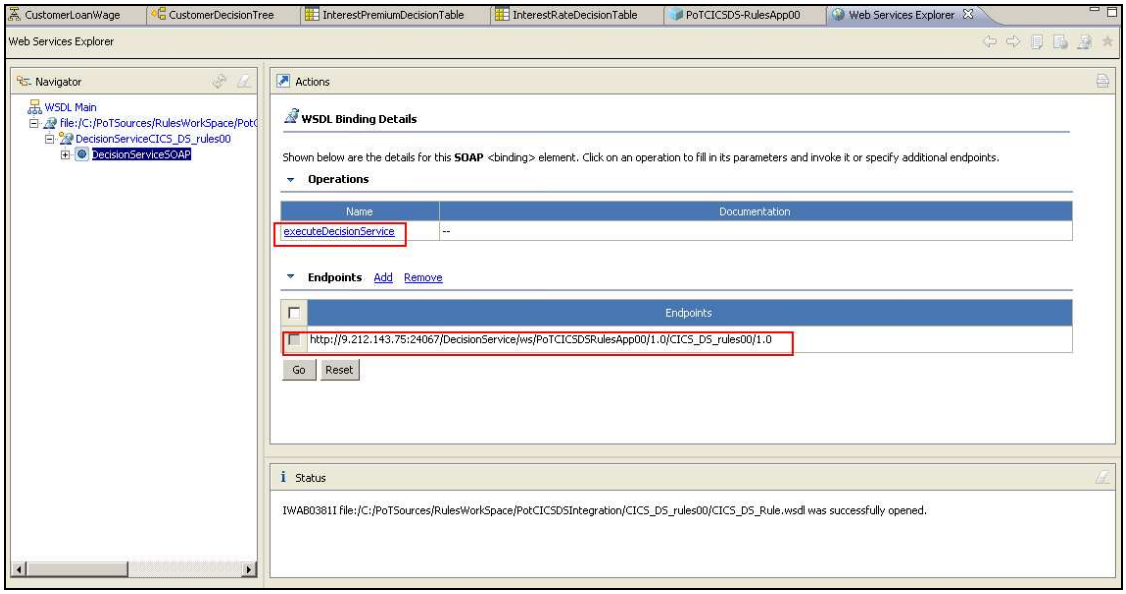
- In the Package Explorer, expand **CICS_DS_rulesxx** (where you need to substitute "xx" with your team number), select the **CICS_DS_Rulesxx.wsdl** file and right-click and choose **Web Services --> Test with Web Services Explorer**



The Web Services Explorer opens. You can double-click on the Web Services Explorer title to display full screen.

The Test wih Web Services Explorer will preload the WSDL in the WSDL page.

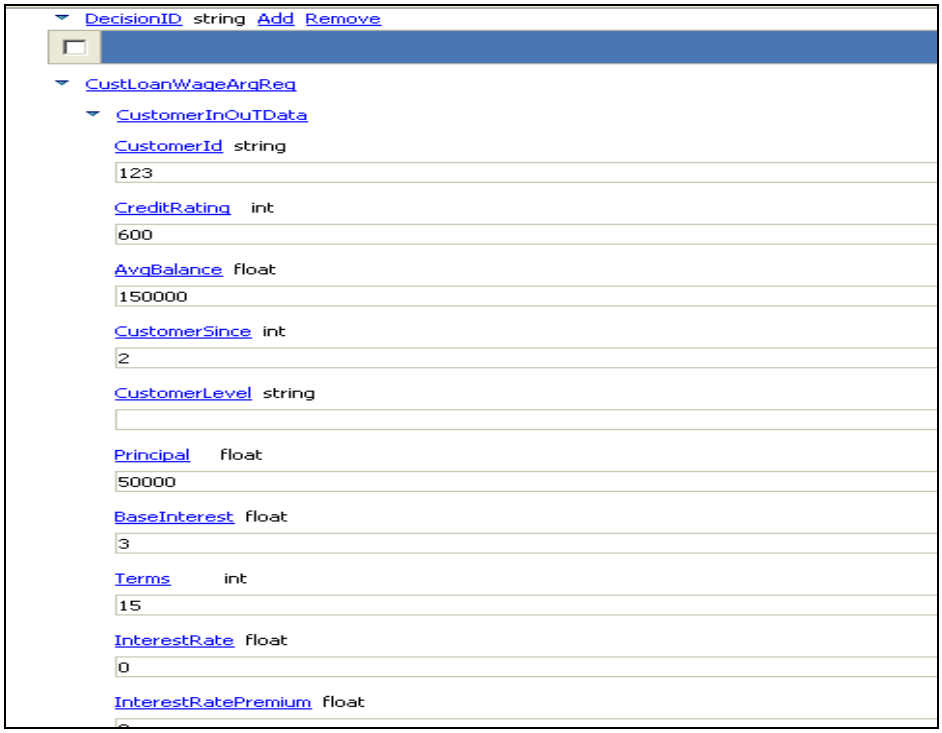
- Select **WSDL Main** in the Navigator pane and expand upto end-point **executionDecisionService**



- In the WSDL Binding Details pane, click the operations **executionDecisionService** to display :

An endpoint for this operation

Fields for each of the parameters of this operation and the type of information the parameter is looking for

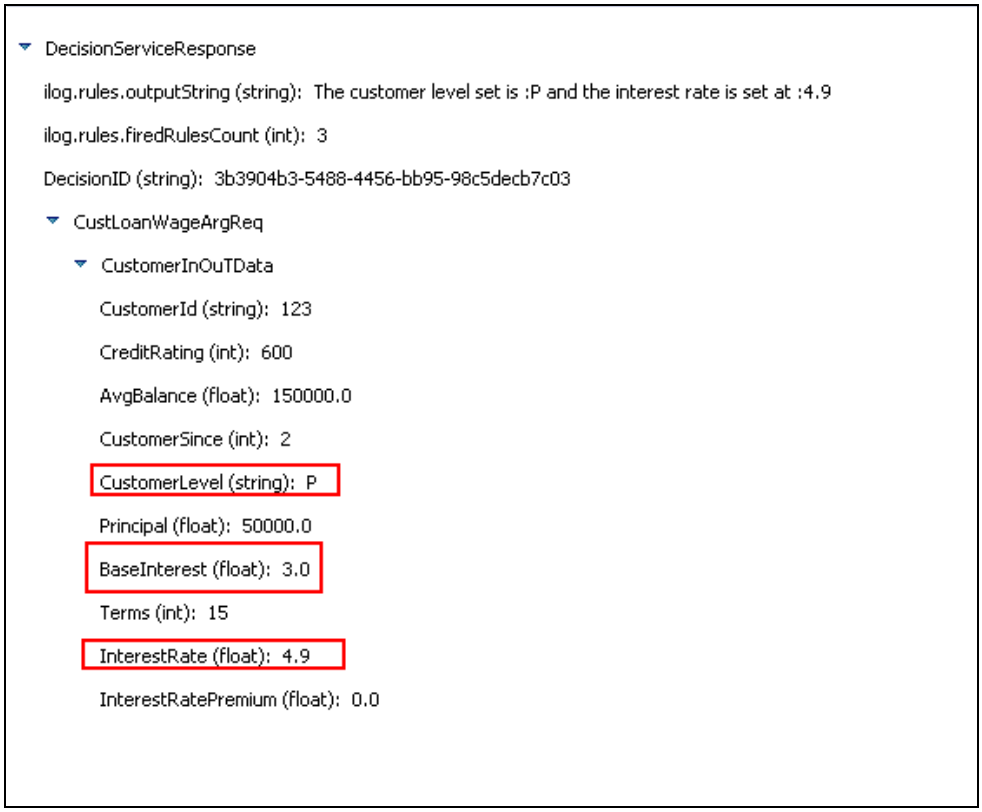


The Form view allows you to enter the parameters for the Web Services operation call. Parameters names are displayed as links whose action is to display a dialog describing embedded XML Schema Definition Language (XSD) information.

- Enter your input for the parameters by picking test data from this table.

Input Parameters Value									
Customer Id	Credit Rating	Avg Balance	Customer Since	Customer Level	Principal	Base Interest	Term	Interest Premium	Interest Rate
123	600	150000	2	blank	500000	3	15	0	0
124	810	40000	7	blank	350000	3	20	0	0
125	715	49000	1	blank	400000	3	20	0	0
126	545	25000	2	blank	500000	3	15	0	0

- Once entered all parameters, scroll down, click Go, as show in the figure below :

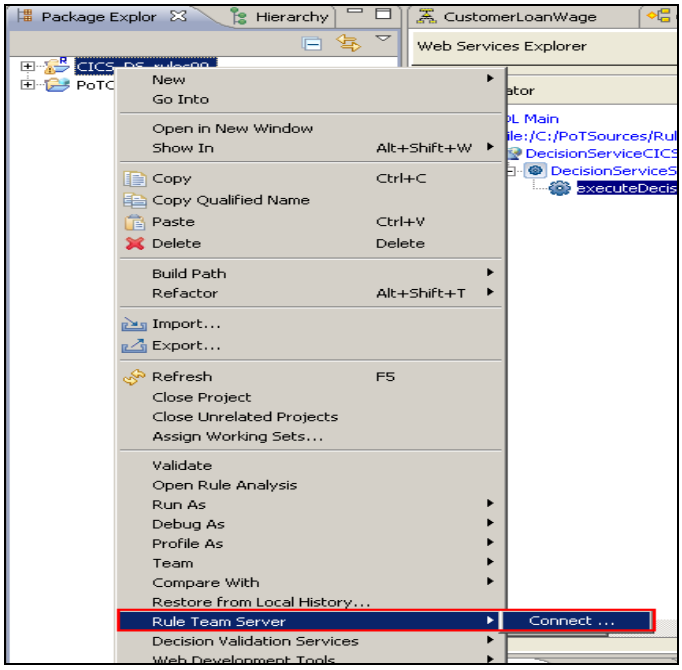


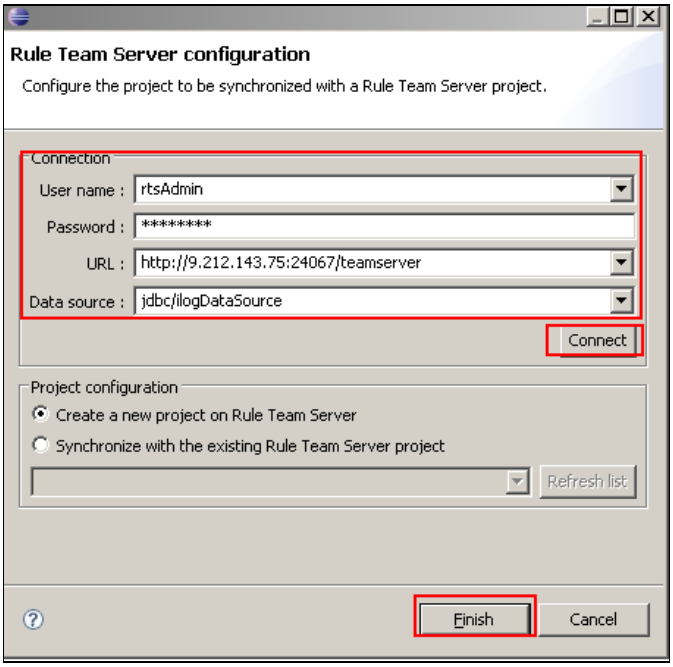
The expected results are shown in the table below:

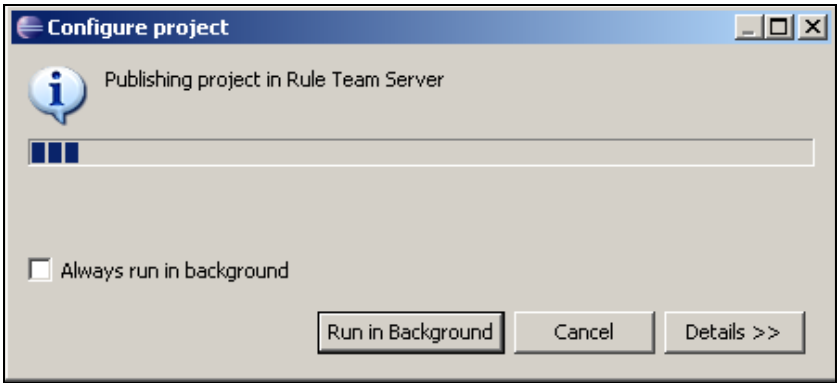
Result Ouput Value	
Customer Level	**Interest Rate**
P	4.95
G	4.75
S	5.0
R	5.7

3.3 Deployment to Rule Team Server

Follow theses steps.







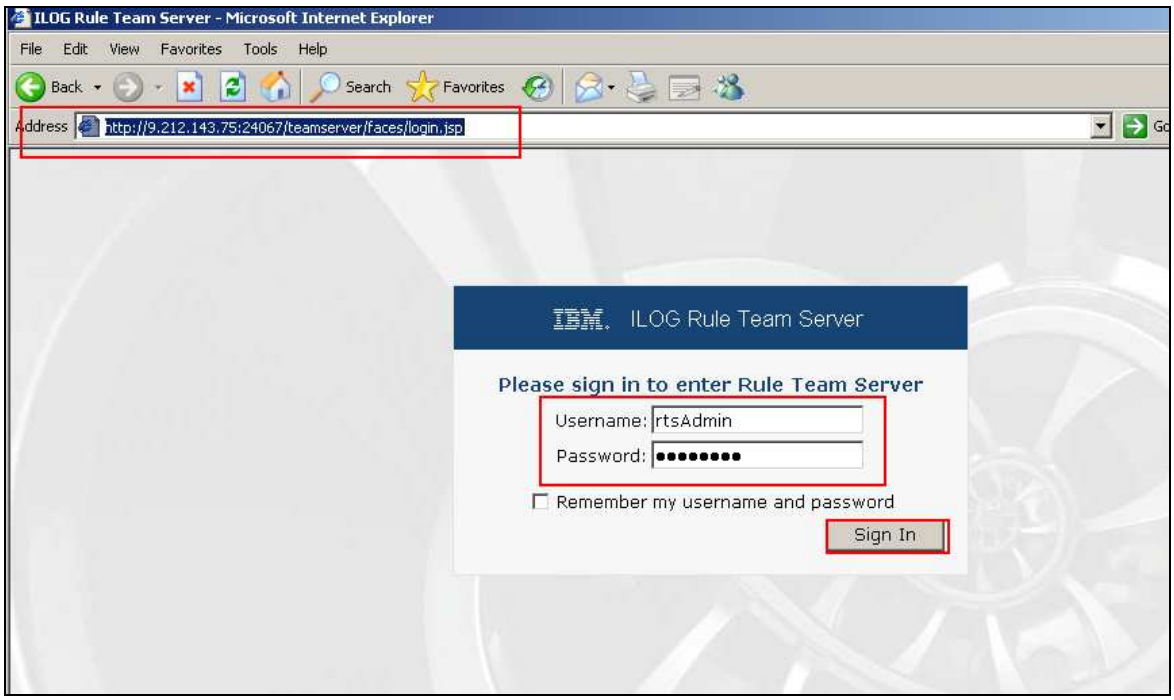
Connect to RTS Admin console using this URL :

- In a browser, access the server at URL *http://9.212.143.75:24067/teamserver*

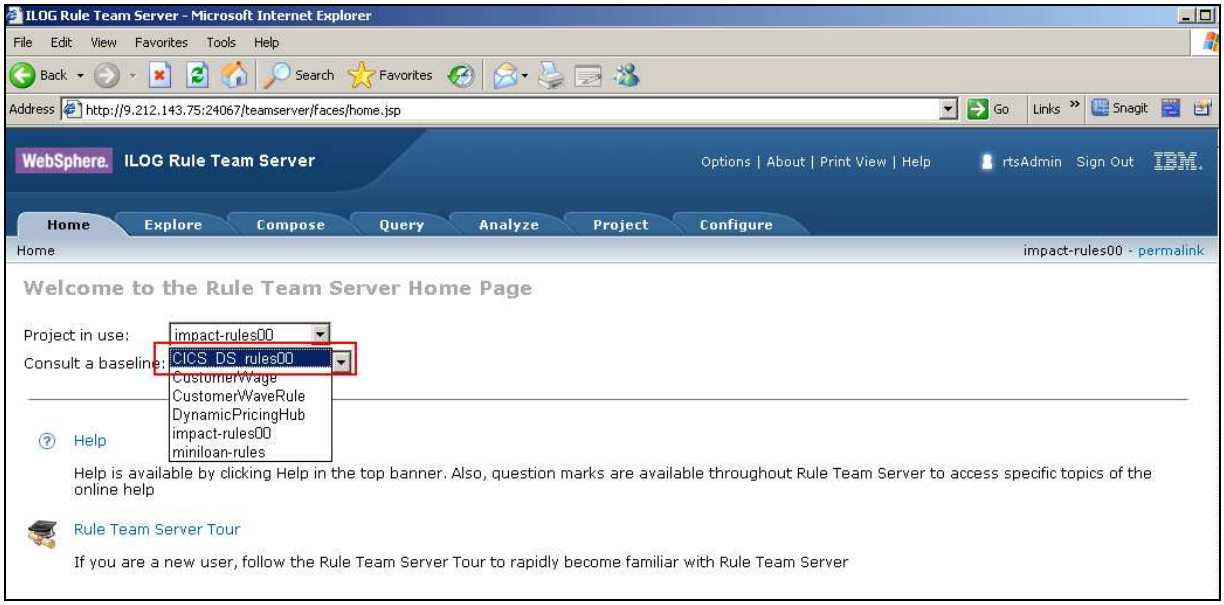
- Sign in to the Rule Team Console using the following details :

Username : < Rule Team Server Userid> ==> *rtsAdmin*

Password : < Rule Team Server Password> ==> *rtsAdmin*

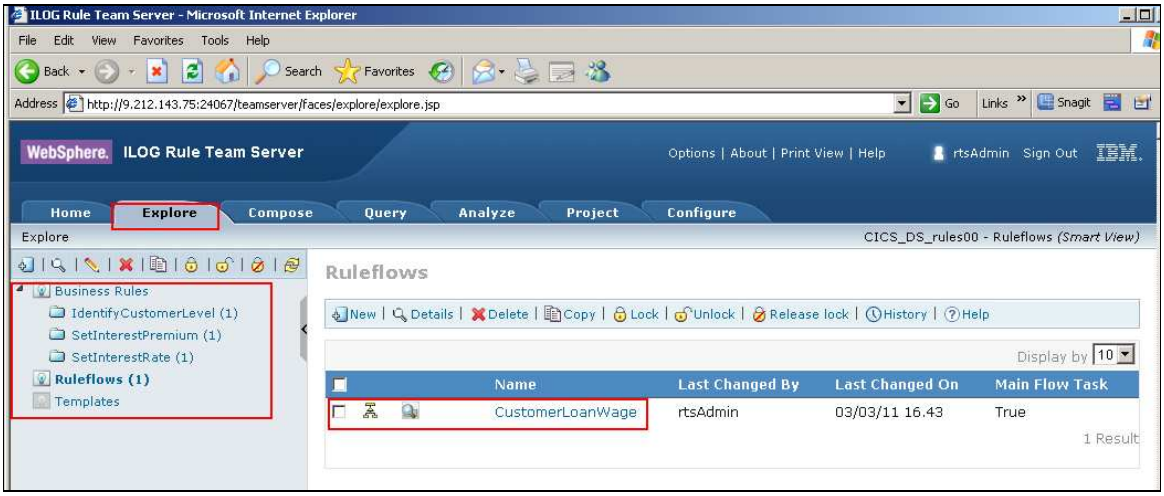


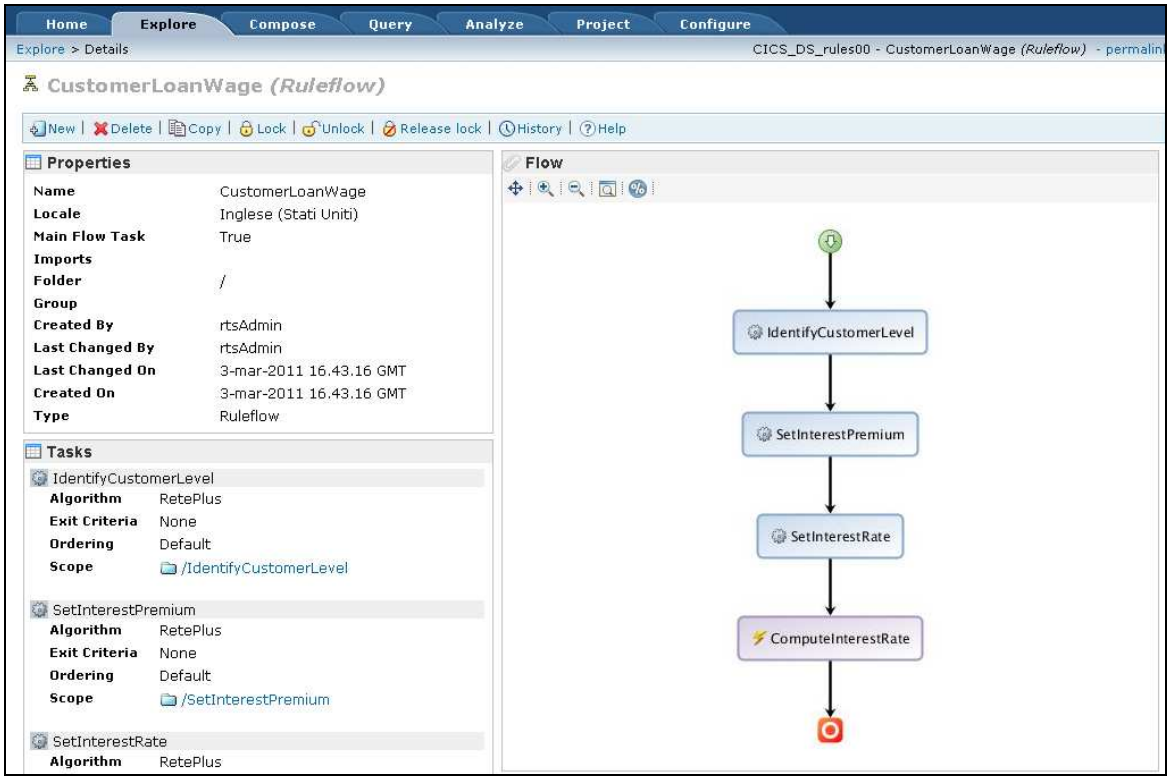
In the Project in use, open the drop-down list and select your project.

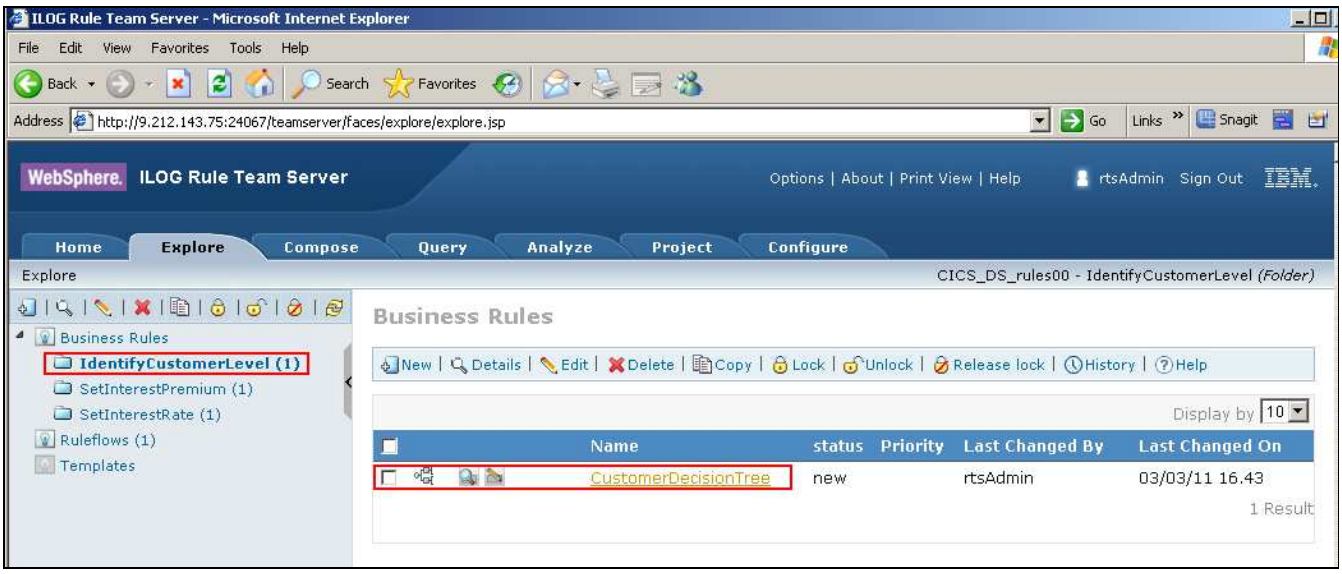


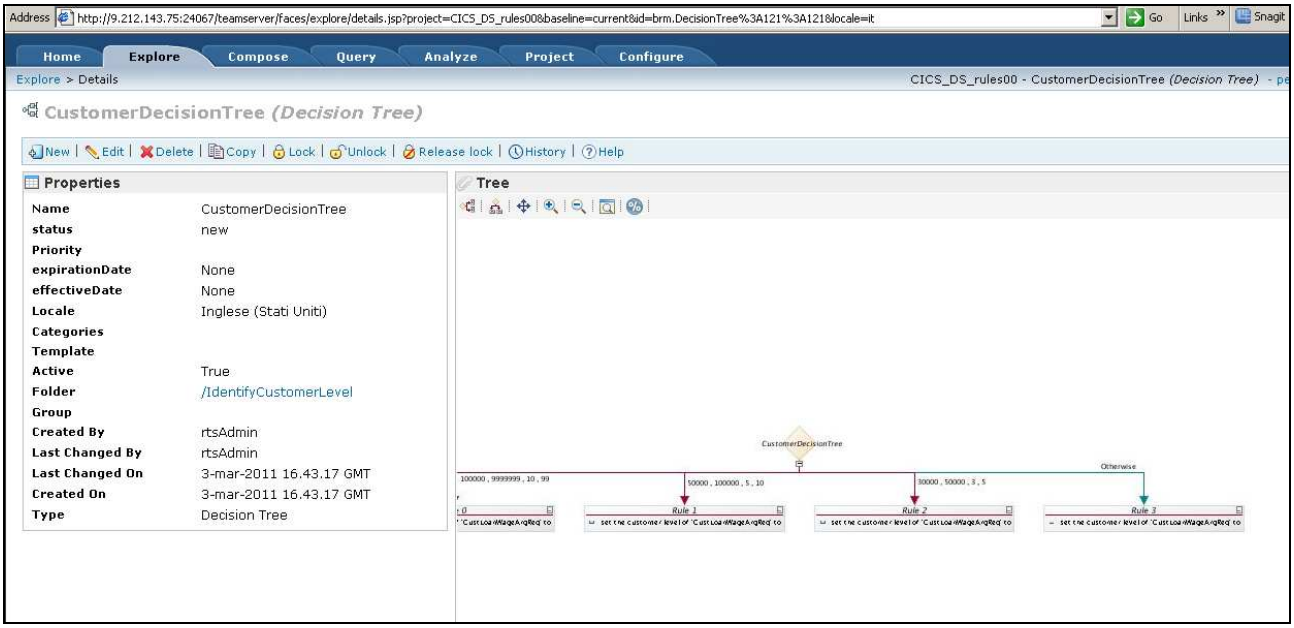
The go to "*explore tab*"

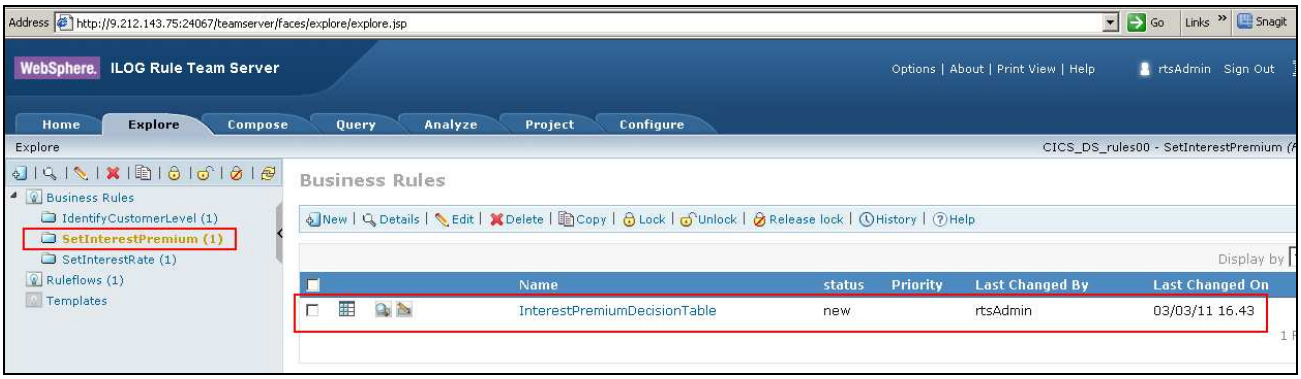
At the left pane, you will see your rule project components and keep time to explore it.

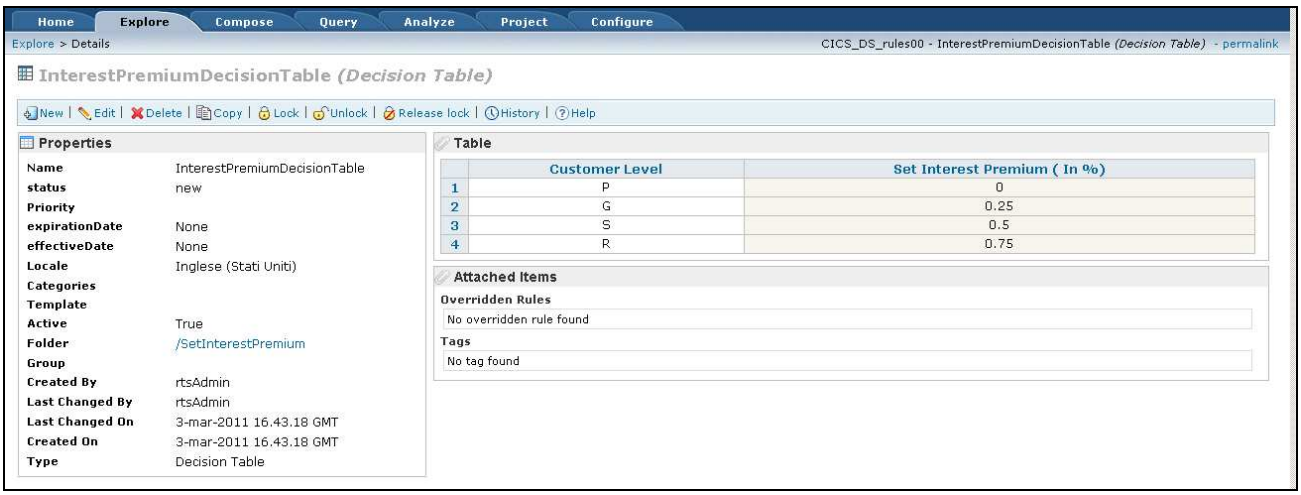



Home
Explore
Compose
Query
Analyze
Project
Configure
Explore > Details
CICS_DS_rules00 - CustomerLoanWage (Ruleflow) - permalin
CustomerLoanWage (Ruleflow)
New | Delete | Copy | Lock | Unlock | Release lock | History | Help
Properties
Name CustomerLoanWage
Locale Inglese (Stati Uniti)
Main Flow Task True
Imports
Folder /
Group
Created By rtsAdmin
Last Changed By rtsAdmin
Last Changed On 3-mar-2011 16.43.16 GMT
Created On 3-mar-2011 16.43.16 GMT
Type Ruleflow
Tasks
IdentifyCustomerLevel
Algorithm RetePlus
Exit Criteria None
Ordering Default
Scope /IdentifyCustomerLevel
SetInterestPremium
Algorithm RetePlus
Exit Criteria None
Ordering Default
Scope /SetInterestPremium
SetInterestRate
Algorithm RetePlus
Flow
IdentifyCustomerLevel
SetInterestPremium
SetInterestRate
ComputeInterestRate


ILOG Rule Team Server - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://9.212.143.75:24067/teamserver/faces/explore/explore.jsp
Go Links Snagit
WebSphere. ILOG Rule Team Server
Options | About | Print View | Help
rtsAdmin Sign Out
IBM.
Home
Explore
Compose
Query
Analyze
Project
Configure
Explore
CICS_DS_rules00 - IdentifyCustomerLevel (Folder)
Business Rules
IdentifyCustomerLevel (1)
SetInterestPremium (1)
SetInterestRate (1)
Ruleflows (1)
Templates
Business Rules
New | Details | Edit | Delete | Copy | Lock | Unlock | Release lock | History | Help
Display by 10
Name
status
Priority
Last Changed By
Last Changed On
CustomerDecisionTree
new
rtsAdmin
03/03/11 16.43
1 Result

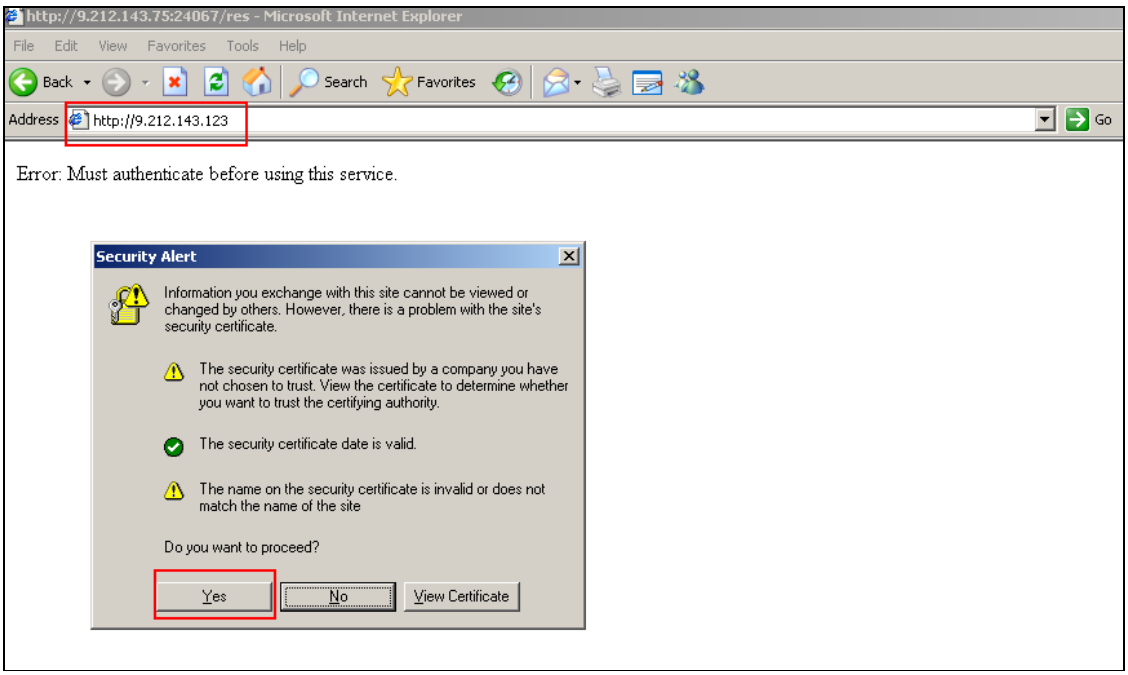

Address http://9.212.143.75:24067/teamserver/faces/explore/details.jsp?project=CICS_DS_rules00&baseline=current&id=brm.DecisionTree%3A121%3A1218&locale=it
Home Explore Compose Query Analyze Project Configure
Explore > Details
CICS_DS_rules00 - CustomerDecisionTree (Decision Tree)
CustomerDecisionTree (Decision Tree)
New | Edit | Delete | Copy | Lock | Unlock | Release lock | History | Help
Properties
Name CustomerDecisionTree
status new
Priority
expirationDate None
effectiveDate None
Locale Inglese (Stati Uniti)
Categories
Template
Active True
Folder /IdentifyCustomerLevel
Group
Created By rtsAdmin
Last Changed By rtsAdmin
Last Changed On 3-mar-2011 16.43.17 GMT
Created On 3-mar-2011 16.43.17 GMT
Type Decision Tree
Tree

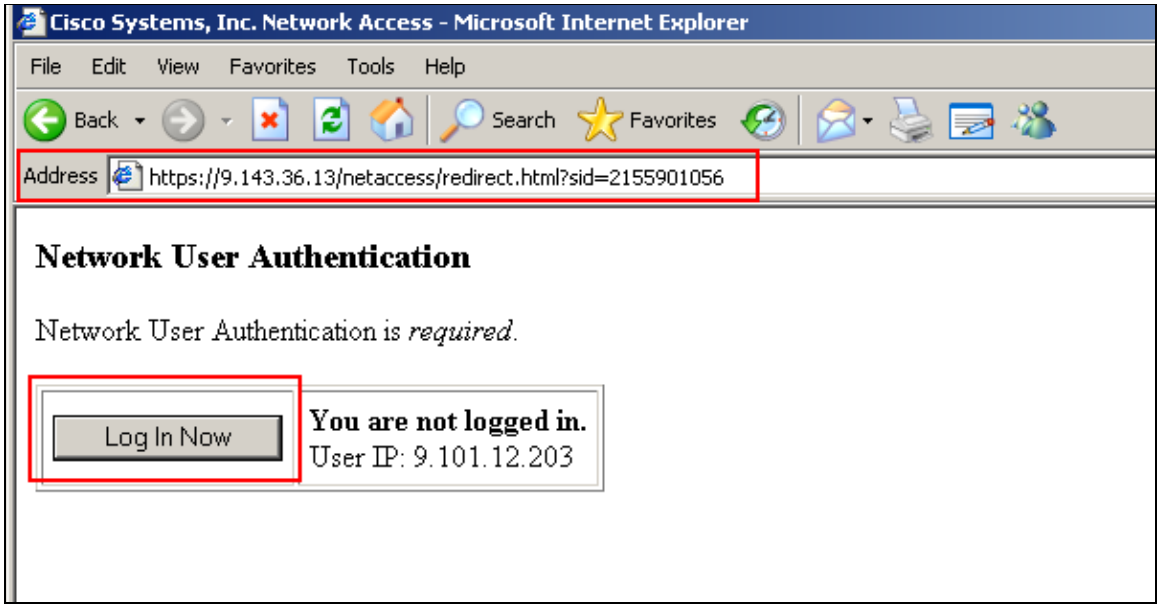

Address http://9.212.143.75:24067/teamserver/faces/explore/explore.jsp
WebSphere. ILOG Rule Team Server
Options | About | Print View | Help
rtsAdmin Sign Out
Home Explore Compose Query Analyze Project Configure
Explore
CICS_DS_rules00 - SetInterestPremium
Business Rules
IdentifyCustomerLevel (1)
SetInterestPremium (1)
SetInterestRate (1)
Ruleflows (1)
Templates
New | Details | Edit | Delete | Copy | Lock | Unlock | Release lock | History | Help
Name status Priority Last Changed By Last Changed On
InterestPremiumDecisionTable new rtsAdmin 03/03/11 16.43

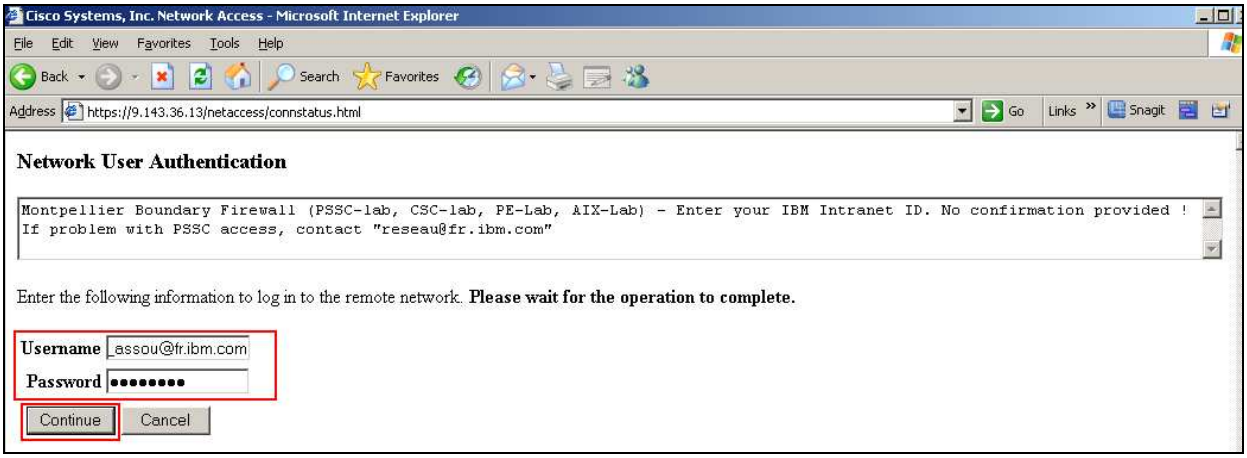

Home Explore Compose Query Analyze Project Configure
Explore > Details
CICS_DS_rules00 - InterestPremiumDecisionTable (Decision Table) - permalink
InterestPremiumDecisionTable (Decision Table)
New | Edit | Delete | Copy | Lock | Unlock | Release lock | History | Help
Properties
Name InterestPremiumDecisionTable
status new
Priority
expirationDate None
effectiveDate None
Locale Inglese (Stati Uniti)
Categories
Template
Active True
Folder /SetInterestPremium
Group
Created By rtsAdmin
Last Changed By rtsAdmin
Last Changed On 3-mar-2011 16.43.18 GMT
Created On 3-mar-2011 16.43.18 GMT
Type Decision Table
Table
Customer Level
Set Interest Premium (In %)
1 P 0
2 G 0.25
3 S 0.5
4 R 0.75
Attached Items
Overridden Rules
No overridden rule found
Tags
No tag found

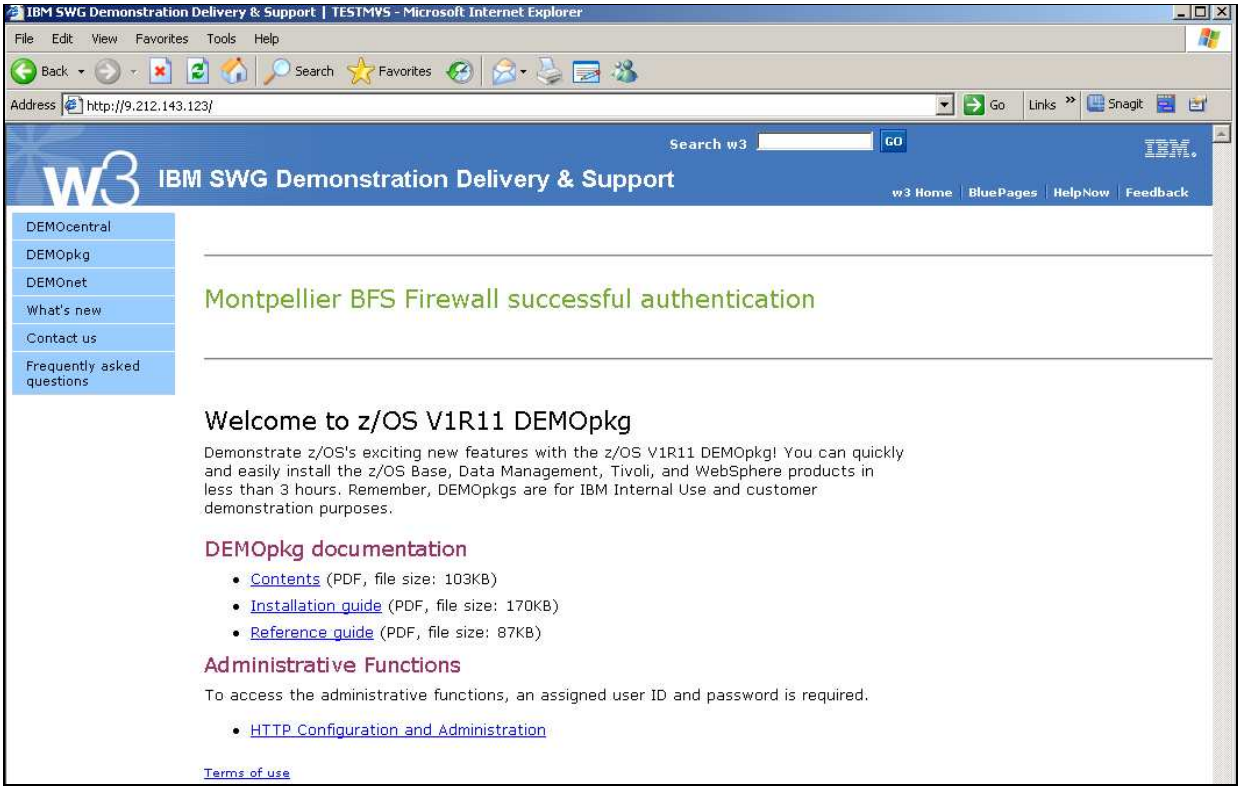
3.4 Annexes

3.4.1 Access through BFS of the System z at Montpellier






Cisco Systems, Inc. Network Access - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address https://9.143.36.13/netaccess/connstatus.html
Go Links Snagit
Network User Authentication
Montpellier Boundary Firewall (PSSC-lab, CSC-lab, PE-Lab, AIX-Lab) - Enter your IBM Intranet ID. No confirmation provided !
If problem with PSSC access, contact "reseau@fr.ibm.com"
Enter the following information to log in to the remote network. Please wait for the operation to complete.
Username assou@fr.ibm.com
Password ••••••••
Continue
Cancel


IBM SWG Demonstration Delivery & Support | TESTMVS - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://9.212.143.123/
Go Links Snagit
Search w3 GO
IBM.
w3
IBM SWG Demonstration Delivery & Support
w3 Home | BluePages | HelpNow | Feedback
DEMOcentral
DEMOpkg
DEMOnet
What's new
Contact us
Frequently asked questions
Montpellier BFS Firewall successful authentication
Welcome to z/OS V1R11 DEMOpkg
Demonstrate z/OS's exciting new features with the z/OS V1R11 DEMOpkg! You can quickly and easily install the z/OS Base, Data Management, Tivoli, and WebSphere products in less than 3 hours. Remember, DEMOpkgs are for IBM Internal Use and customer demonstration purposes.
DEMOpkg documentation
Contents (PDF, file size: 103KB)
Installation guide (PDF, file size: 170KB)
Reference guide (PDF, file size: 87KB)
Administrative Functions
To access the administrative functions, an assigned user ID and password is required.
HTTP Configuration and Administration
Terms of use

Lab 4 Implementing Event Based Decision Making

In this part we will use the business event processing provided with decision server to integrate the decision into the application.

First we will use Design Data to map the necessary fields from events sources in order to prepare a new business application.

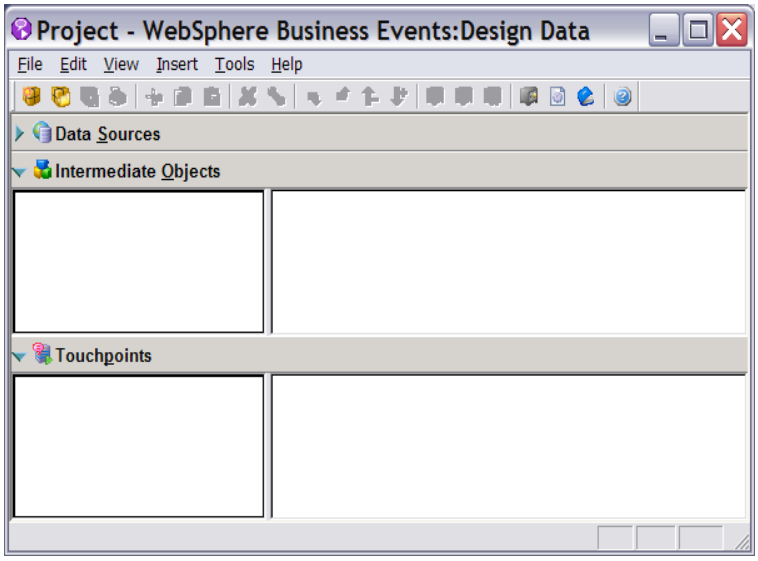
4.1 Creating an Event Project

4.1.1 Open Design Data

Design Data is the development environment for event processing applications. Developers specify the connections and events that interface with other applications. This includes the information models and schemas used to describe the events and actions that can take place.

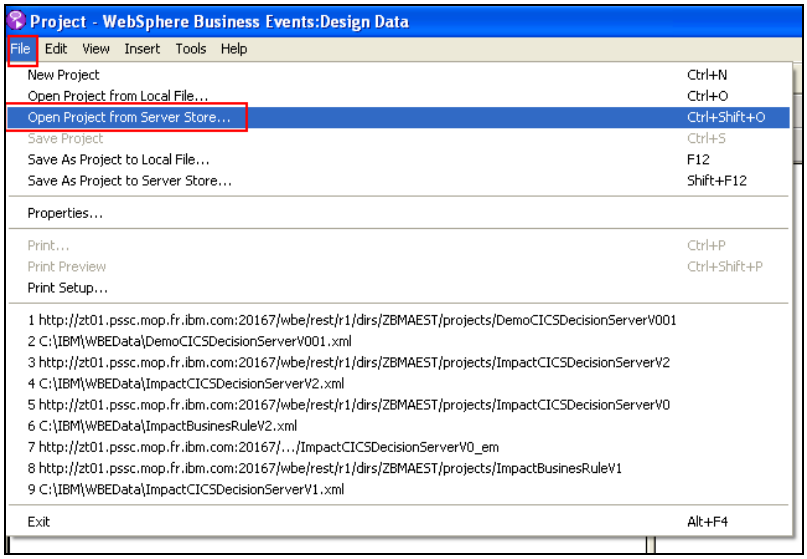
To launch Design Data:

1. From the **Start** menu, click **Start > All Programs > IBM WebSphere Business Events V7.0.1 > Design Data**.
2. Design Data opens with an empty project.



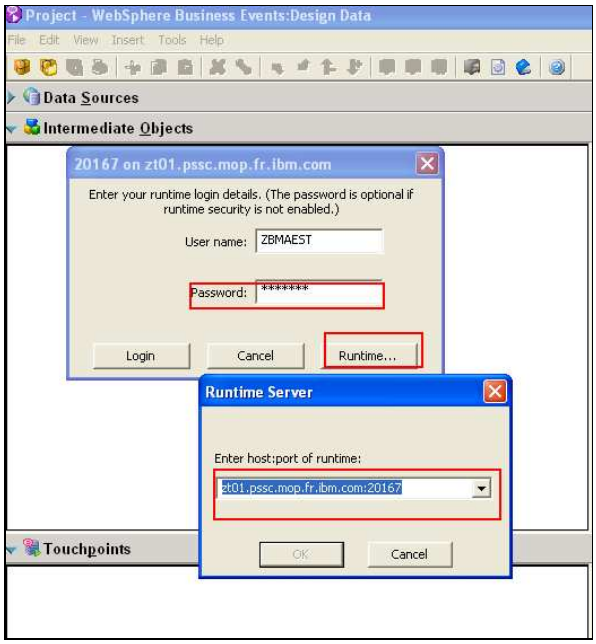
3. Get a model from store (This model contains the WBE project setup by CICS Event Processing data from CICS Explorer)

4. From the **File** menu select Open Project from Server Store



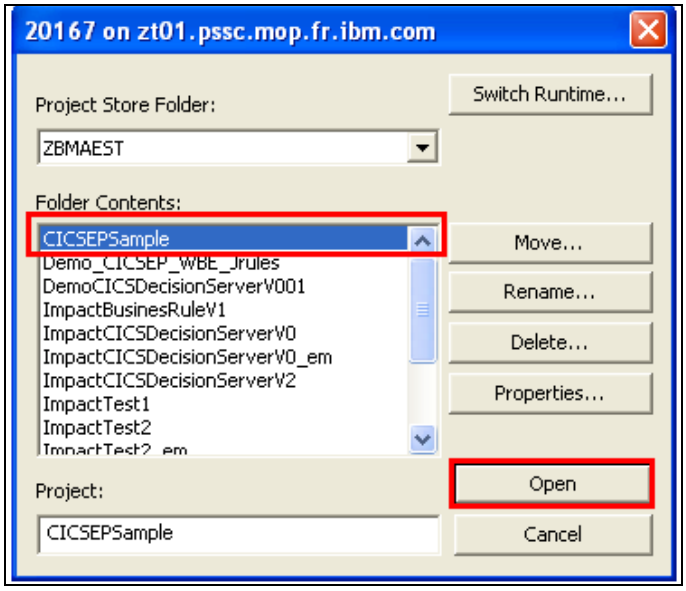
5. Click the Runtime... button to configure WBE Administration location on Montpellier z/OS plateform. URL : **zt01.pssc.mop.fr.ibm.com:20167**

Click **OK**

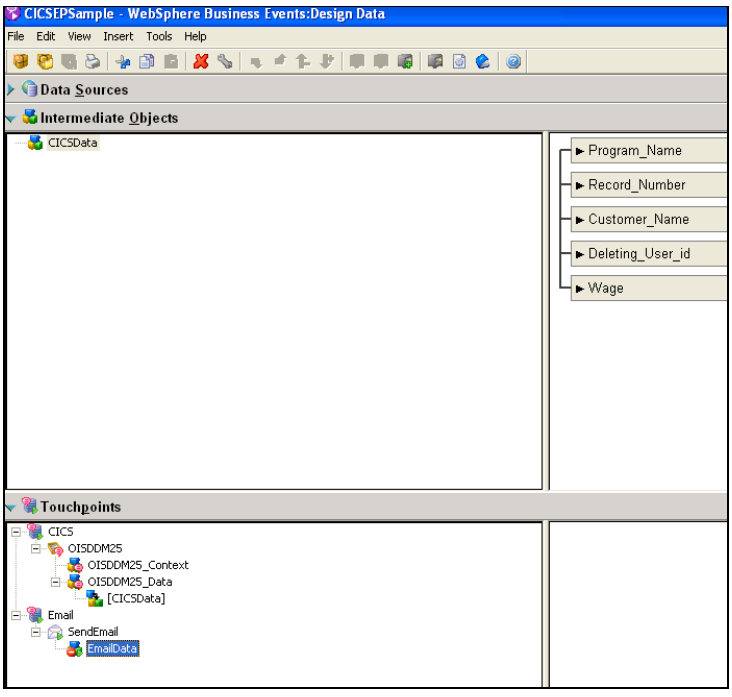


6. Enter the username and password ZBMAEST / zbmaest and click **Login**.

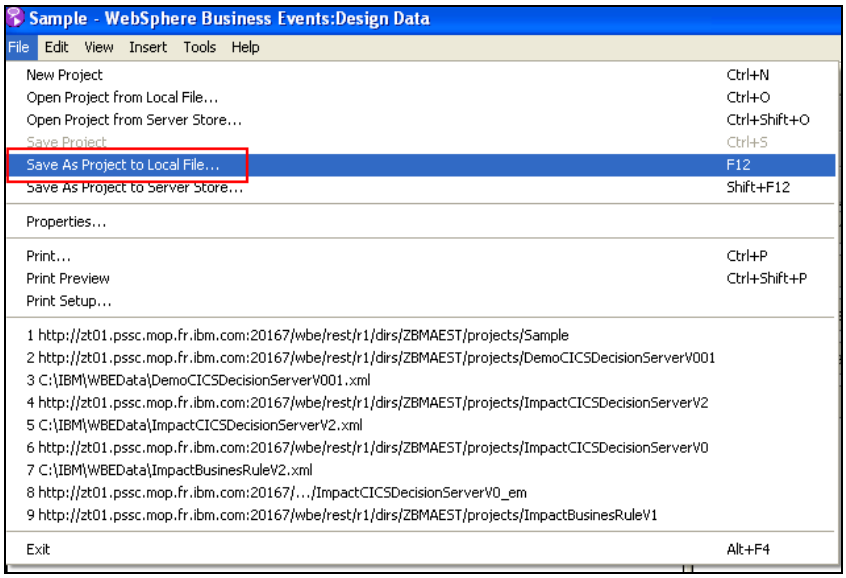
7. In the **Folder Contents**, select ***CICSEPSample*** project and click **Open**.



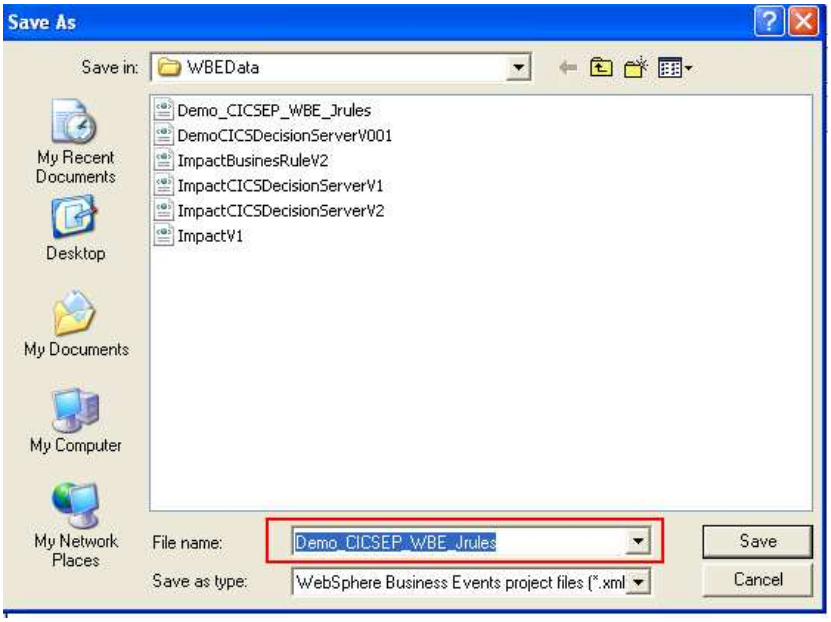
8. The project is now ready to be configured as shown below.



9. From the **File** menu select **Save As Project to Local File...**



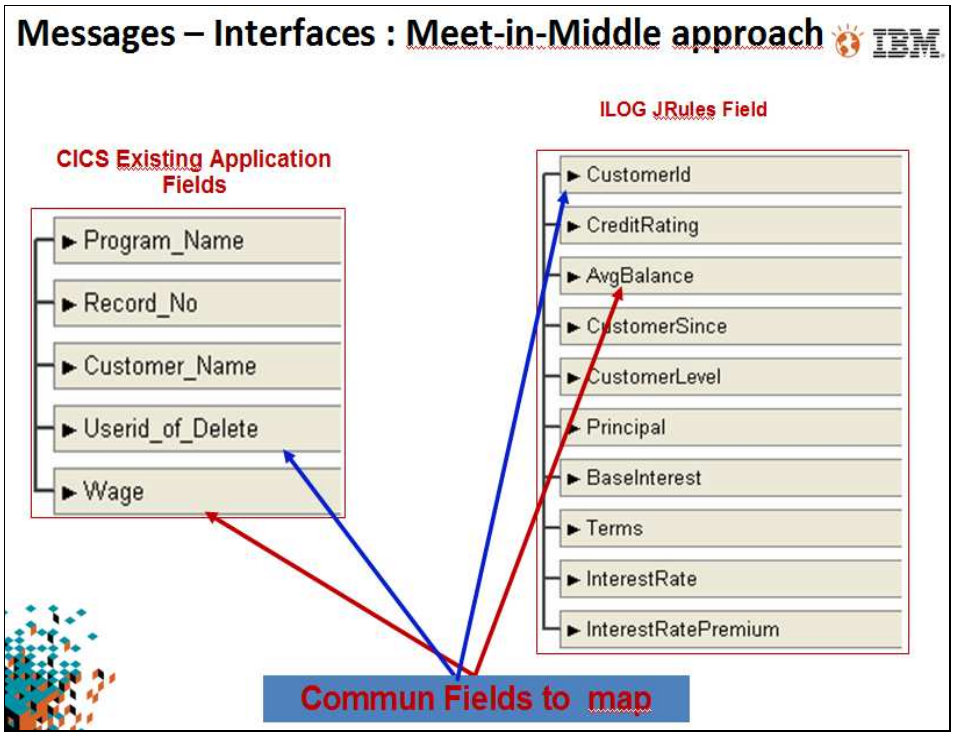
10. In the **Project**: field **type Demo_CICSEP_WBE_JrulesVxx** (where "**xx**" is your team number to substitute) and click **Save**.



4.1.2 Step 3 Import the decision service and information models

In this next step you are going to use the information model defined for the decision service to setup the structure of the events, actions and intermediate objects. The intermediate objects act as variables that can be referenced when making event based decisions.

So, we need to share some data coming from CICS and not necessary to make ILOG Jrules Decision. We call this procedure , Meet-in-the-middle mapping.



In our case :

Data from CICS Event :

Fields	Format
Programe_Name	String
Record_Number	Real
Customer_Name	String
Userid_Of_Delete	String
Wage	Real

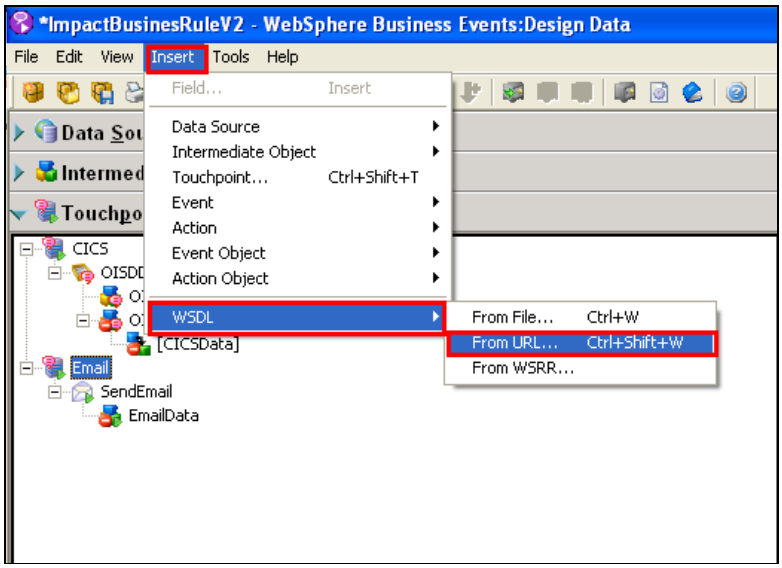
Input / Output Data needed for Jrules

Fields	Format
CustomerId	String
CreditRating	Integer
AvgBalance	Float
CustomerLevel	String
Principal	Float
BaseInterest	Float
Terms	Integer
InterestRate	Float
InterestRatePremium	Float

Mapping fields between CICS Data and Jrules interfaces

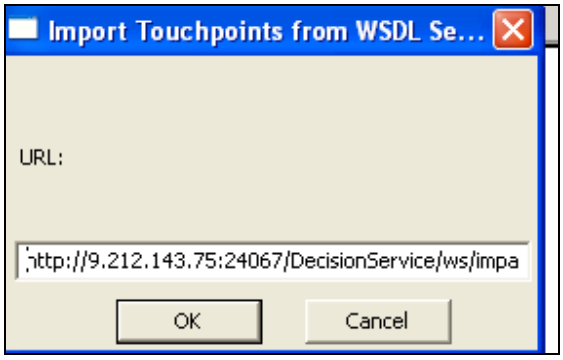
CICS Data Fields	ILOG Jrules Fields	Format
Customer_Name	CustomerId	String
Wage	AvgBalance	Real

1. From the Insert menu, select WSDL > From URL... .

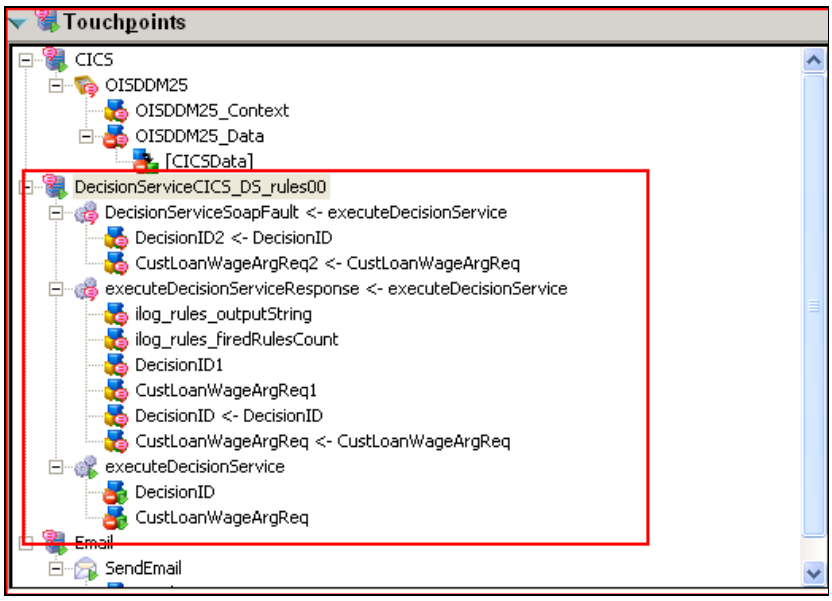


2. In Import Touchpoint from WSDL, type the Rule WSDL URL location

http://9.212.143.75:24067/DecisionService/ws/PoTCICSDSRulesApp00/1.0/CICS_DS_rules00/1.0?WSDL



3. The web service is imported into the Event project as shown below.



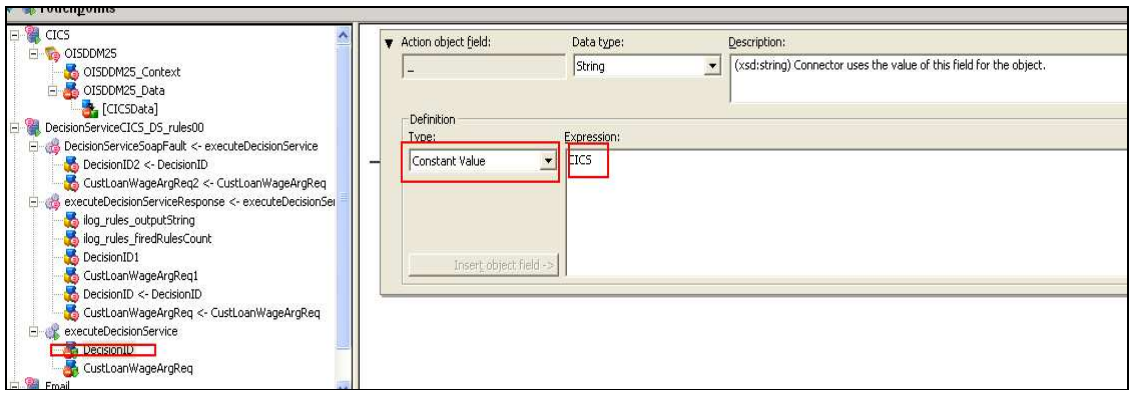
4. Save the project and examine the web service structure.

5. executeDecisionService represents the action that is executed to invoke the decision service together with its parameters represented as individual Event Objects.

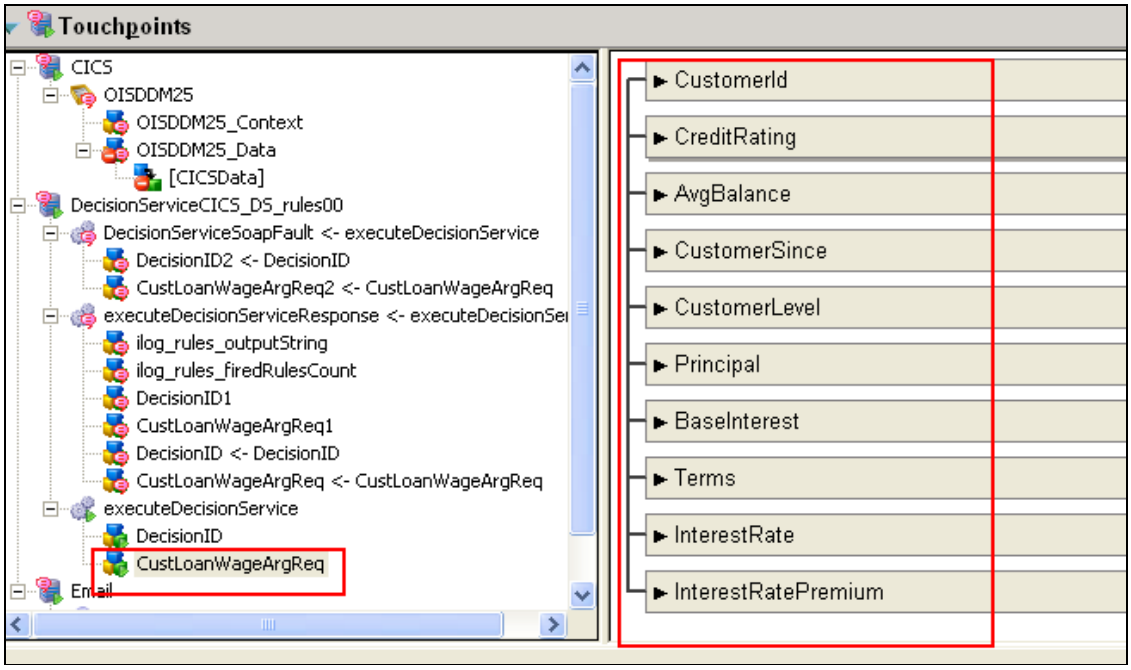
6. Initialize necessaries data for Rule

Some fields must be initialized for the Web Service input Data

7. Click on **DecisionID** and on the right panel, select Constant Value in the Type combo box and set Expression to **CICS**



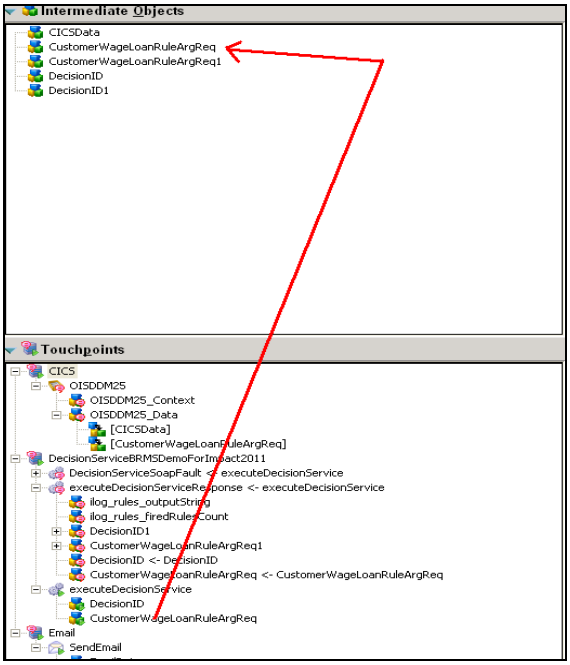
8. Do the for these others fields in **CustLaonWageArgReq**.



Input Parameters Value									
Customer Id	**Credit Rating**	**Avg Balance**	**Customer Since**	**Customer Level**	**Principal**	**Base Interest**	**Term**	**Interest Premium**	**Interest Rate**
123	600	150000	2	blank	500000	3	15	0	0

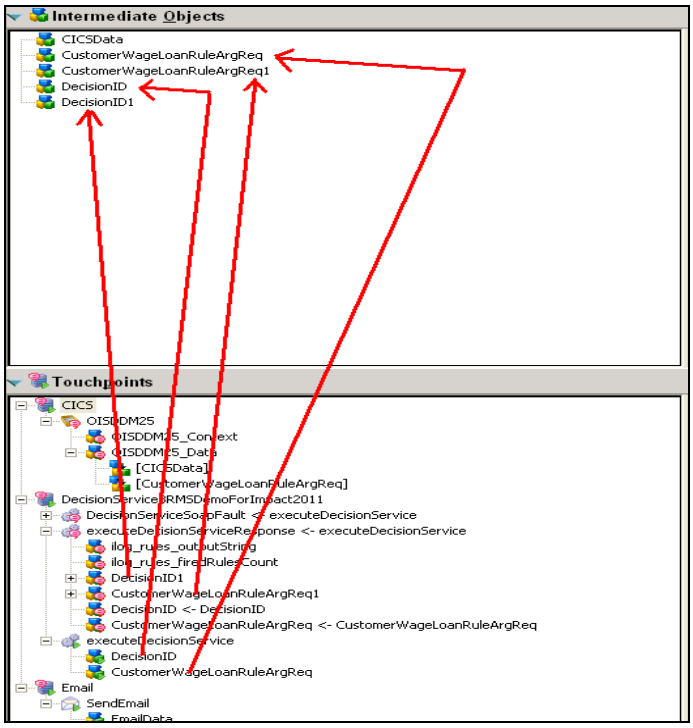
9. executeDecisionServiceResponse is the event that gets created in response to the decision service. Note that the input parameters are copied unmodified to this event as well as the returned parameters.

10. Select the CustLoanWageArgReq object, drag it onto the Intermediate Objects panel and release. An Intermediate object with same fields as the **CustLoanWageArgReq** event object is setup.

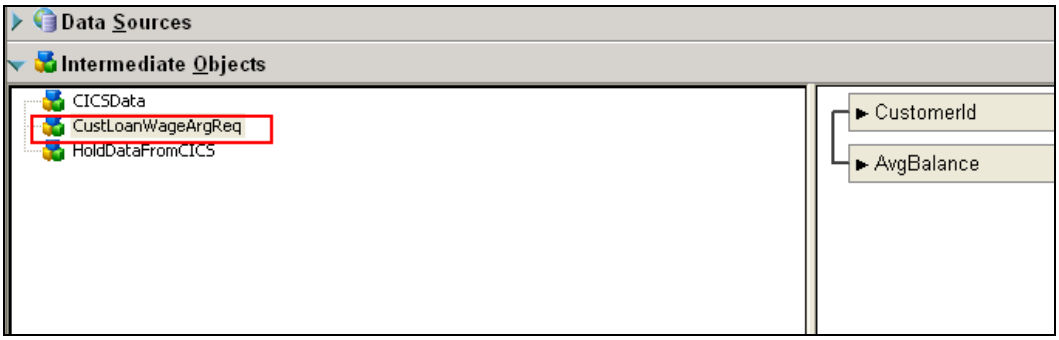


As shown in step 6, we need only two field from CICS to construct the message for Jrules: CusomerID and Wage fields.

11. Select the **DecisionId**, **DecisionId1**, **CustLoanWageArgReq**1 for object, drag it onto the Intermediate Objects panel and release.

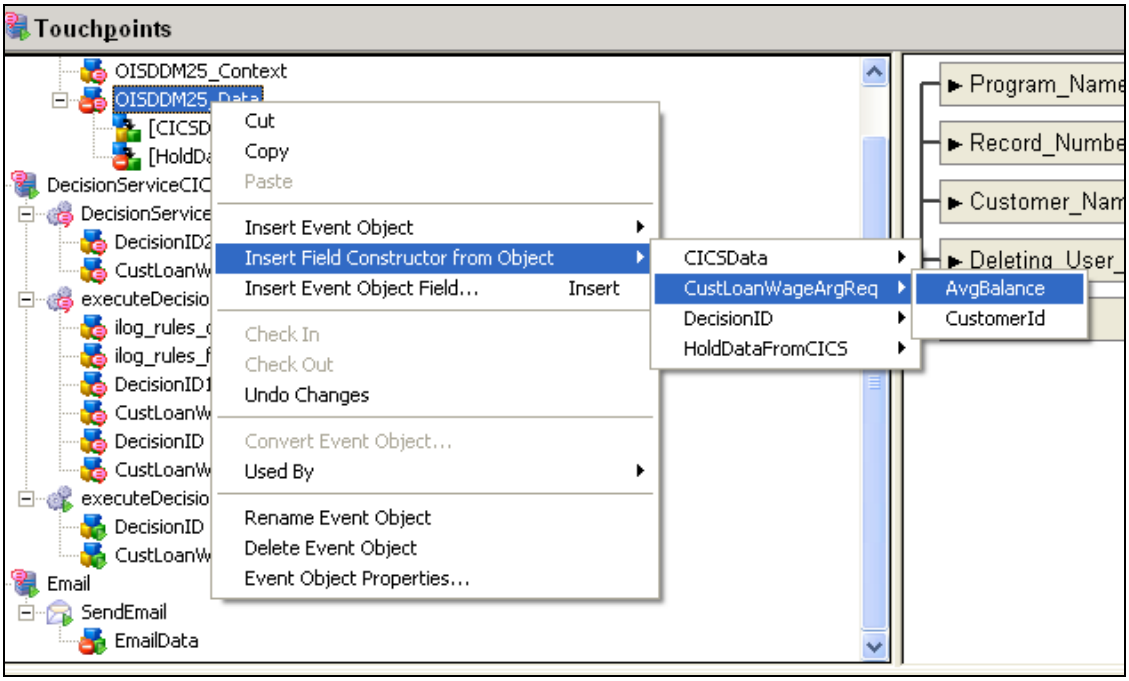


12. In Intermediate Object pane, click on **CustLoanWageArgReq** and delete all fields excepts ***CustomerId*** and ***AvgBalance***



13. Do the mapping for **CustLoanWageArgReq** object onto an existing **CustLoanWageArgReq** intermediate object. Right click the **CustLoanWageArgReq** object and select **Insert Field Constructor from Object > CustLoanWageArgReq > AvgBalance**. This maps the returned **CustLoanWageArgReq AvgBalance** field into the CustLoanWageArgReq intermediate object. In this case we have only created one field so need to repeat this for each of the other fields in the loan object.

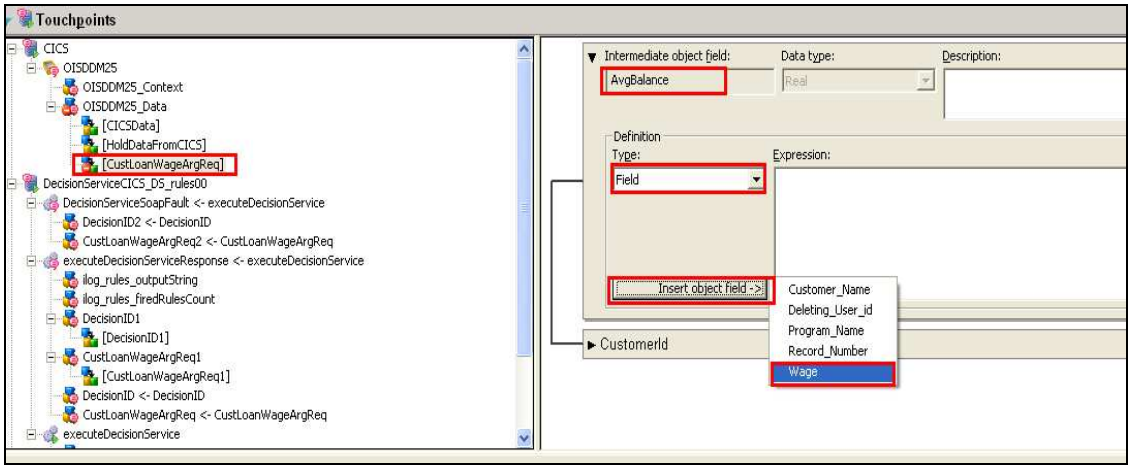
Right click CustLoanWageArgReq **> [**CustLoanWageArgReq**]** and select **Insert Field Constructor > CustomerId**



14. Select the CustLoanWageArgReq > [CustLoanWageArgReq] intermediate object constructor and open up the twistie for the AvgBalance field.

In the Definition Type field select **Field** from the pulldown.

Click **Insert object field** and select **Wage** (Field from CICS) as shown below.



15. We use JavaScript as Jrules return a trailing space at the end of **CustomerID**. So this Java Script code will be put in the field constructors for **CustomerId** and **Customer_Name** in the CICS Event TouchPoint.

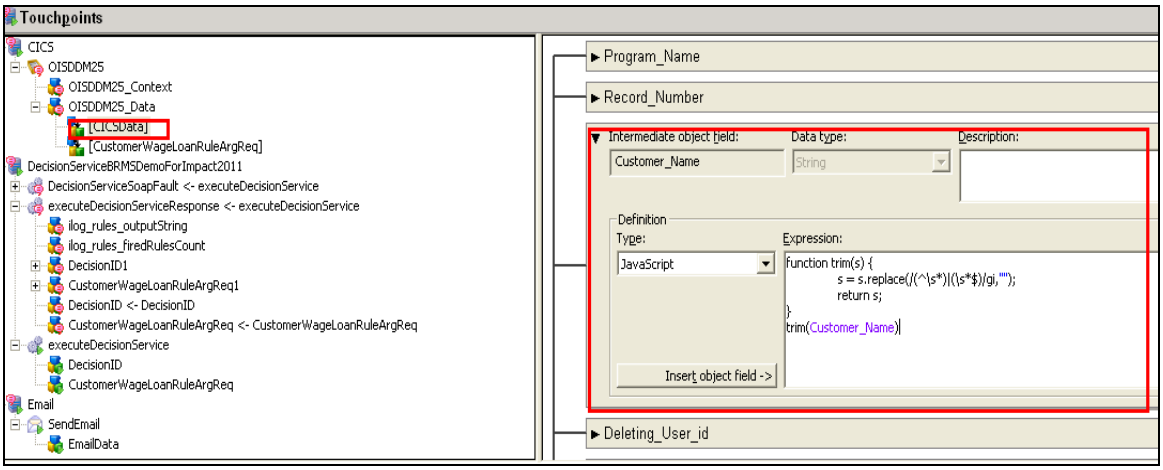
- Select the CICSData > [CICSData] intermediate object constructor and open up the twistie for the **Deleting_User_id** field.

In the Definition Type field select **JavaScript** from the pulldown

.

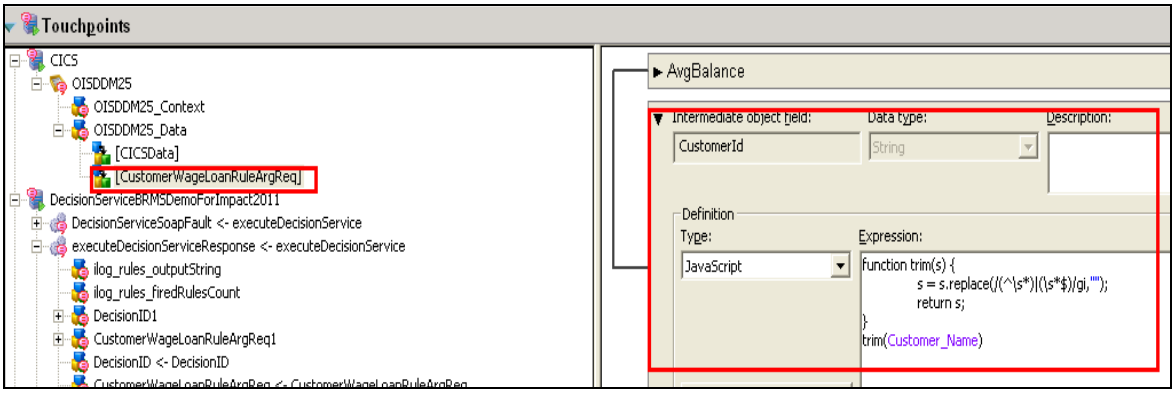
Click **Type this code** as shown below.

```
function trim(s) {

        s = s.replace(/(^\s*)|(\s*$)/gi,"");

        return s;

}

trim(Customer_Name)
```

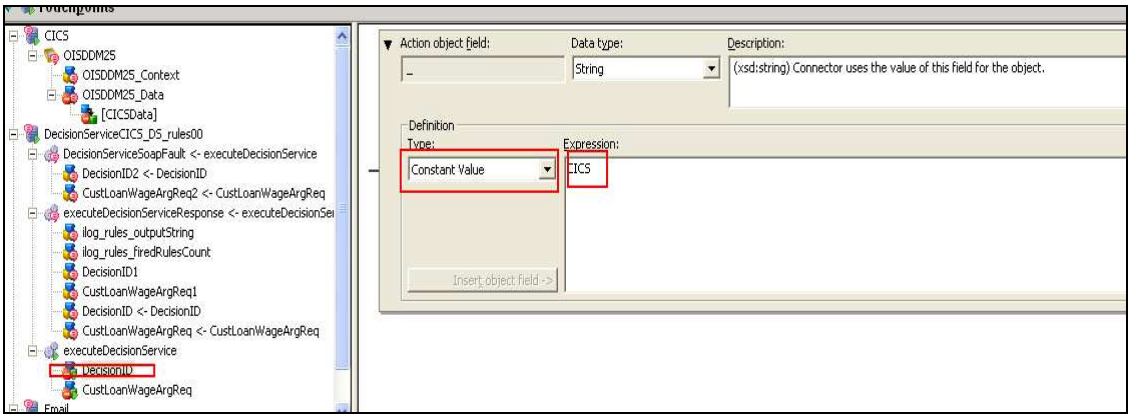


- Select the CustLoanWageArgReq > [CustLoanWageArgReq] intermediate object constructor and open up the twistie for the **CustomerId** field.

In the Definition Type field select **JavaScript** from the pulldown



16. The next error we need to resolve is the DecisionID field in the executeDecisionService Action. Select the object and open the action object field. This should be set to an arbitrary constant value as shown below.

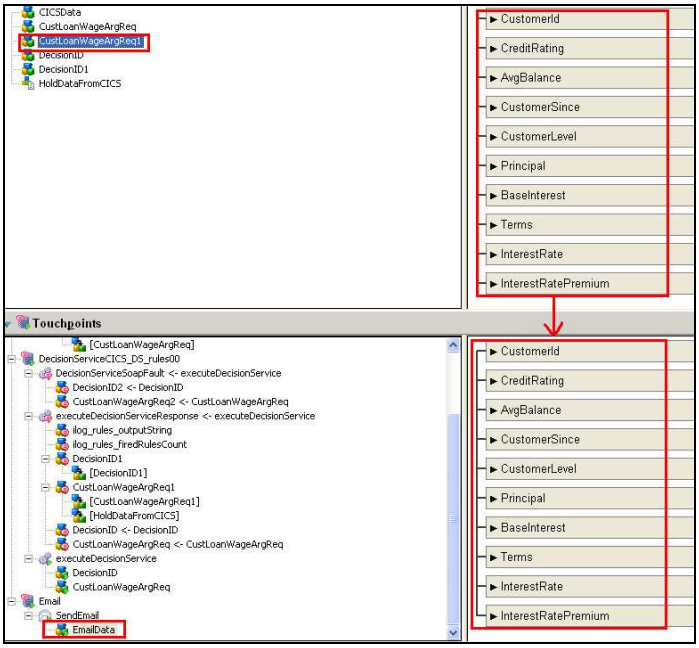


17. Save your work – you have now mapped the intermediate objects that you use for the event processing to the parameters used in the Rule based Decision making.

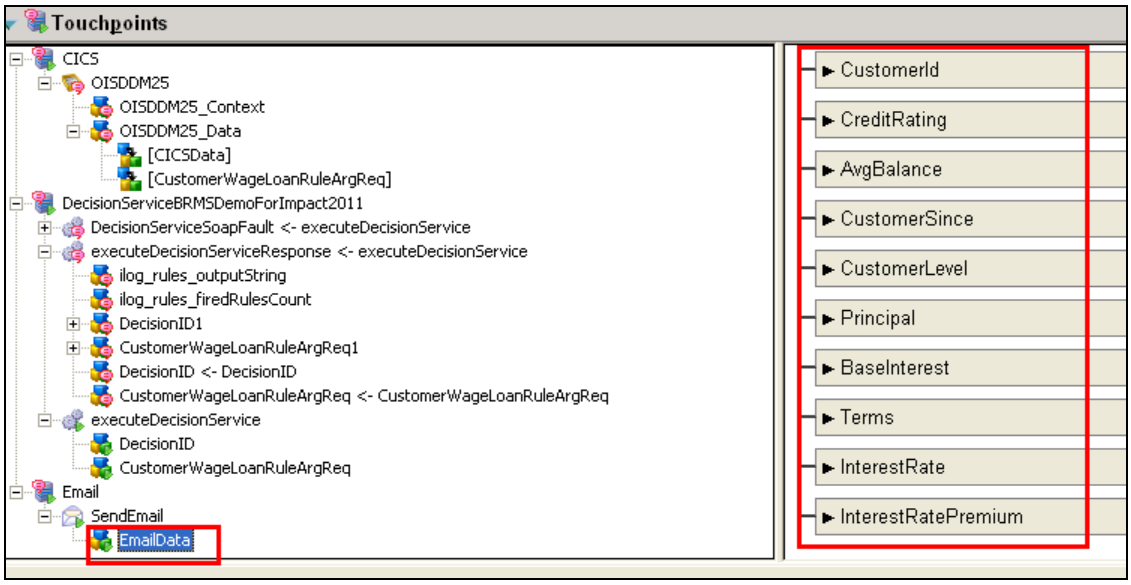
4.1.3 Step 4 Map the intermediate objects to Email actions

The final development step in designing the Event processing is to map the eMail Data

1. Select Email data in the TouchPoint.
   Click on the **CustLoanWageArgReq1** intermediate object and select the all fields in the right panel and drag it to the **eMail** data form

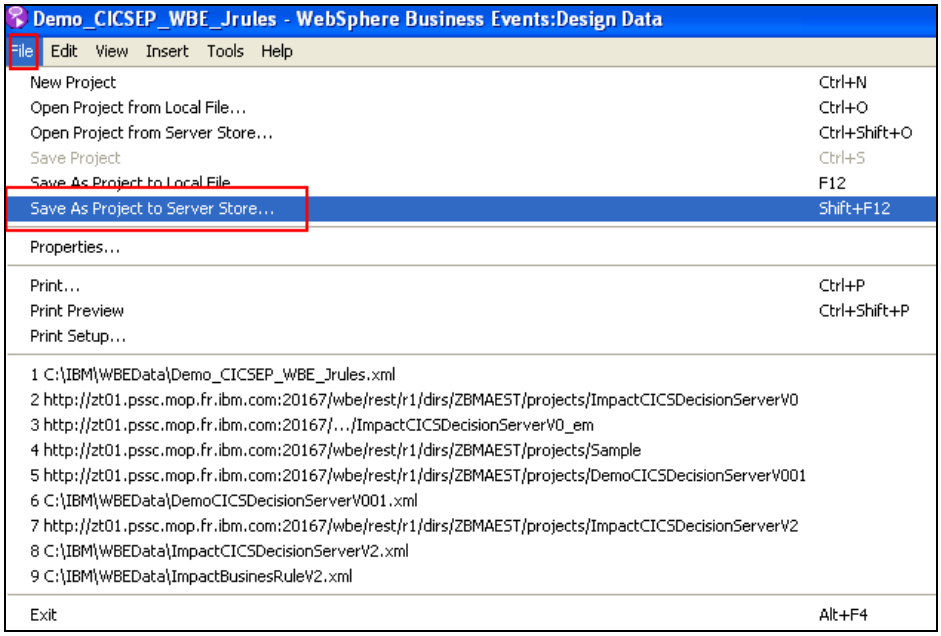


2. The Email data will look like this:

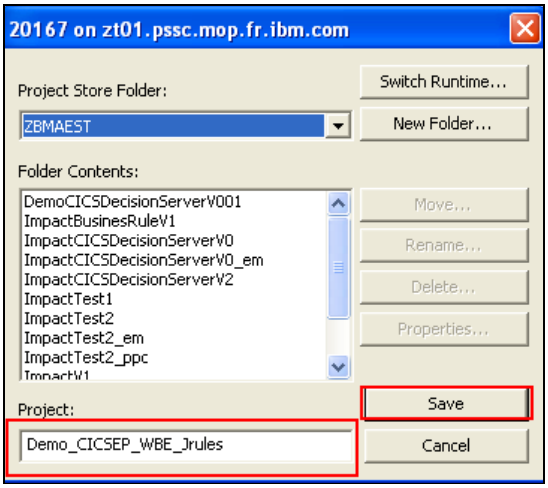


11. Save work in the Server Store ...

From the **File** menu select **Save As Project to Server Store...**



In the **Project**: field **type Demo_CICSEP_WBE_JrulesVxx** (where "**xx**" is your team number to substitute) and click **Save**



4.2 Defining Basic Interaction Sets

WebSphere Business events provides an environment in Business Space to allow Business users to quickly decide what actions to take in response to what situations and events, In this task you will use Business Space to define some simple interaction sets that will allow the rules based decisions to be invoke as part of the event processing.

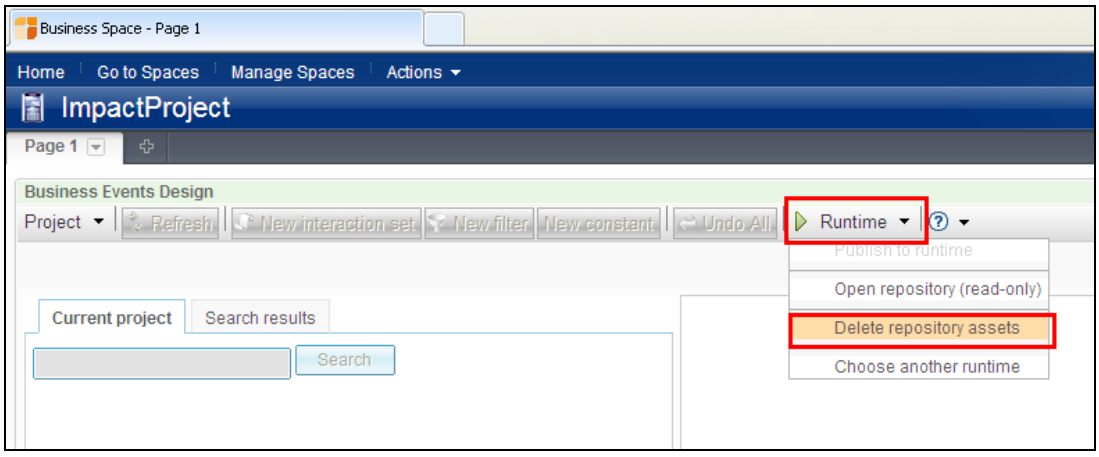
4.2.1 Open Business Space

WebSphere Business events provides an environment in Business Space to allow Business users to quickly decide what actins to take in response to what situations and events, In the step you will

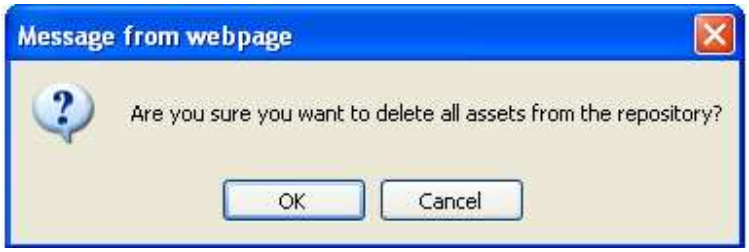
To launch Business Space:

1. Open a browser at url https://zt01.pssc.mop.fr.ibm.com:20168/mum/resources/bootstrap/login.jsp

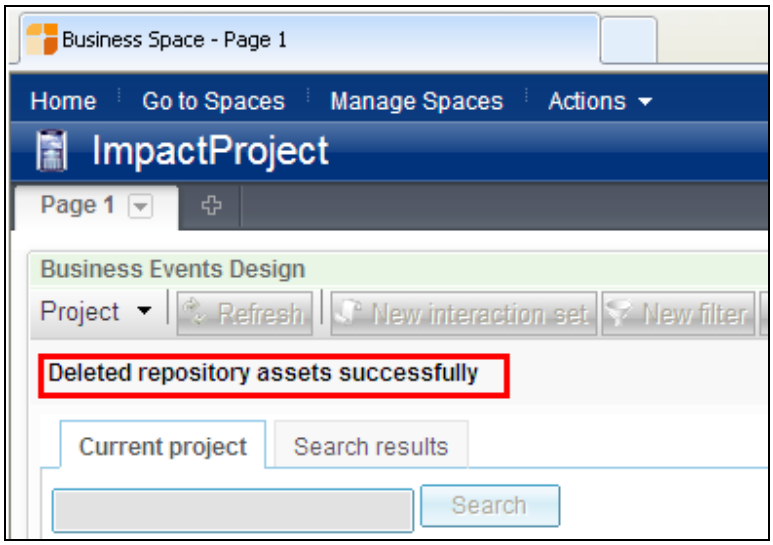
2. Login with User **ZBMAEST** password **zbmaest** and navigate to the Business Events Design tab.



Click on Runtime ==> Delete repository assets to recycle your business space

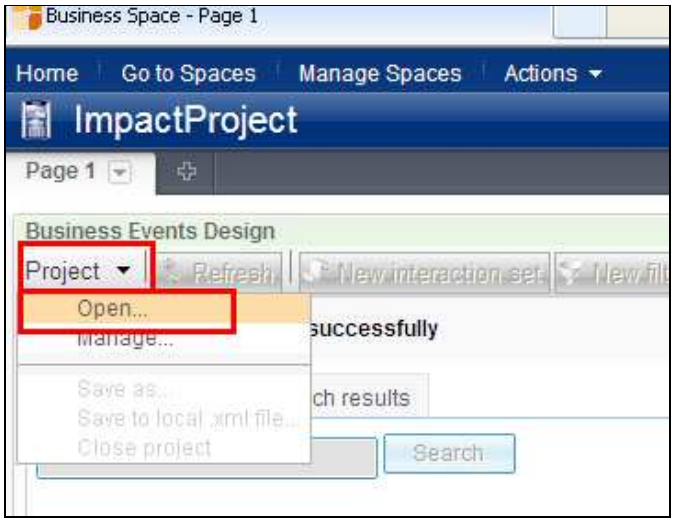


Click **OK.**

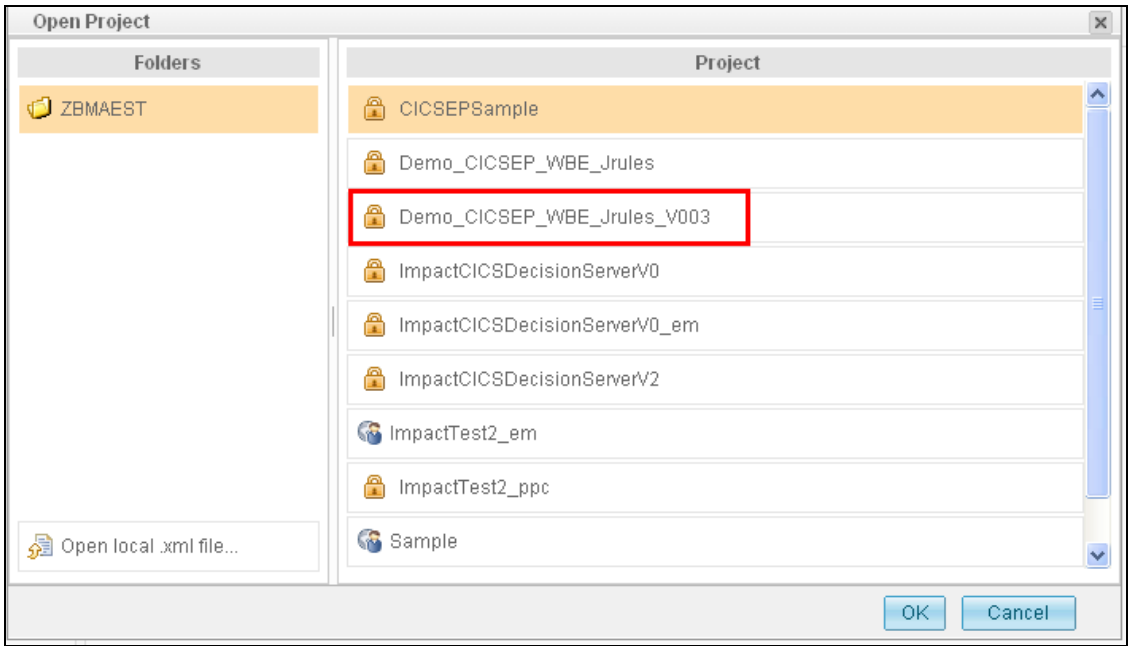


You will see this message will mean your business space is recycle to design a new business.

3. Select **Project > Open**



and then select the **Demo_CICSEP_WBE_Jrules_V0xx** where "xx" is your team number Project from the wbeAdmin folder.



4. Type **OK** and examine the structure of the project created in **Design Data.**

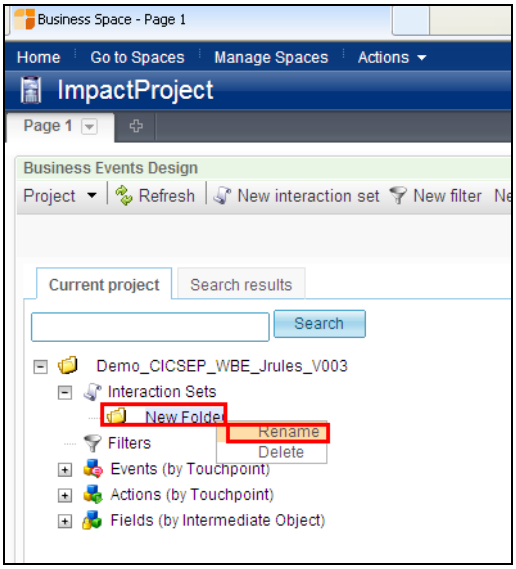
4.2.2 Define the interaction sets

During this step you will define interactions that will invoke the Rule decision service in response to a CICS Application event, and then send the response to an Offer Loan Action.

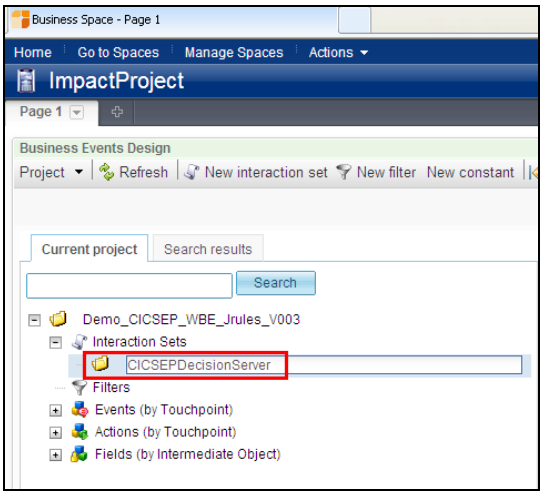
1. First we will define a folder to store the interactions

Right click on the Interaction Sets and Click Create Folder .

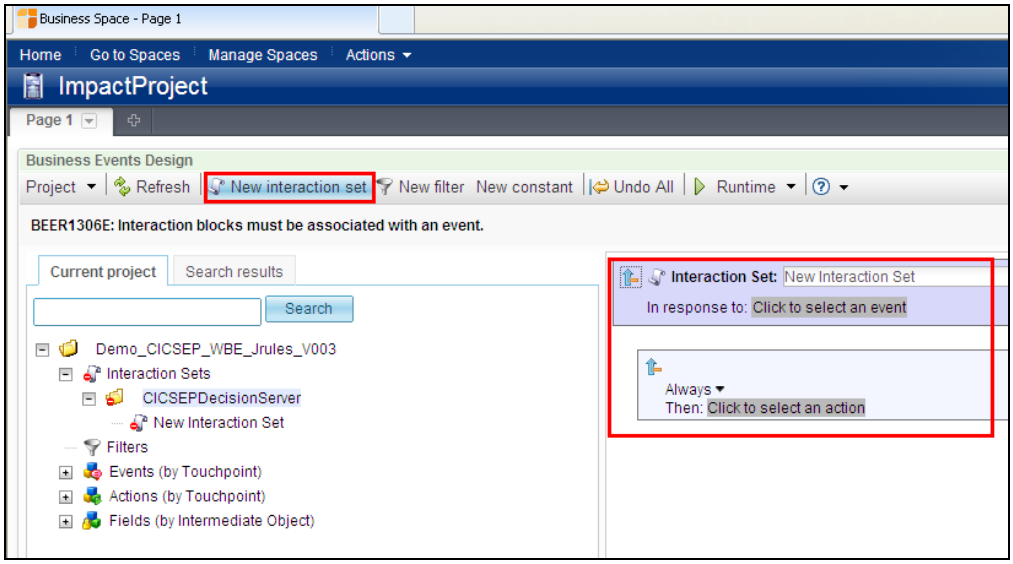
Right click on the new Folder and Rename



2. Set the Interaction Set name field to **CICSEPDecisionServer**.

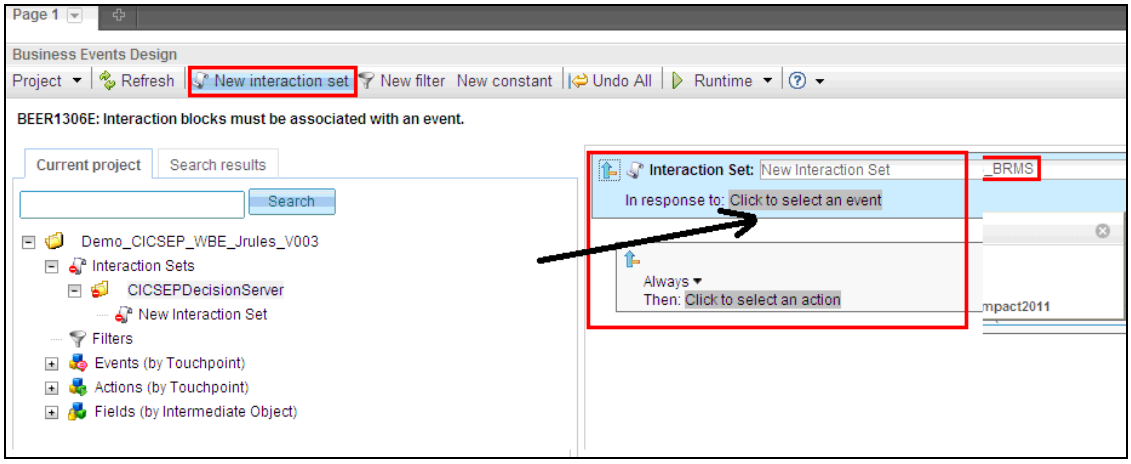


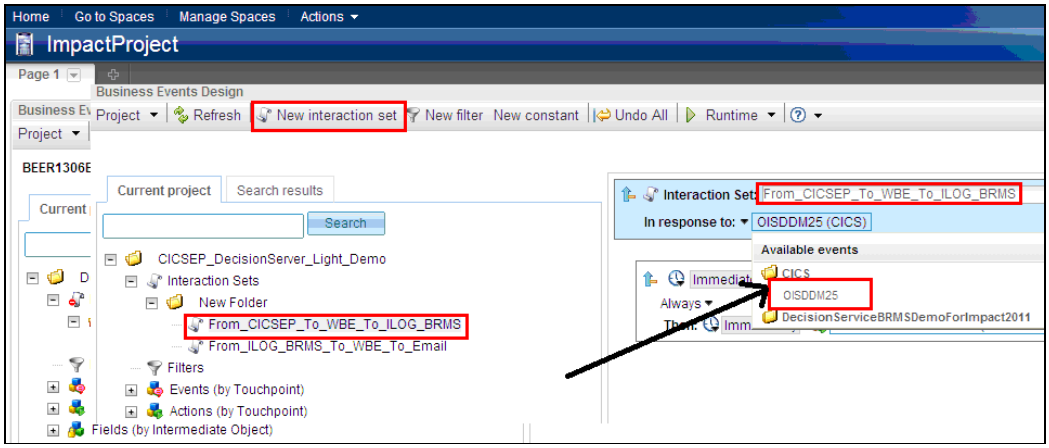
3. Now define the interaction to invoke the decision service. Click the **New Interaction Set** button.



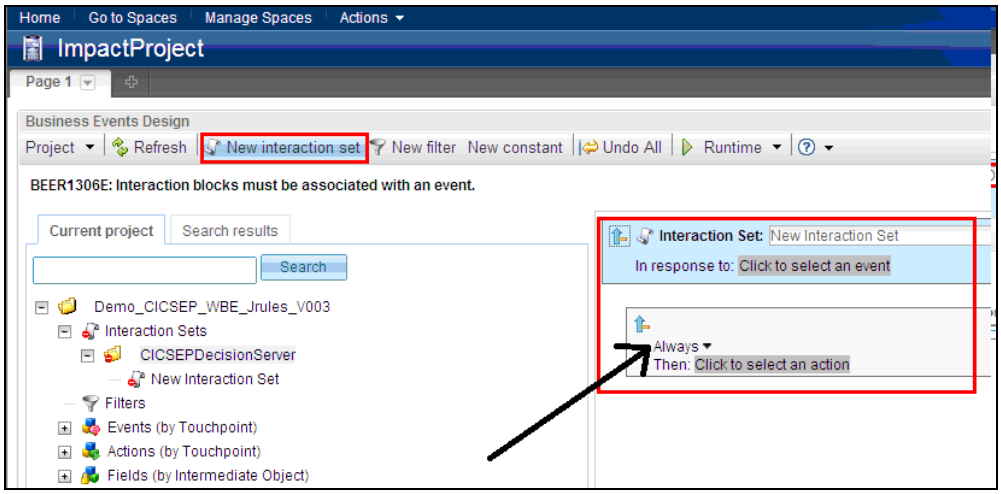
Name your Interaction Set (Ex: From_CICSEP_To_WBE_To_ILOG_BRMS)

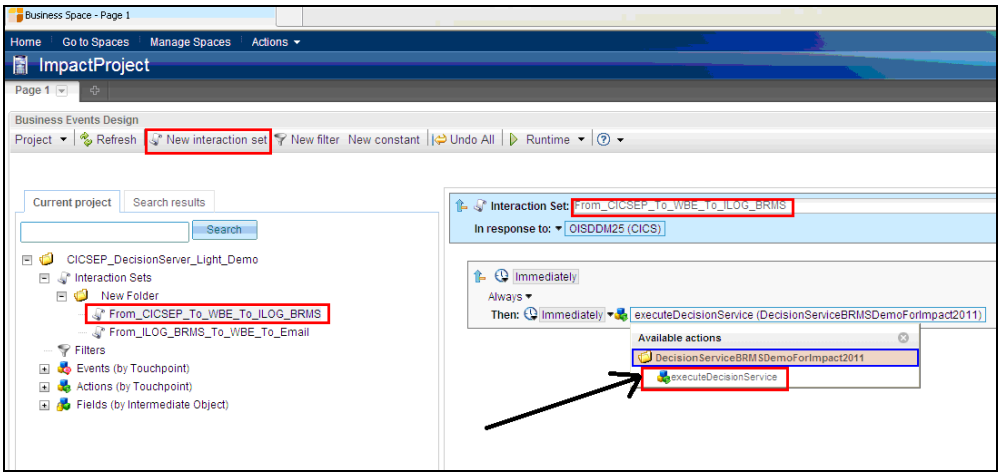
4. Click "**Click to select event"** link and then select the **CICS > OISDDM25** event.



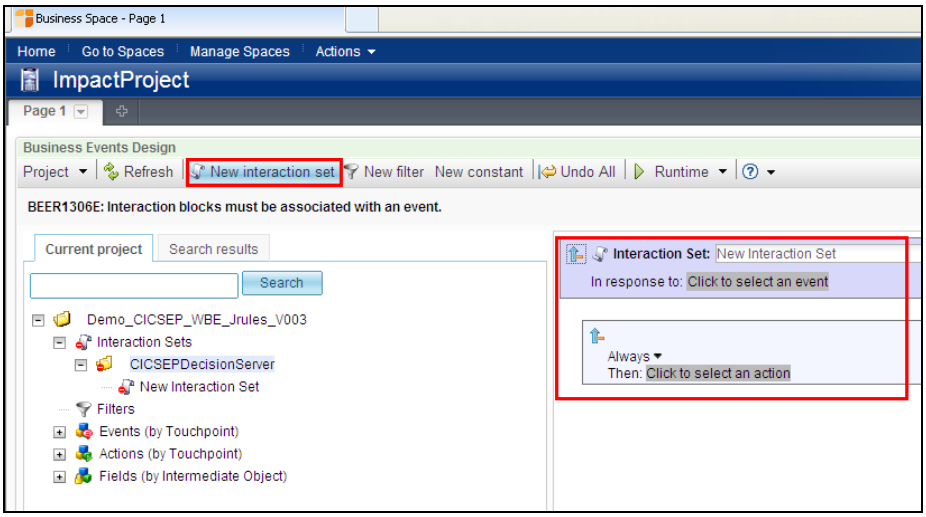


5. Click **click to select an action** link and then select the **DecisionServiceCICS_DS_rulesxx > executeDecisionService** action.

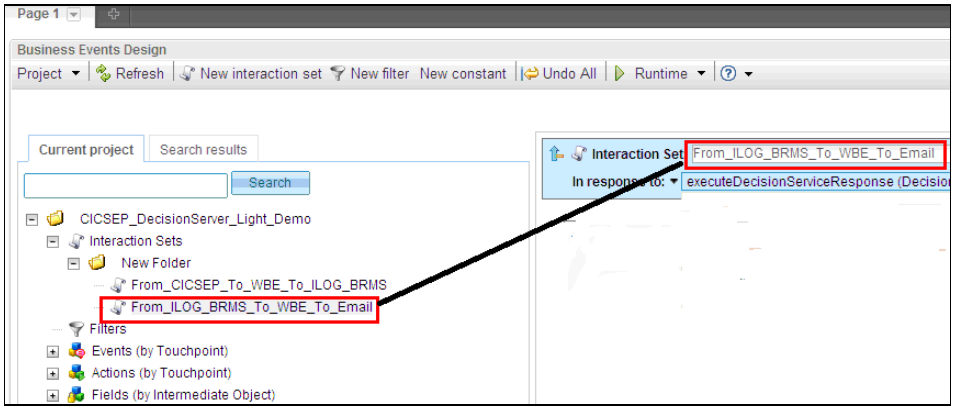




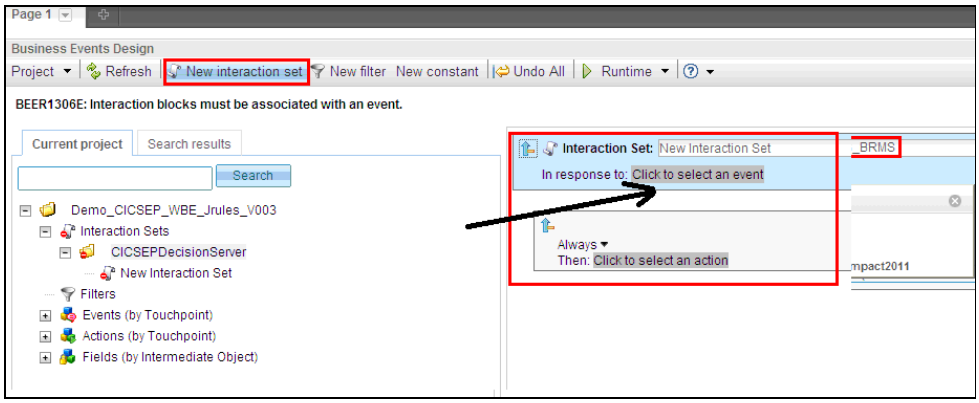
6. Now you will define the interaction to route the response from the decision service to an action. Click the **New Interaction Set** button.

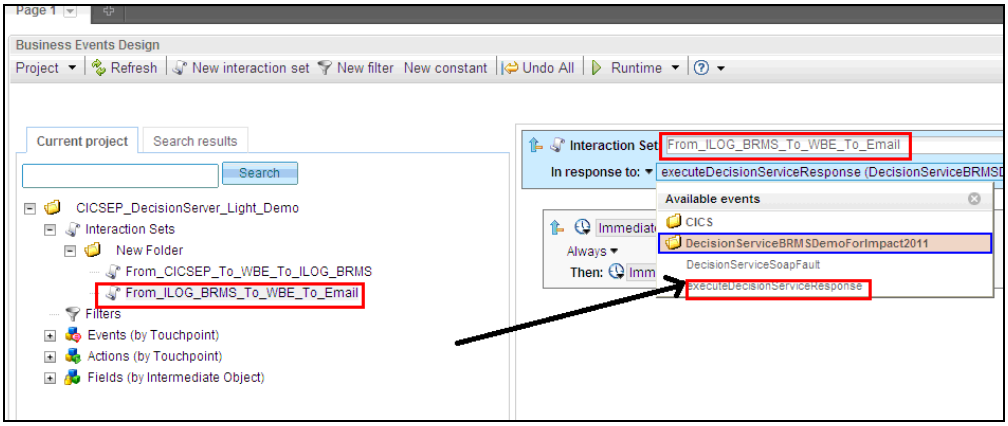


7. Set the Interaction Set name field to : From_ILOG_BRME_To_WBE_To_Email.

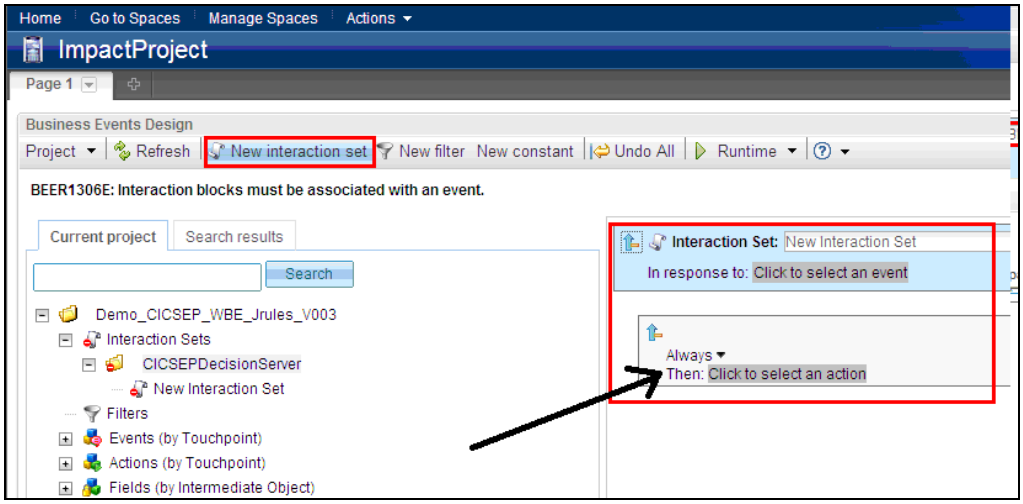


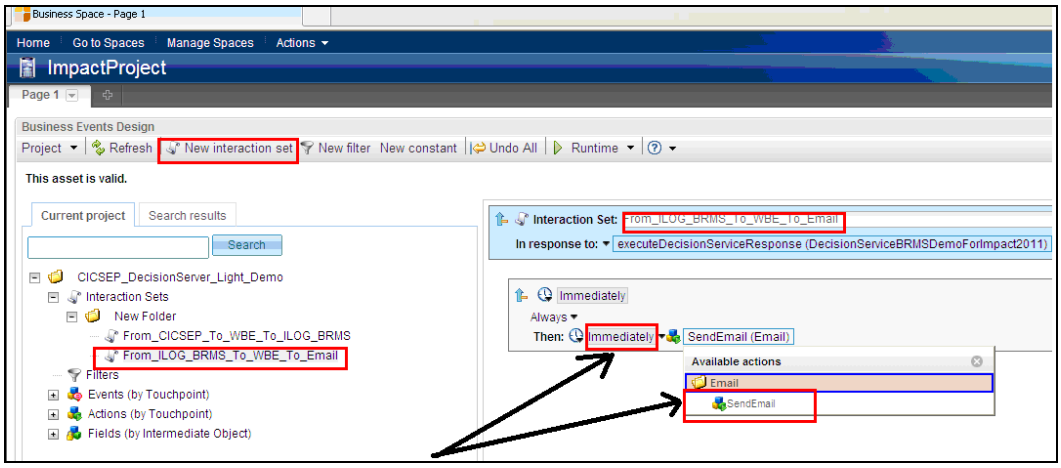
8. Click **click to select event** link and then select the **DecisionServiceCICS_DS_rulesxx > executeDecisionServiceResponse** event.



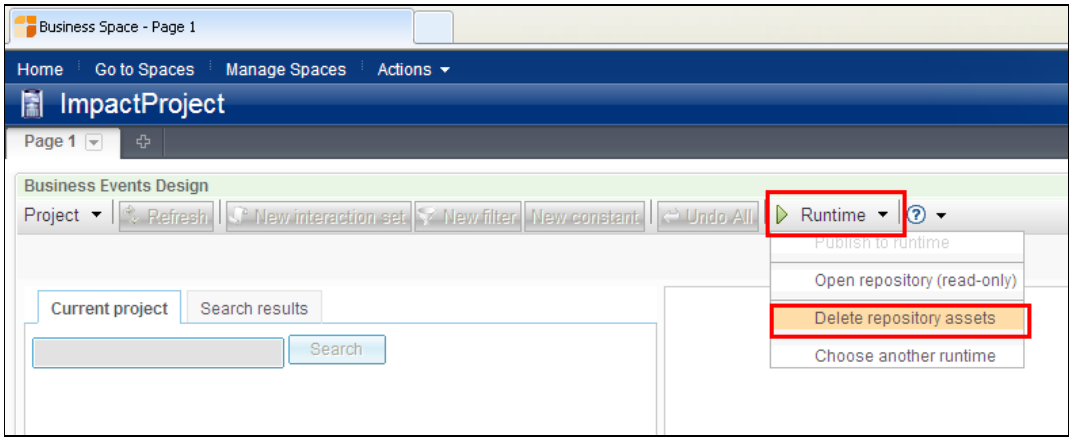


9. Click **click to select an action** link and then select **Email > sendEmail** action.

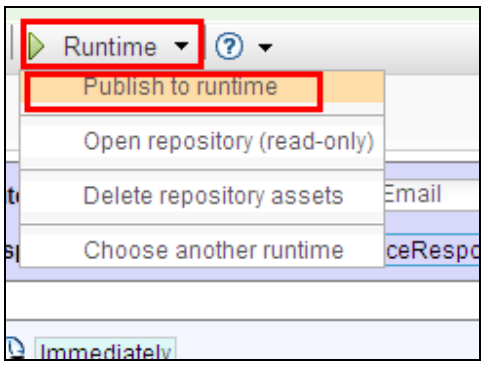




10. Select **Runtime > Delete repository assets** to clear any previous deployments



11. Select **Runtime > Publish to runtime** to deploy the Event project to the runtime.



12. Before you test these interactions, you must make sure that the connectors are running.

Go to TSO ==> SDSF and check job prefix ZB*

```
  Display  Filter  View  Print  Options  Help
--------------------------------------------------------------------------------
SDSF DA ZT01     ZT01     PAG  0  CPU/L/Z   1/  2/  2   DATA SET DISPLAYED
COMMAND INPUT ===>                                        SCROLL ===> CSR
NP   JOBNAME  StepName ProcStep JobID    Owner    C Pos DP Real Paging    SIO
_    ZBCONN   CONNECT           JOB08648 F058971  A LO  FF  290    0.00   0.09
     ZBCONN3  STEP1             STC05976 F058971    LO  FF  326    0.00   0.00
     ZBCONN2  *OMVSEX           STC05508 F058971    IN  F0 160T    0.01  37.83
     ZBSR11AS ZBSR11AS BBOPASR  STC08645 ZBASRU     IN  F4 265T    0.00   0.01
     ZBDEMNA  ZBDEMNA  BBODAEMN STC08642 ZBACRU     NS  FE 7488    0.00   0.00
     ZBSR11A  ZBSR11A  BBOPACR  STC08641 ZBACRU     NS  FE  79T    0.00   0.00
     ZBCONN1  *OMVSEX           STC08034 F058971    LO  FF  388    0.00   0.00
     ZBSR11AA ZBSR11AA BBOPCRA  STC08644 ZBACRU     IN  F4 119T    0.00   0.07
```

Ask for job ZBCONN:

```
 Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF DA ZT01     ZT01     PAG  0  CPU/L/Z   1/  2/  1  LINE 1-8 (8)
COMMAND INPUT ===>                                      SCROLL ===> CSR
NP   JOBNAME  StepName ProcStep JobID    Owner    C Pos DP Real Paging    SIO
?_   ZBCONN   CONNECT           JOB08648 F058971  A LO  FF  290   0.00   0.00
     ZBCONN3  STEP1             STC05976 F058971    LO  FF  326   0.00   0.00
     ZBCONN2  *OMVSEX           STC05508 F058971    IN  F0 160T   0.00   2.24
     ZBSR11AS ZBSR11AS BBOPASR  STC08645 ZBASRU     IN  F4 265T   0.00   0.00
     ZBDEMNA  ZBDEMNA  BBODAEMN STC08642 ZBACRU     NS  FE 7488   0.00   0.00
     ZBSR11A  ZBSR11A  BBOPACR  STC08641 ZBACRU     NS  FE  79T   0.00   0.00
     ZBCONN1  *OMVSEX           STC08034 F058971    LO  FF  388   0.00   0.00
     ZBSR11AA ZBSR11AA BBOPCRA  STC08644 ZBACRU     IN  F4 119T   0.00   0.00
```

Then select STDOUT to verify the Connector availability

```
 Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF JOB DATA SET DISPLAY - JOB ZBCONN   (JOB08648)    LINE 1-8 (8)
COMMAND INPUT ===>                                     SCROLL ===> CSR
NP   DDNAME   StepName ProcStep DSID Owner    C Dest              Rec-Cnt Page
     JESJCLIN                    1 F058971  H                        14
     JESMSGLG JES2               2 F058971  H                         4
     JESJCL   JES2               3 F058971  H                        15
     JESYSMSG JES2               4 F058971  H                         0
     $INTTEXT JES2               5 F058971  A                         6
     $JOURNAL                    6 F058971  A                        31
s_   STDOUT   CONNECT          101 F058971  H                         0
     STDERR   CONNECT          102 F058971  H                         0
```

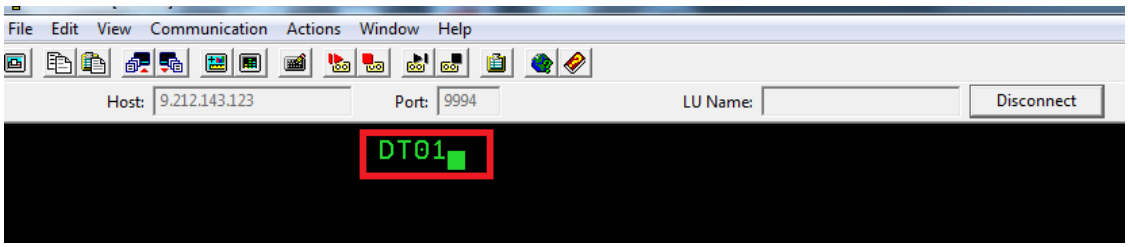
You will see these messages showing the connector ready to monitor ILOG Rule execution method "executeDecisionService" and the eMail action sending.

```
 SDSF OUTPUT DISPLAY ZBCONN   JOB08648  DSID   101 LINE 319     COLS 02- 81
 COMMAND INPUT ===> _                                          SCROLL ===> CSR
BEER4675I: Recovering the transaction log from job queue executeDecisionServiceQ
BEER4675I: Recovering the transaction log from job queue executeDecisionServiceQ
BEER4674I: Reading log file /wasv70config/zbcell/zbwbe1a/queues//executeDecision
BEER0637I: Starting one action worker
BEER0632I: The SOAP module is monitoring action: executeDecisionService
BEER0631I: The SMTP module is monitoring action: SendEmail
BEER0616I: The JMS module is monitoring event: OISDDM25
BEER0639I: The Connector reload is complete
```

4.2.3 Test the all scenario from CICS

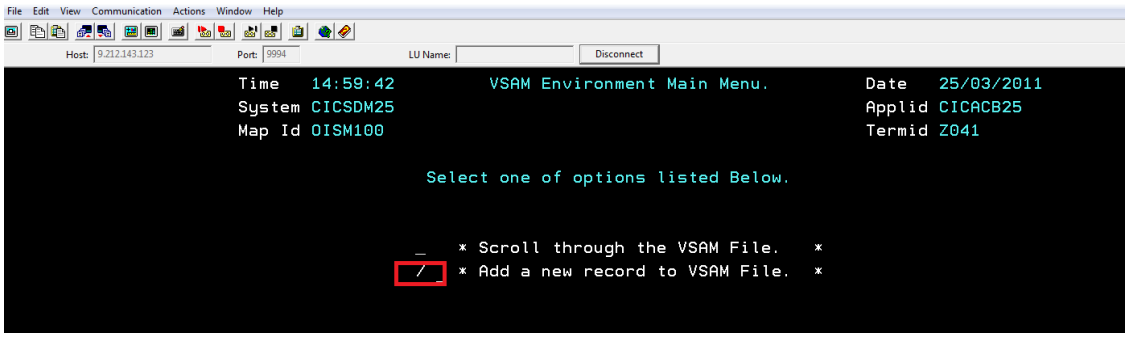
In this step you will test an event arriving from CICS Application, Capturing the event, invoking the decision service and sending an eMail as described in the previous step.

1. Connect to CICS Application and run DT01 transaction.



2. You will obtain this menu.

Set ***/*** on "*Add a new record to VSAM File*"



3. Fill values for fields :

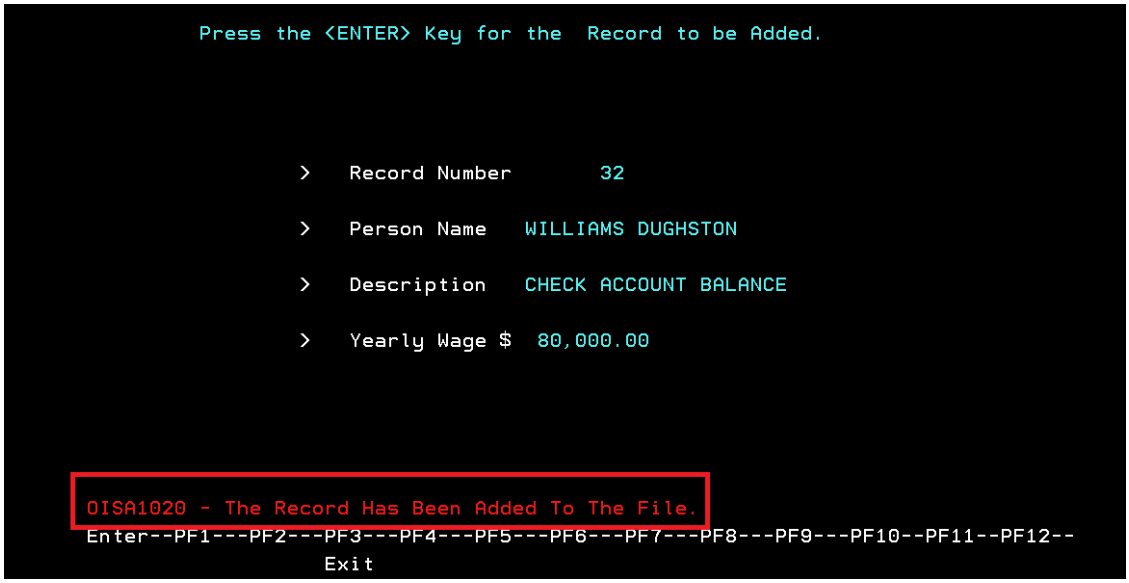
   Person Name

   Description

   YearlyWage

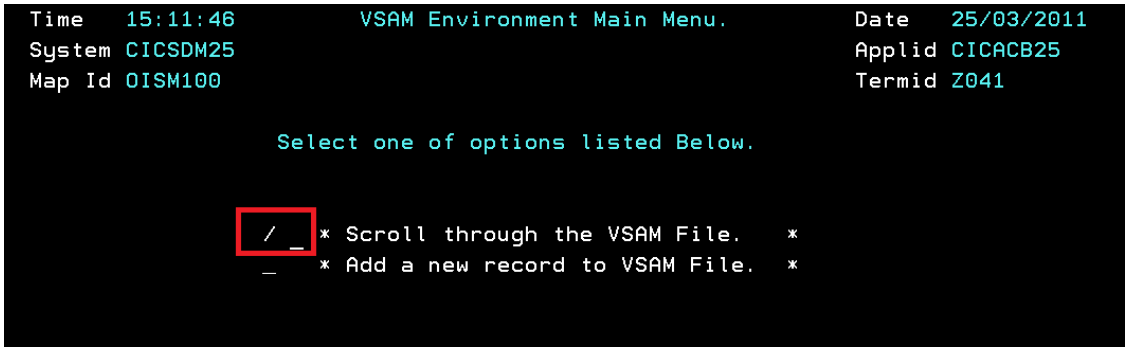
And hit "Enter"

You will obtain this message "*OISA1020 - The Record Has been Added To The File*"



Hit "PF3" key to return to main menu

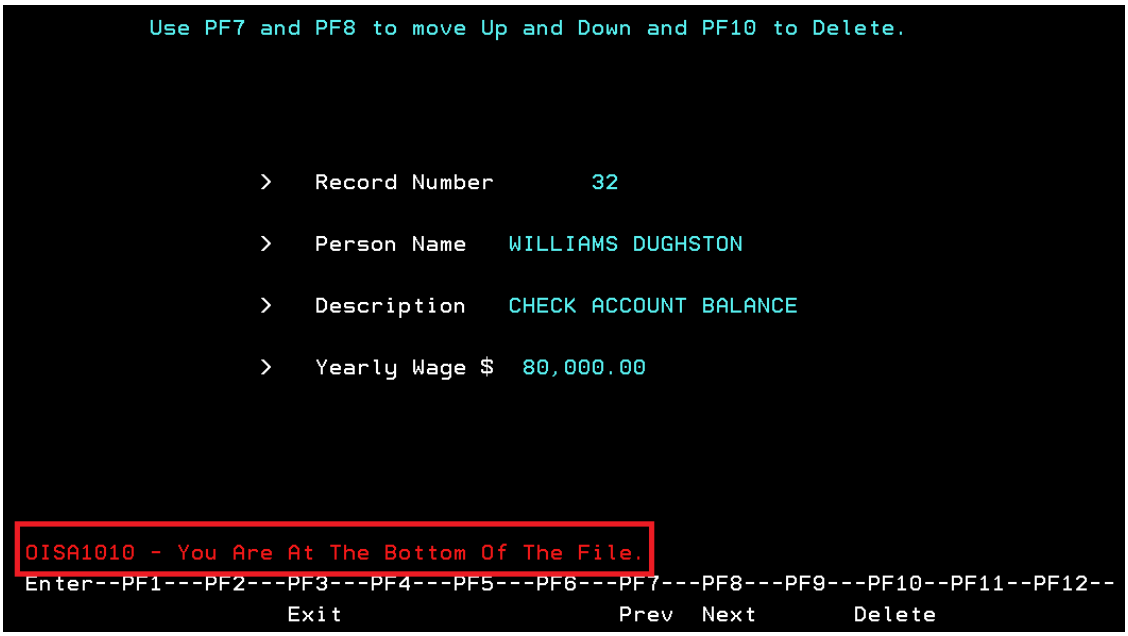
4. Now select "Scroll through the VSAM File" as shown below



5. Browse the record by hitting the "**PF8**" key until the end of the record on the VSAM.

At the end, you will obtain this message : "*OISA1010 - You Are At The Bottom Of The File*"

The current record will be the one you create.



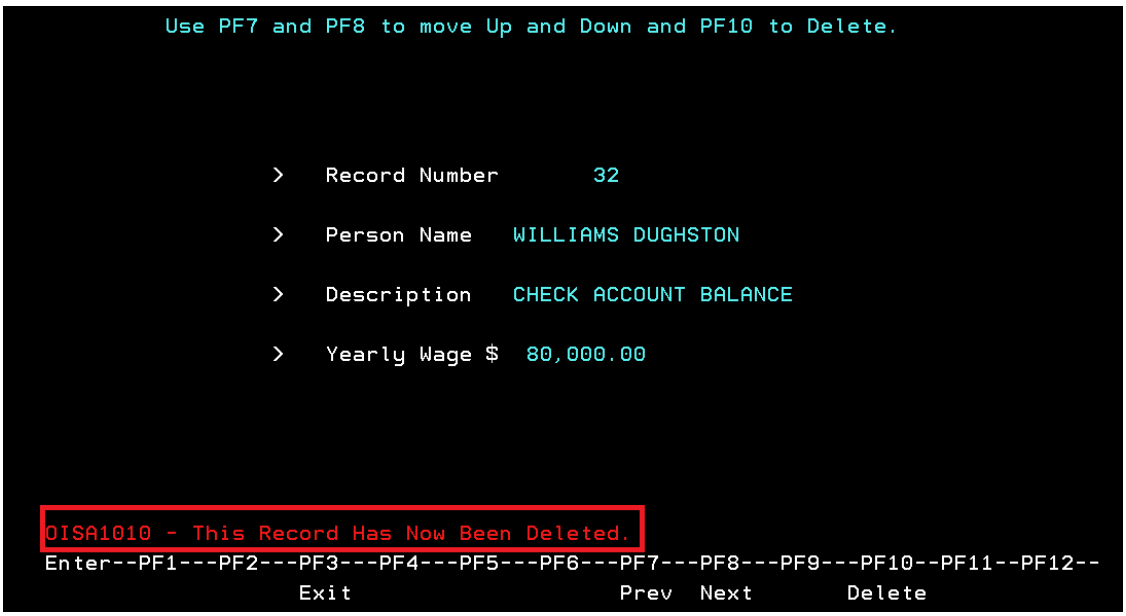
6. Delete the current record.

Hit the PF10 key to delete your record.

By this action, CICS Event will capture and format the event of Record deletion to WBE.

Then WBE will ask to Jrules to decide action.

Return from Jrules, WBE will merge data and Send Email.

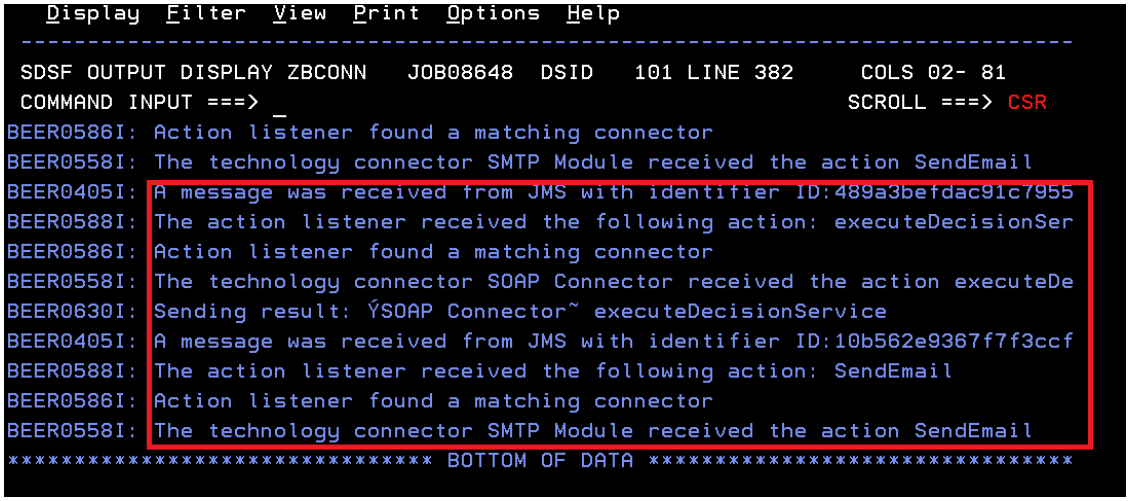


By hitting "**PF10**" key, you obtain this message : "*OISA10110 - This Record Has Now Been Deleted.*"

Now , go to TSO SDSF to check the connector

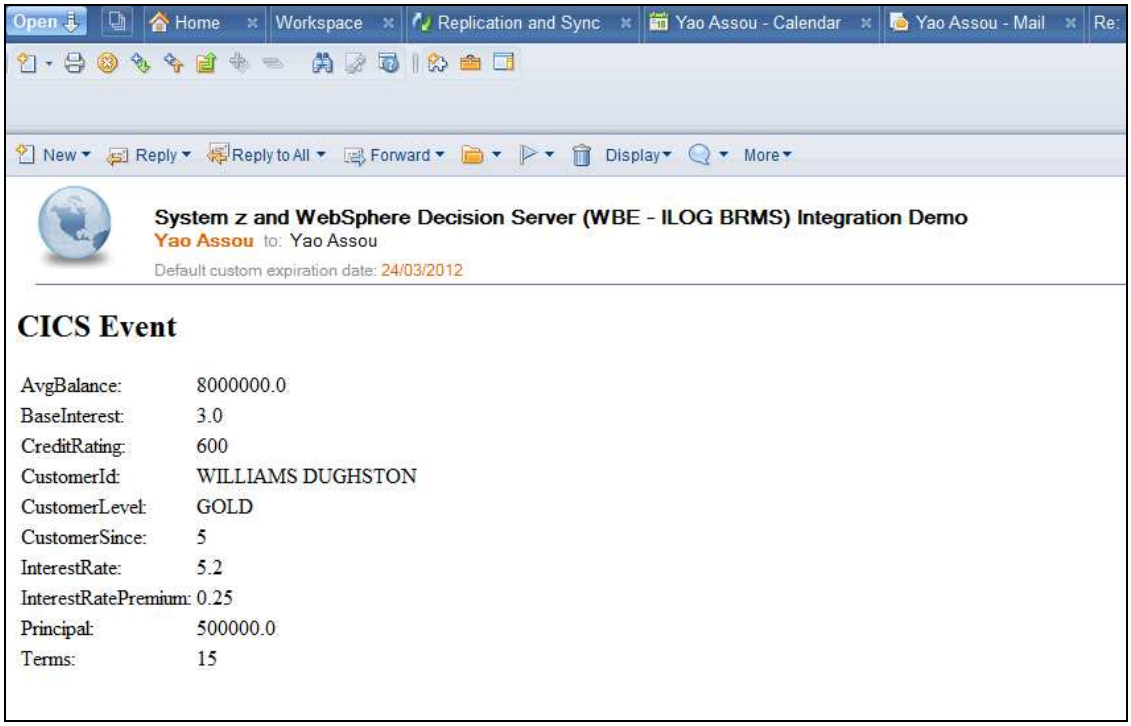
Connector will receive theses messages on outpout.

Message "*The technology connector SMTP Module receive the action SendEmail*" mean everything is ok.



7. Check now your email.

.



4.3 Summary

Congratulations! You have completed the IBM CICS Event Processing and WebSphere Decision Server integration scenario.

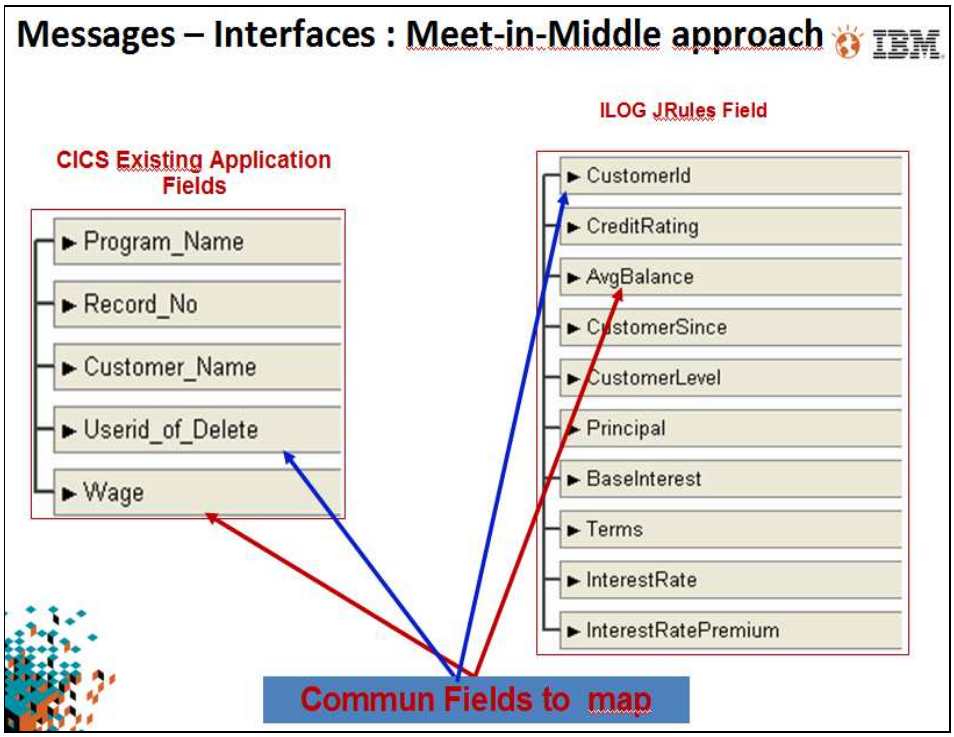
During this lab you became familiar with all the Decision Server modules:

- **Rule Studio** to design and develop the business rule application.
- **Design Data** to design and develop the business event application.
- **Rule Team Server** for business users to manage the Rule Based Decisions.
- **Rule Execution Server** to execute and monitor the business rules.
- **Business space** to manage your event project by creating a new business process

We hope that this lab helped you understand how you can use IBM WebSphere Decision Server to externalize the business logic from your own application and place it in the hands of the business users.

Lab 5 Taking a Context and Sharing Data between events

In this part we will use the business event processing provided with decision server to integrate the decision into the application and to share some data coming from CICS and not necessary to make ILOG Jrules Decision but needed to put on the email data.



Example :

Data from CICS Event :

Fields	Format
Programe_Name	String
Record_Number	Real
Customer_Name	String
Userid_Of_Delete	String
Wage	Real

Input / Output Data needed for Jrules

Fields	Format
CustomerId	String
CreditRating	Integer
AvgBalance	Float
CustomerLevel	String
Principal	Float
BaseInterest	Float
Terms	Integer
InterestRate	Float
InterestRatePremium	Float

Mapping fields between CICS Data and Jrules interfaces

CICS Data Fields	ILOG Jrules Fields	Format
Customer_Name	CustomerId	String
Wage	AvgBalance	Real

Now Data needed on eMail Data :

Fields	Format
Programe_Name	String
Record_Number	Real
Userid_Of_Delete	String
CustomerId	String
CreditRating	Integer
AvgBalance	Float
CustomerLevel	String
Principal	Float
BaseInterest	Float
Terms	Integer
InterestRate	Float
InterestRatePremium	Float

So these Data must be hold between each event in WBE process.

Fields	Format
Programe_Name	String
Record_Number	Real
Userid_Of_Delete	String

5.1 Update your Event Project

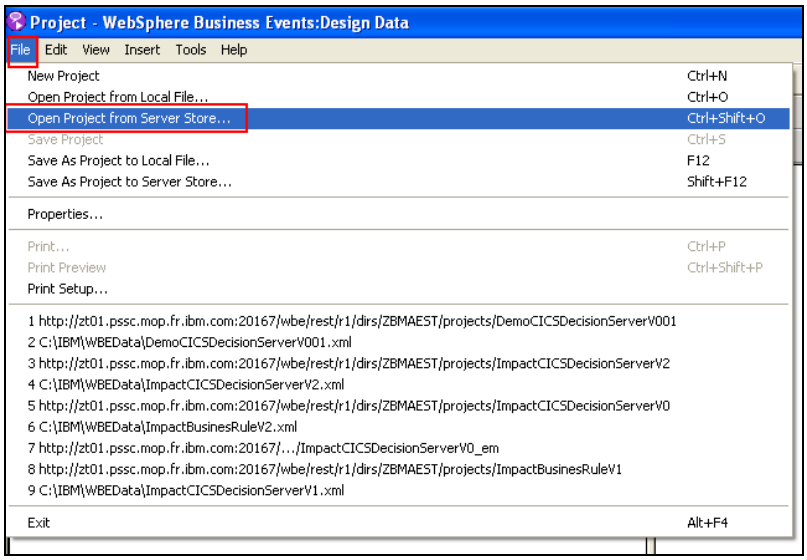
5.1.1 Open Design Data

To launch Design Data:

12. From the **Start** menu, click **Start > All Programs > IBM WebSphere Business Events V7.0.1 > Design Data**.

13. Design Data opens with an empty project.

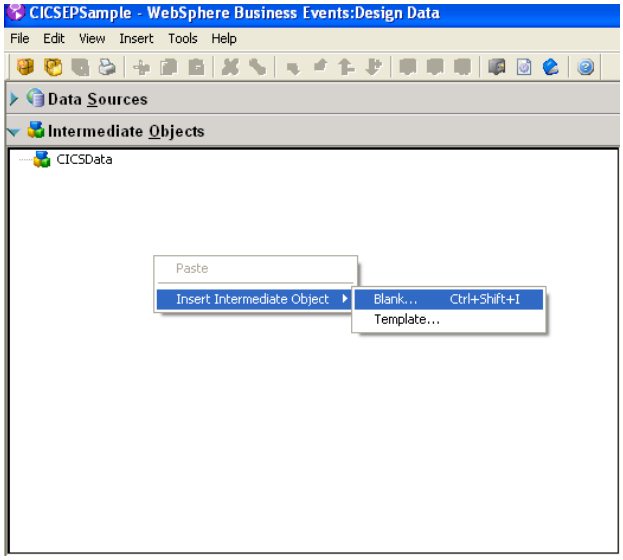
14. From the **File** menu select Open Project from Server Store



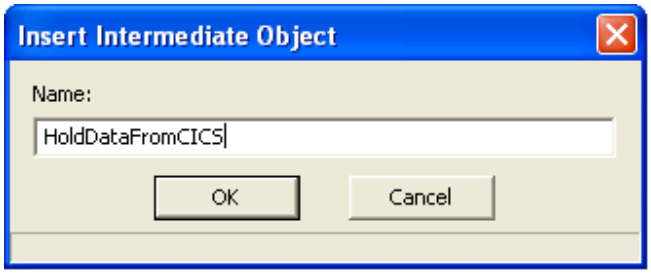
15. Click the Runtime... button to configure WBE Administration location on Montpellier z/OS plateform. URL : **zt01.pssc.mop.fr.ibm.com:20167**

16. In the **Folder Contents**, select your Project **Demo_CICSEP_WBE_JrulesVxx** project and click **Open**.

17. Right-click on Intermediate Object form pane and select Insert **Intermediate Object --> Blank**

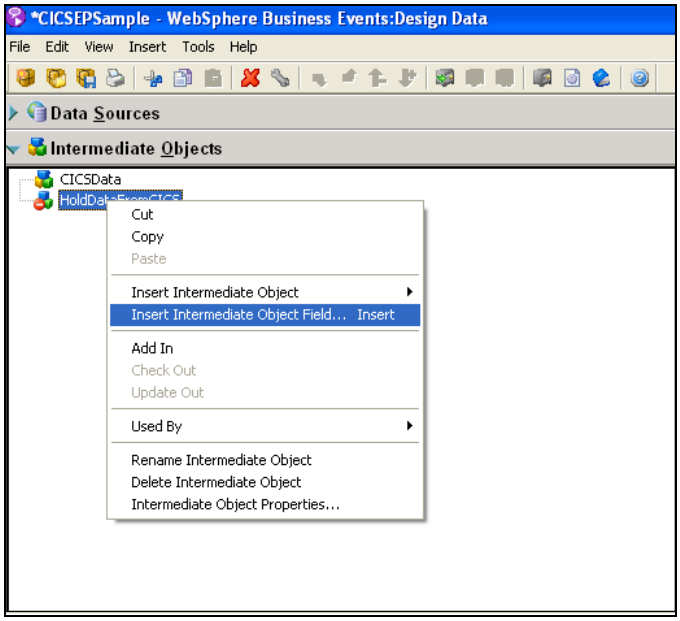


18. In the pane, type a Name of the Intermediate Object (Ex : HoldDataFromCICS) and click **OK**.



19. Compose fields

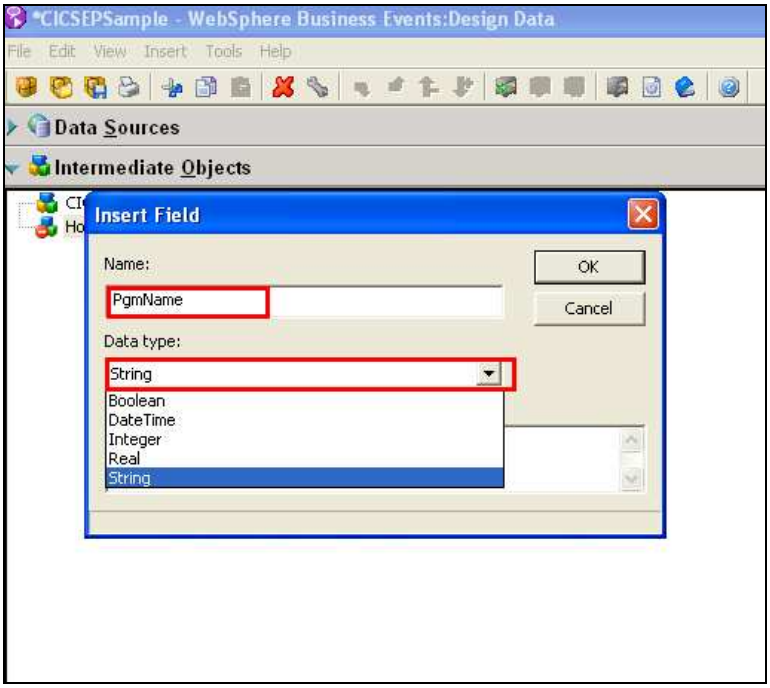
Right-click on the intermediate object you just create (HoldDataFromCICS) and **select Insert Intermediate Object Field... Insert**



Type **PgmName** in the *Name* field

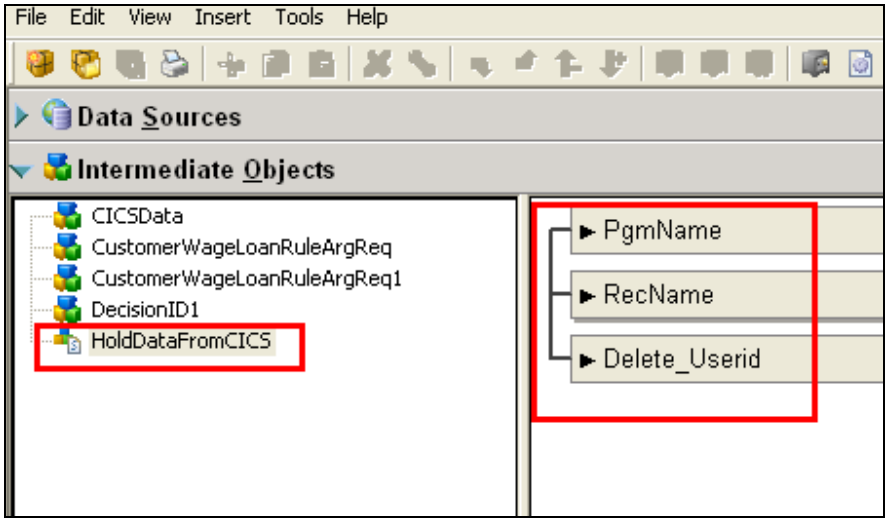
Select **String** in the Data type Combo Box

Click **OK**.



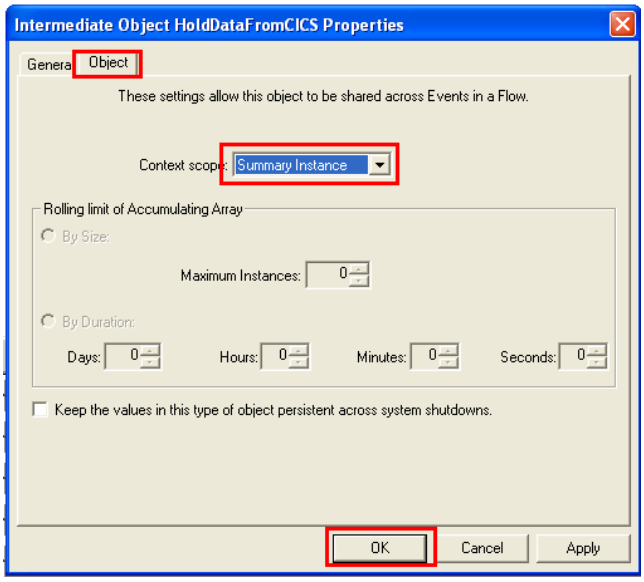
20. Create 2 more fields

Name	Type
Delete_Userid	String
RecNumber	Real



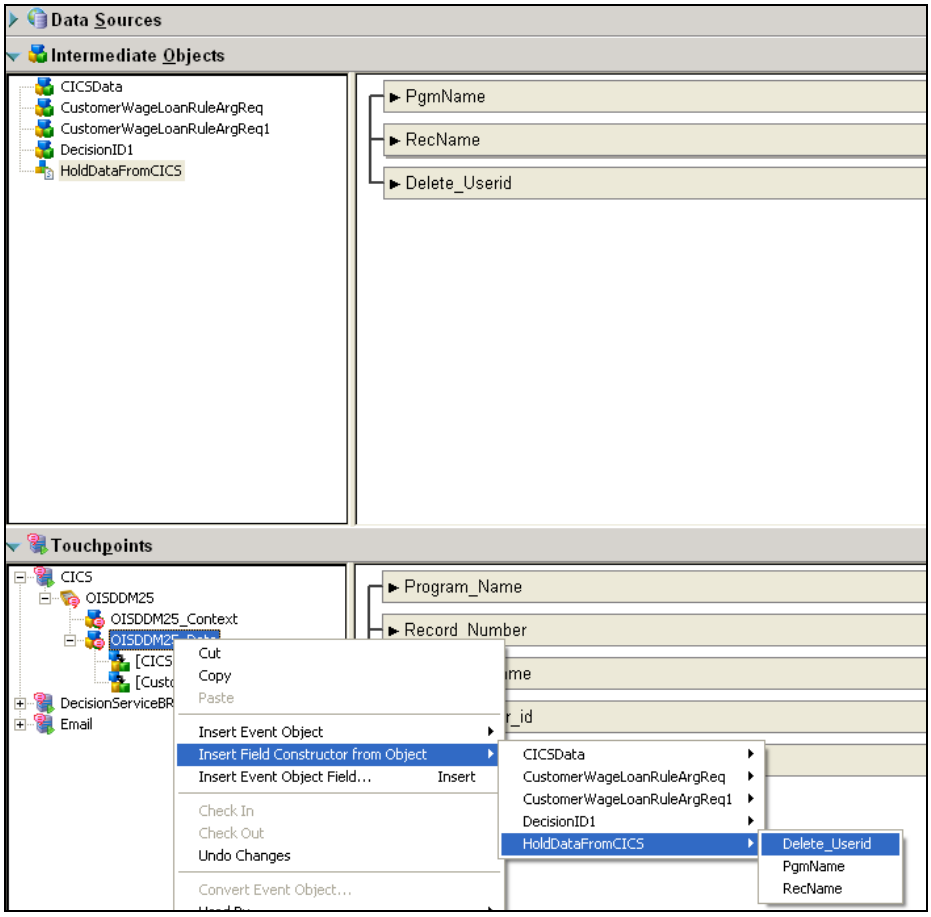
Right click on the Intermediate object HoldDataFromCICS and the properties and in Object tab, set the Content scope to **Summary Instance**

Click **OK** .

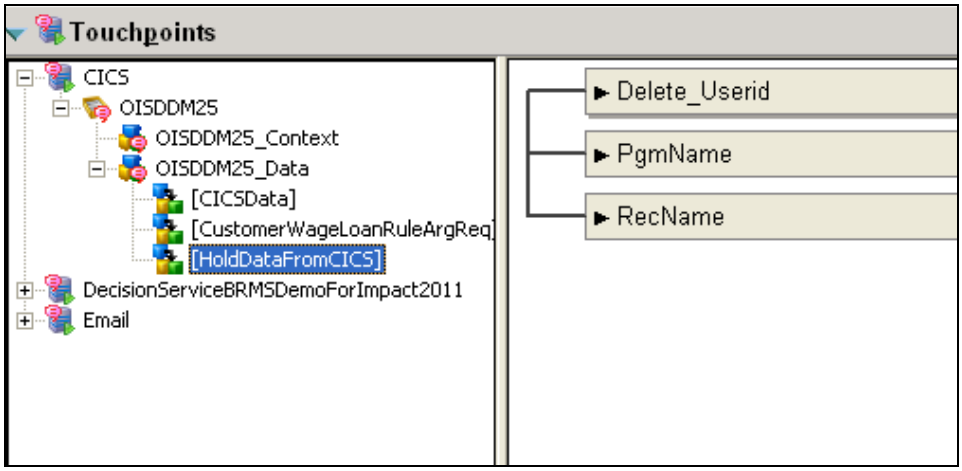


21. Now you are going to map the **HoldDataFromCICS** object onto an existing **HoldDataFromCICS** intermediate object. Right click the **HoldDataFromCICS** object and select **Insert Field Constructor from Object > HoldDataFromCICS > Delete_Userid**. This maps the returned HoldDataFromCICS **Delete_Userid** field into the HoldDataFromCICS intermediate object. In this case we have only created one field so need to repeat this for each of the other fields in the loan object.

Right click HoldDataFromCICS **> [**HoldDataFromCICS **]** and select **Insert Field Constructor > PgmName**
Right click HoldDataFromCICS **> [**HoldDataFromCICS **]** and select **Insert Field Constructor > RecNumber**



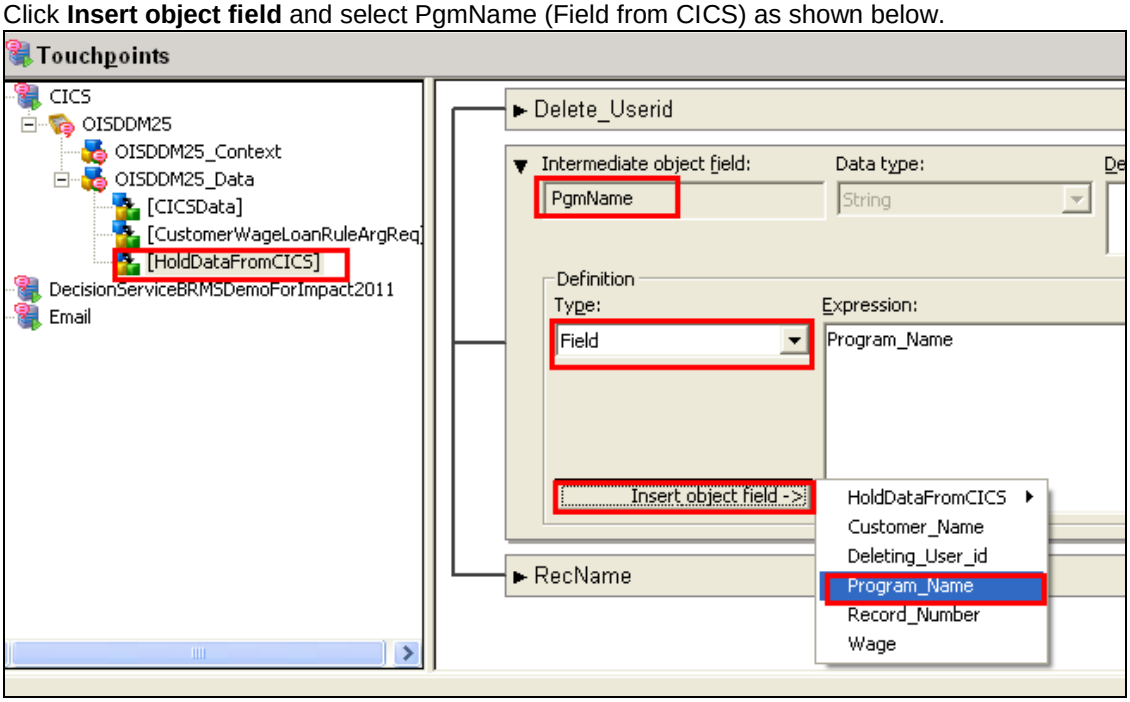
Your screen will look like this :



22. When this approach is taken it is possible to adopt different approaches for how each field is mapped. We simply need to adopt the direct field mapping approach so select the HoldDataFromCICS > [HoldDataFromCICS] intermediate object constructor and open up the twistie for the PgmName field.

In the Definition Type field select **Field** from the pulldown.

Click **Insert object field** and select PgmName (Field from CICS) as shown below.



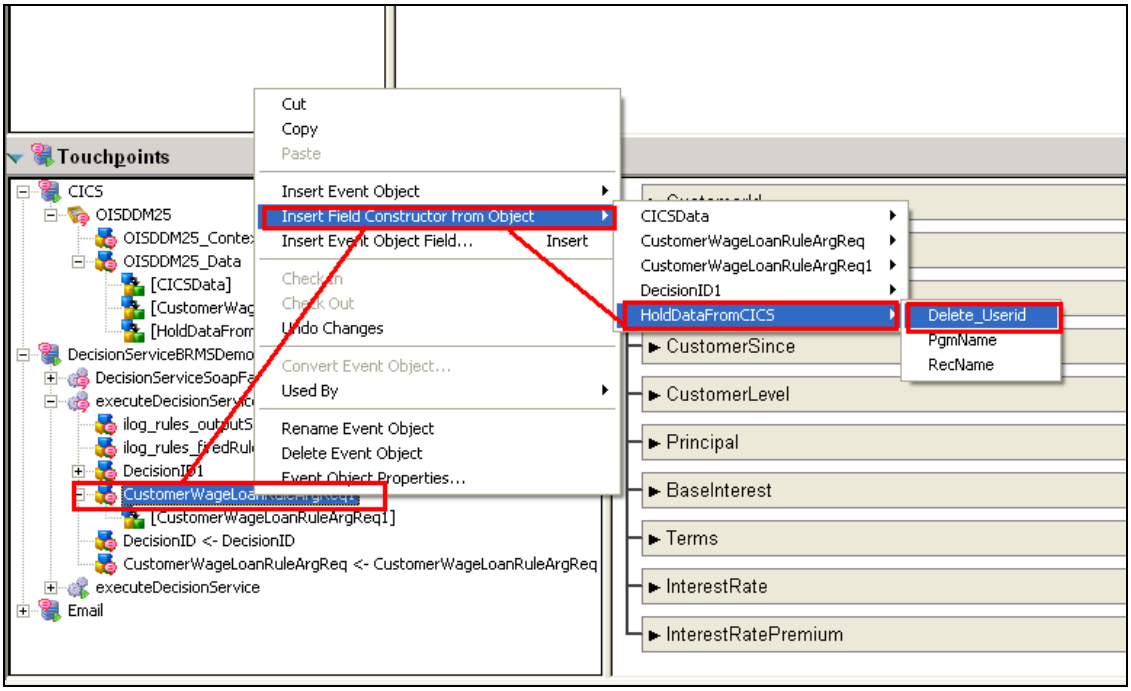
Repeat this process for all the other fields in the object.

Delete_Userid	==>	Deleting_User_id
RecNumber	==>	Record_Number

You should not need to set the Definition Type pulldown for each field.
Save your work and and the error indicator against the HoldDataFromCICS >[HoldDataFromCICS] object should disappear.

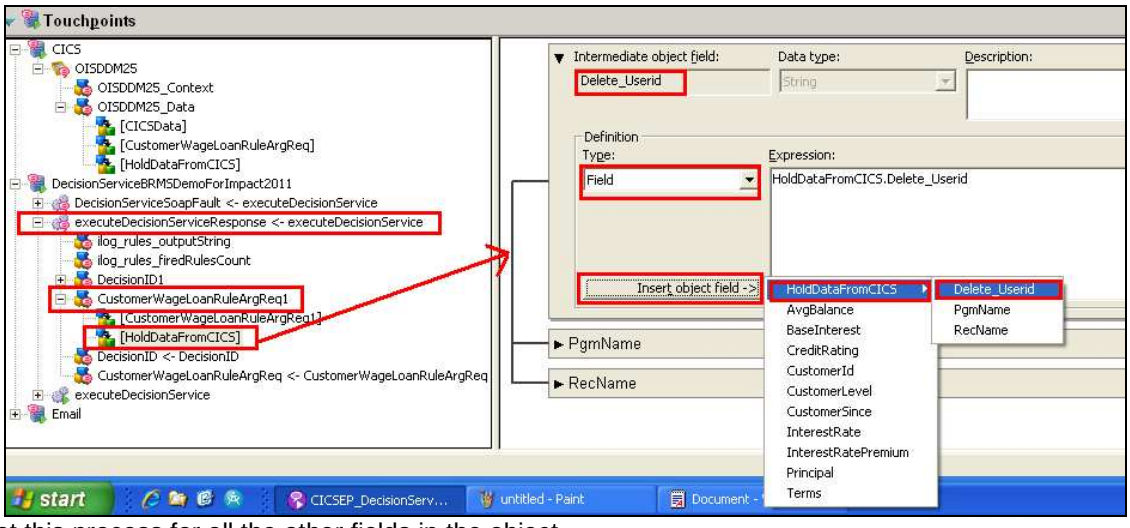
18. We need also to keep the data in context after return from Jrules. Follow these steps: We are again going to map the **HoldDataFromCICS** object onto an existing **HoldDataFromCICS** intermediate object from Dec. Right click the **executionDecisionServiceResponse** object and select **Insert Field Constructor from Object > HoldDataFromCICS > Delete_Userid**. This maps the returned HoldDataFromCICS **Delete_Userid** field into the HoldDataFromCICS intermediate object. In this case we have only created one field so need to repeat this for each of the other fields in the loan object.

Right click **executionDecisionServiceResponse** object and select **Insert Field Constructor from Object > HoldDataFromCICS > PgmName**
Right click **executionDecisionServiceResponse** object and select **Insert Field Constructor from Object > HoldDataFromCICS > RecNumber**



23. When this approach is taken it is possible to adopt different approaches for how each field is mapped. We simply need to adopt the direct field mapping approach so select the HoldDataFromCICS > [HoldDataFromCICS] intermediate object constructor and open up the twistie for the PgmName field.

In the Definition Type field select **Field** from the pulldown.
Click **Insert object field** and select PgmName (Field from CICS) as shown below.



Repeat this process for all the other fields in the object.

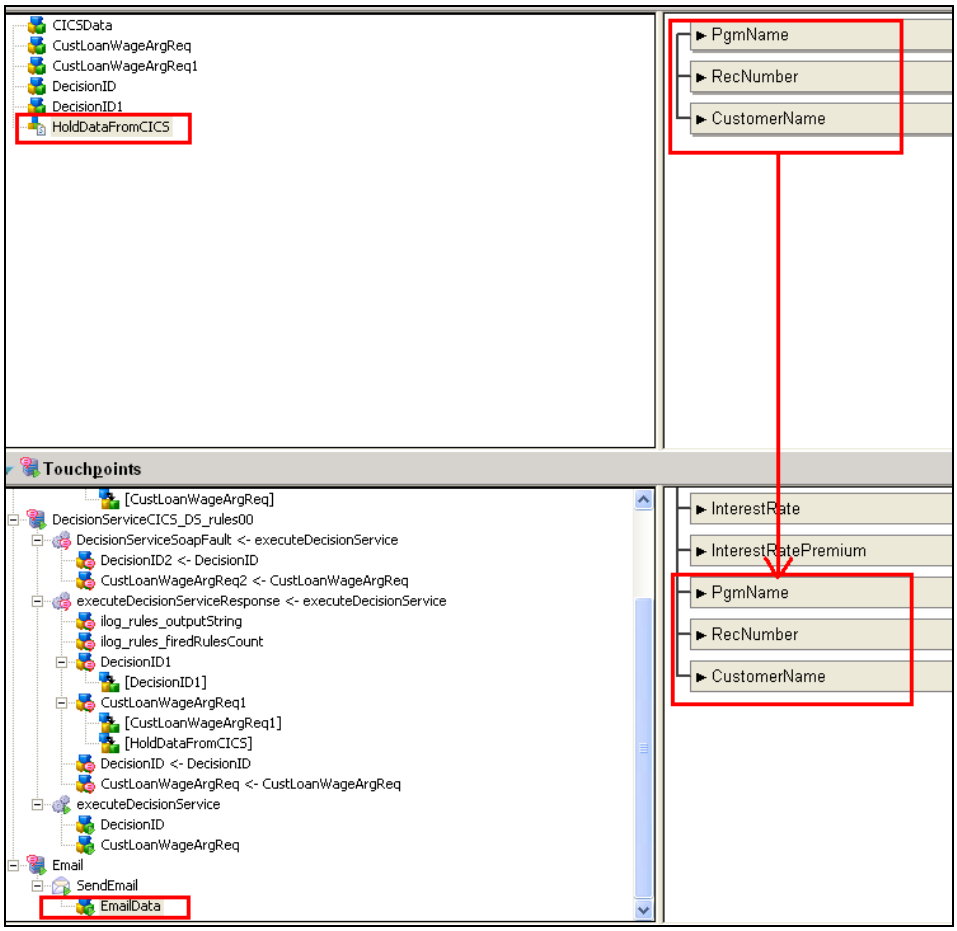
Delete_Userid ==> Deleting_User_id
RecNumber ==> Record_Number

5.1.2 Step 4 Map the intermediate objects to Email actions

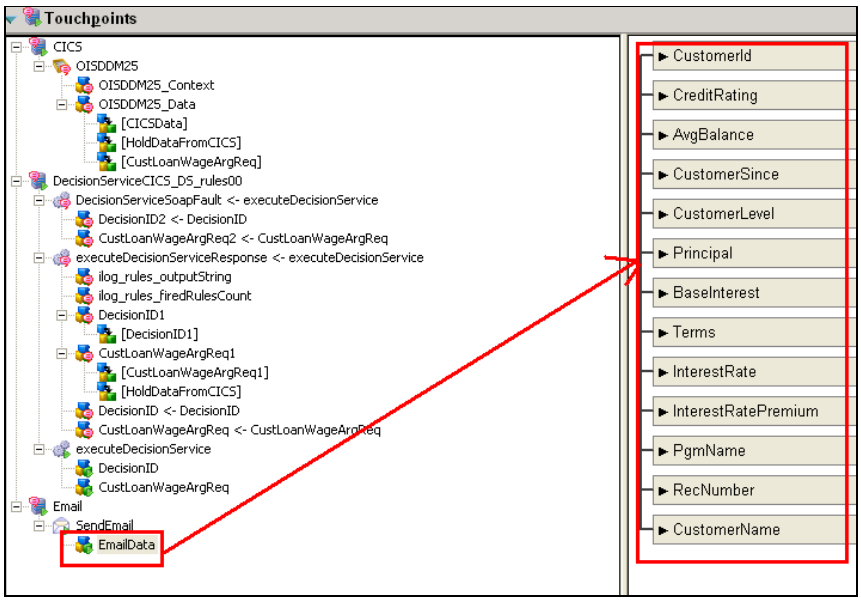
The final development step in designing the Event processing is to complete the mapping of the eMail Data by adding fields from **HoldDataFromCICS**

3. Select Email data in the TouchPoint.

   Click on the **HoldDataFromCICS** intermediate object and select the all fields in the right panel and drag it to the **eMail** data form

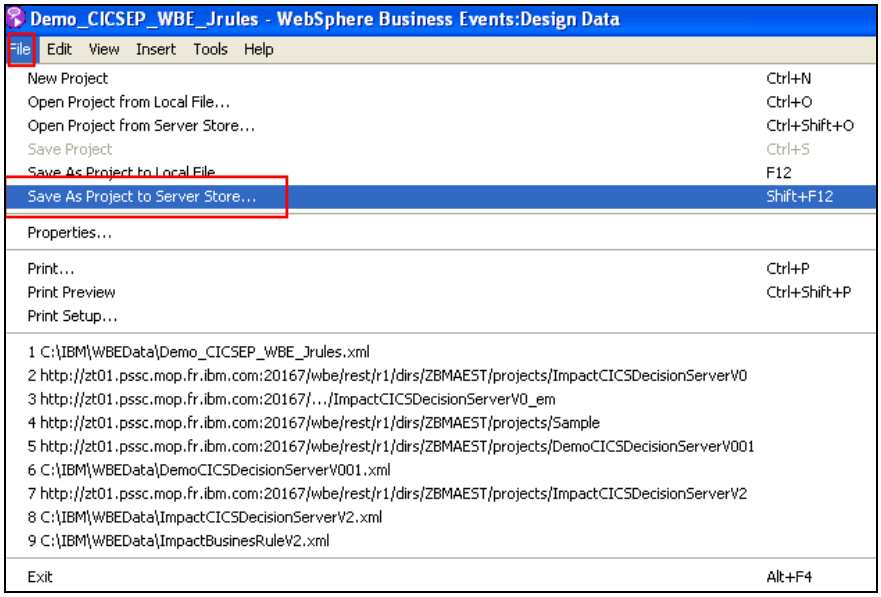


4. The Email data will look like this:

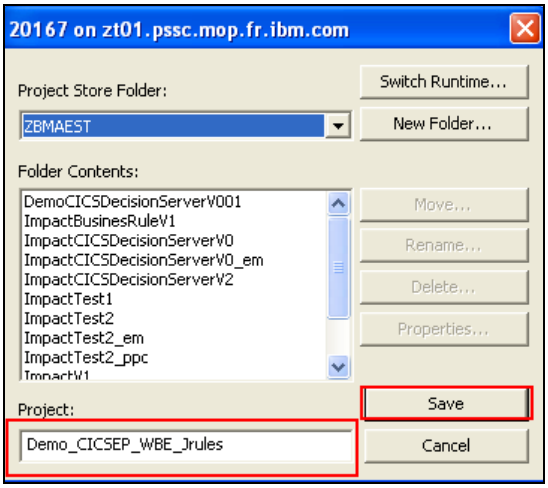


24. Save work in the Server Store ...

From the File menu select Save As Project to Server Store...



In the **Project**: field **type Demo_CICSEP_WBE_JrulesVxx** (where "**xx**" is your team number to substitute) and click **Save**



5.2 Defining Basic Interaction Sets

WebSphere Business events provides an environment in Business Space to allow Business users to quickly decide what actions to take in response to what situations and events, In this task you will use Business Space to define some simple interaction sets that will allow the rules based decisions to be invoke as part of the event processing.

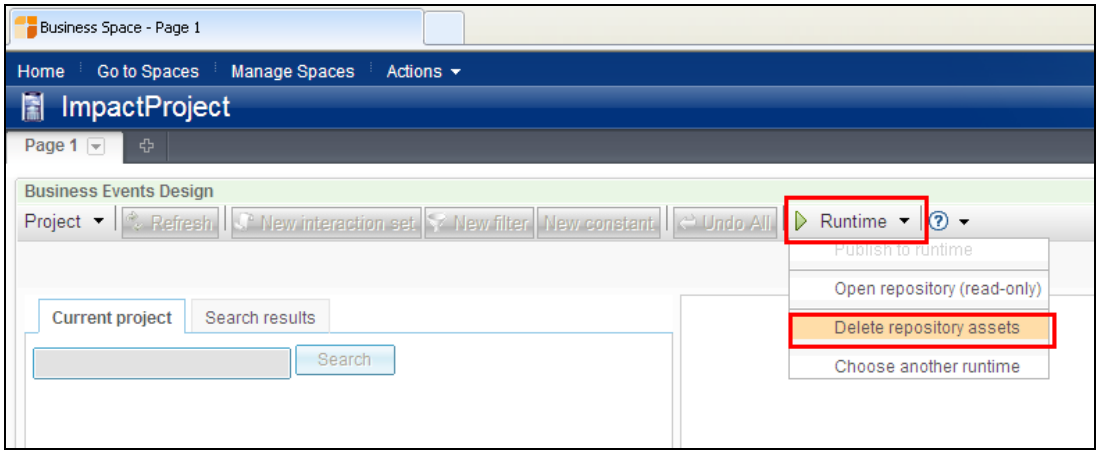
5.2.1 Open Business Space

WebSphere Business events provides an environment in Business Space to allow Business users to quickly decide what actins to take in response to what situations and events, In the step you will

To launch Business Space:

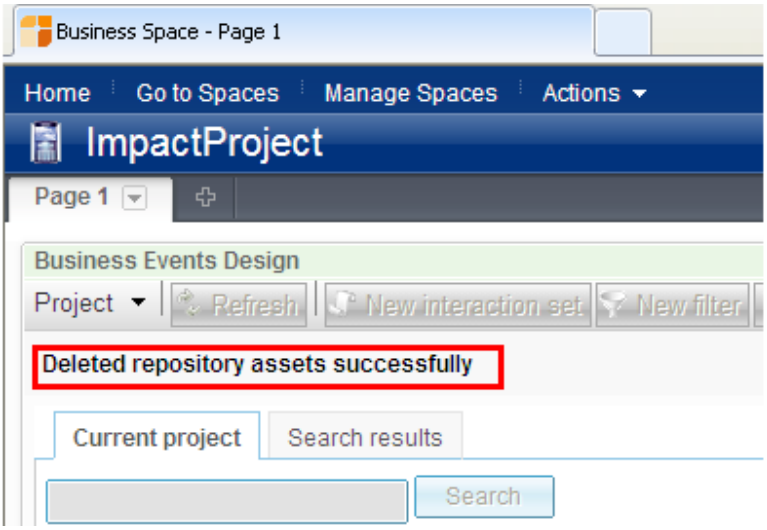
5. Open a browser at url https://zt01.pssc.mop.fr.ibm.com:20168/mum/resources/bootstrap/login.jsp

6. Login with User **ZBMAEST** password **zbmaest** and navigate to the Business Events Design tab.



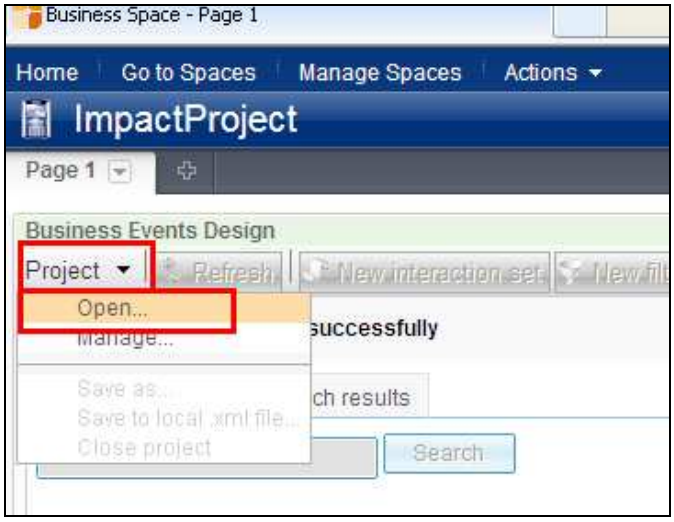
Click on Runtime ==> Delete repository assets to recycle your business space

Click OK.

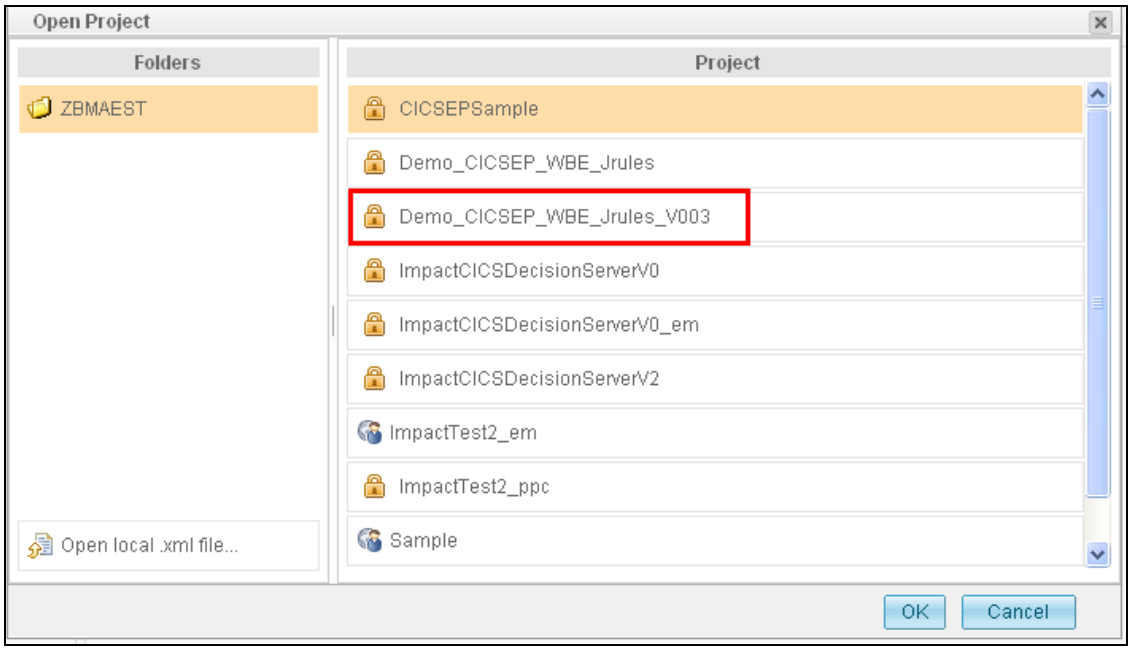


You will see this message will mean your business space is recycle to design a new business.

7. Select **Project > Open**



and then select the **Demo_CICSEP_WBE_Jrules_V0xx** where “xx” is your team number Project from the wbeAdmin folder.



8. Type **OK** and examine the structure of the project created in **Design Data.**

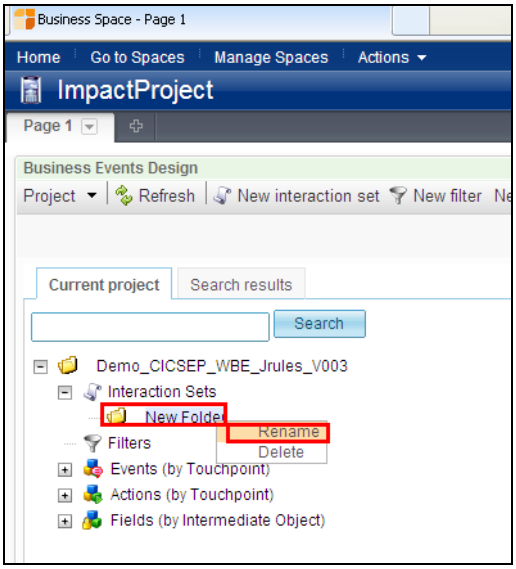
5.2.2 Define the interaction sets

During this step you will define interactions that will invoke the Rule decision service in response to a CICS Application event, and then send the response to an Offer Loan Action.

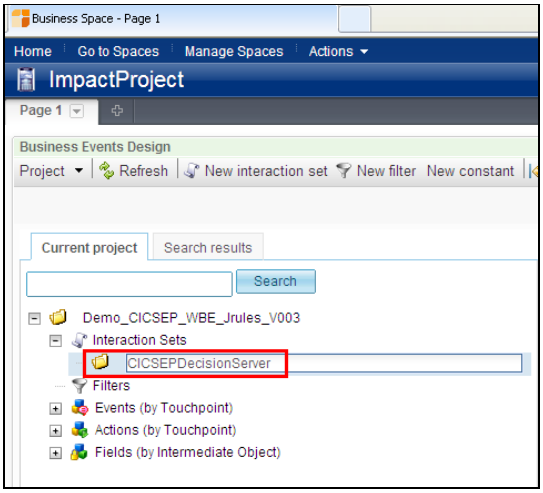
13. First we will define a folder to store the interactions

    Right click on the Interaction Sets and Click Create Folder .

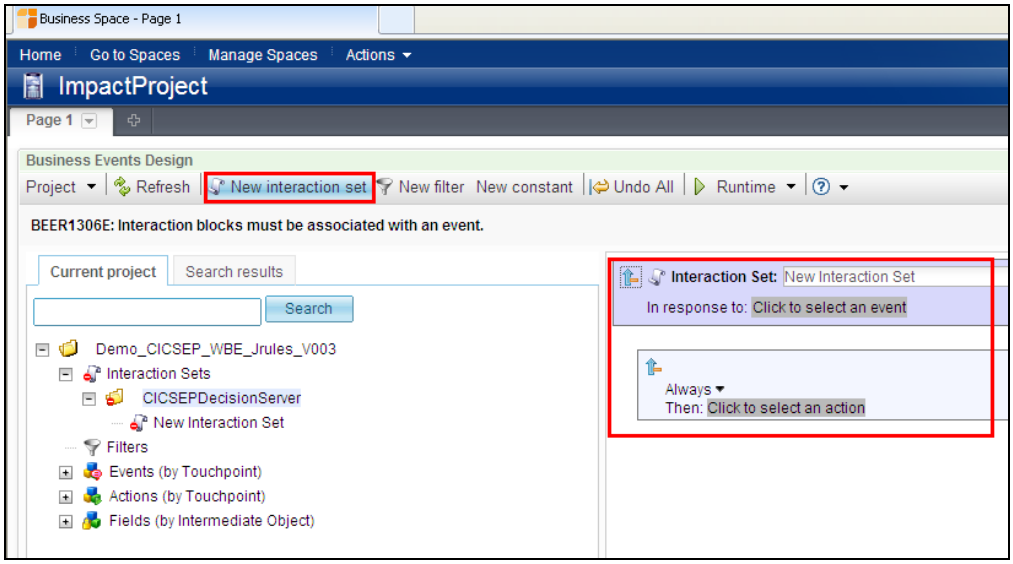
    Right click on the new Folder and Rename



14. Set the Interaction Set name field to **CICSEPDecisionServer**.

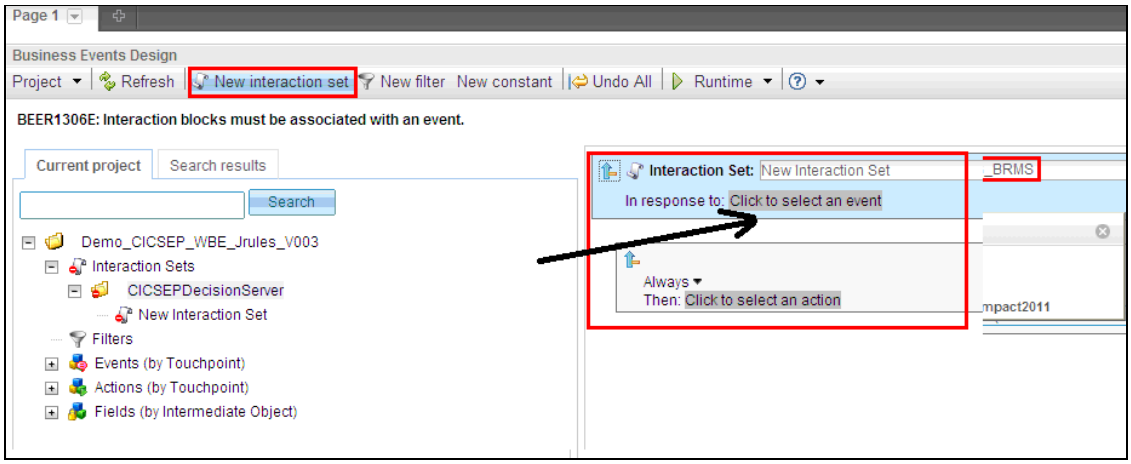


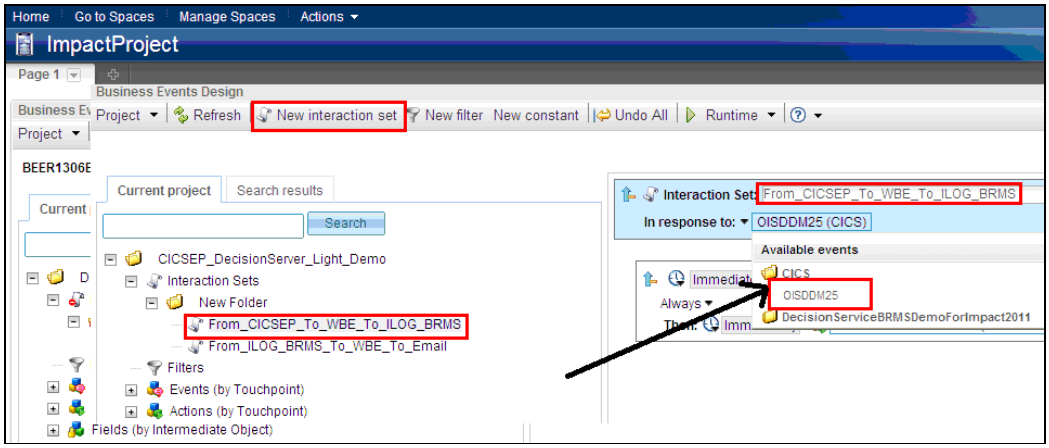
15. Now define the interaction to invoke the decision service. Click the **New Interaction Set** button.



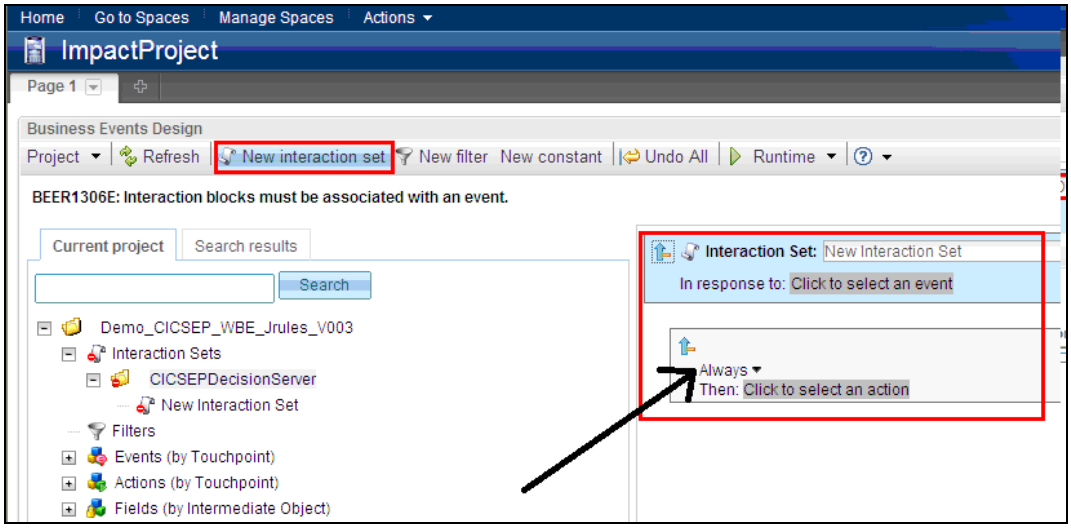
Name your Interaction Set (Ex: From_CICSEP_To_WBE_To_ILOG_BRMS)

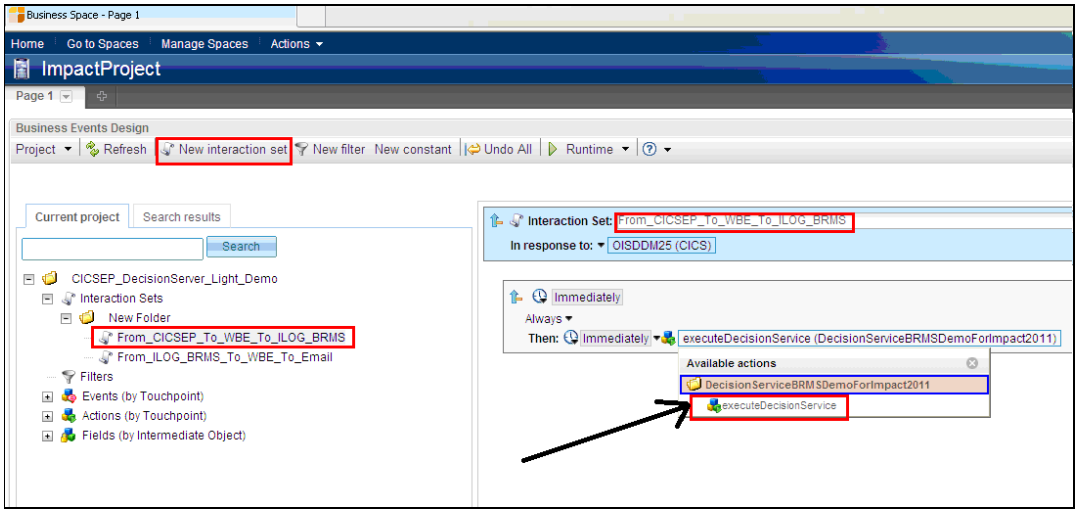
16. Click "**Click to select event"** link and then select the **CICS > OISDDM25** event.



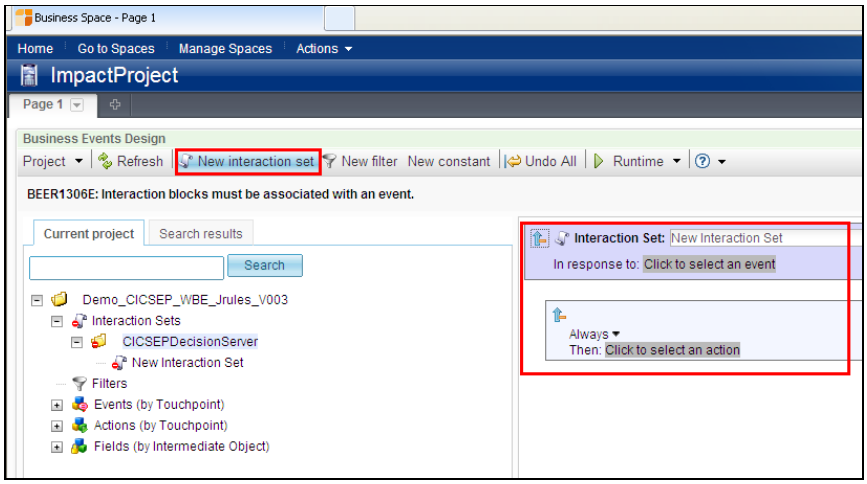


17. Click click to select an action link and then select the DecisionServiceCICS_DS_rulesxx > executeDecisionService action.

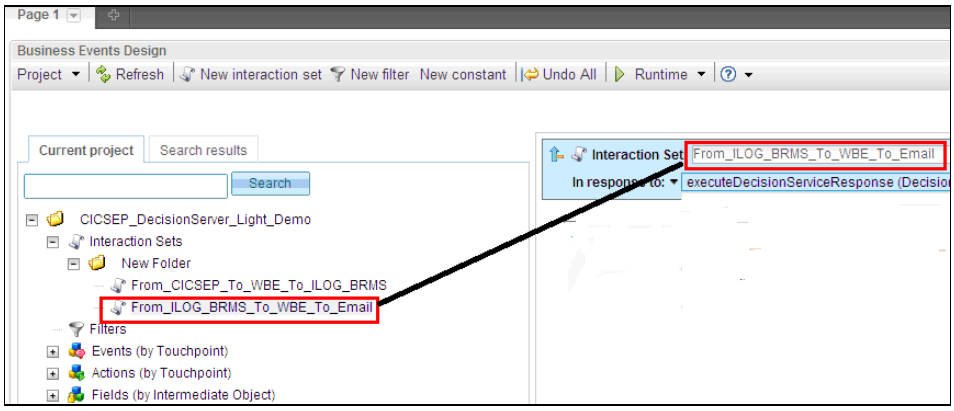




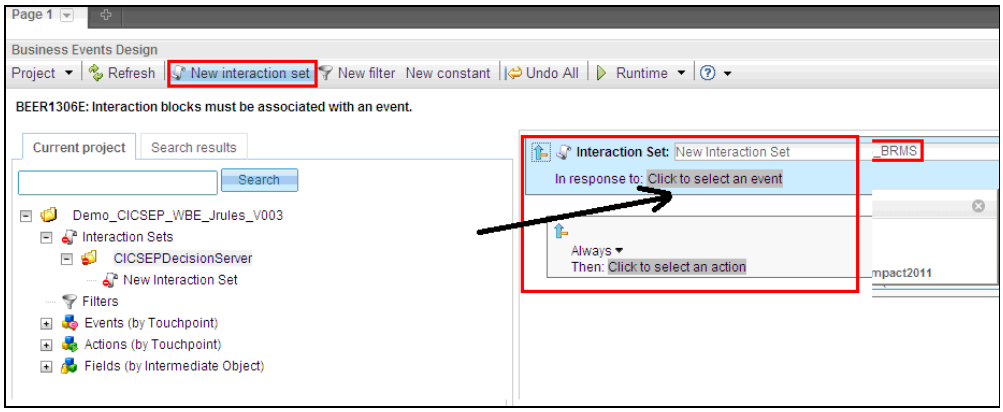
18. Now you will define the interaction to route the response from the decision service to an action. Click the **New Interaction Set** button.

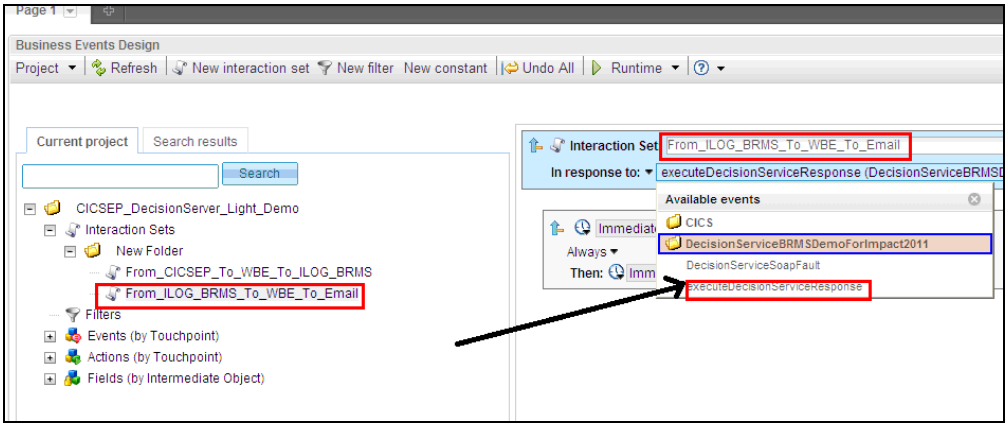


19. Set the Interaction Set name field to : *From_ILOG_BRME_To_WBE_To_Email.*

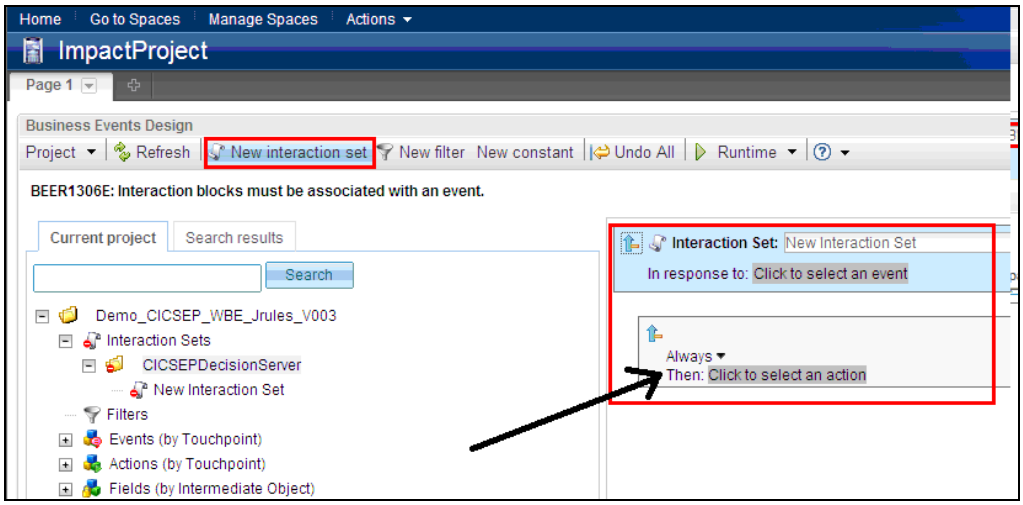


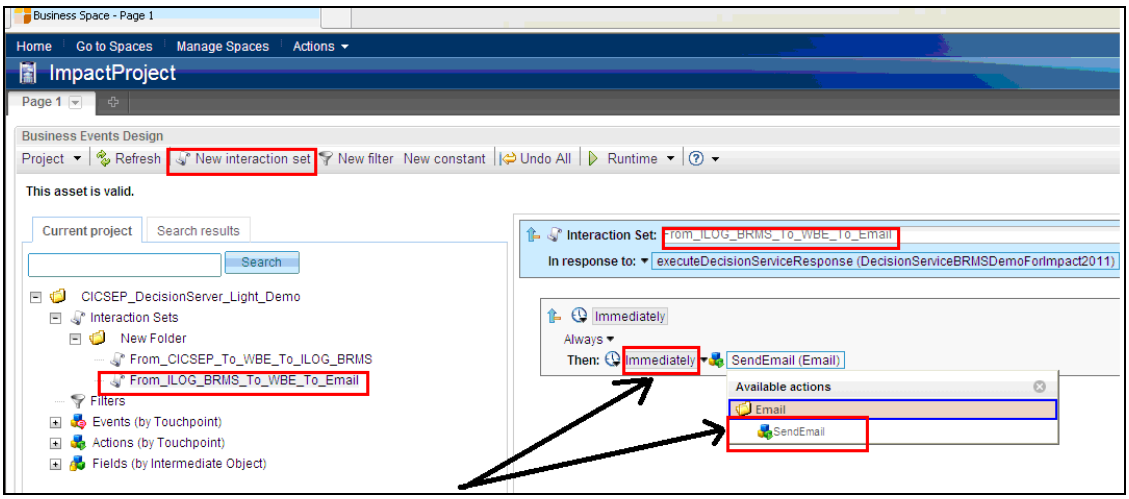
20. Click click to select event link and then select the DecisionServiceCICS_DS_rulesxx > executeDecisionServiceResponse event.





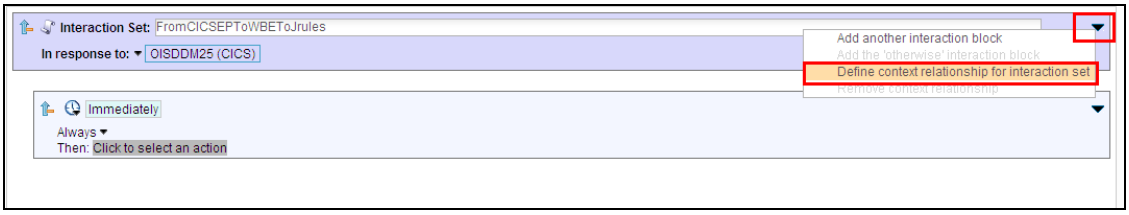
21. Click click to select an action link and then select Email > sendEmail action.



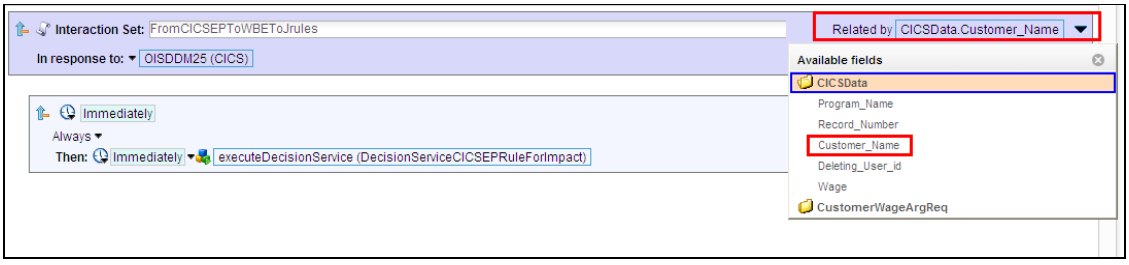


22. Now we need to define context of a relationship between the interaction in order to keep the Holding Data

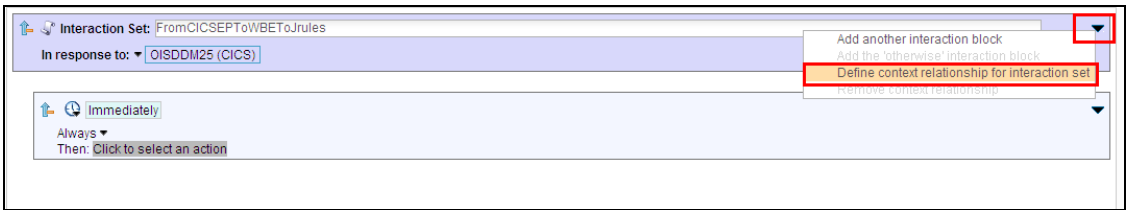
Open the combo in front of Interaction Set " *From_CICSEP_To_WBE_To_ILOG_BRMS*" ,



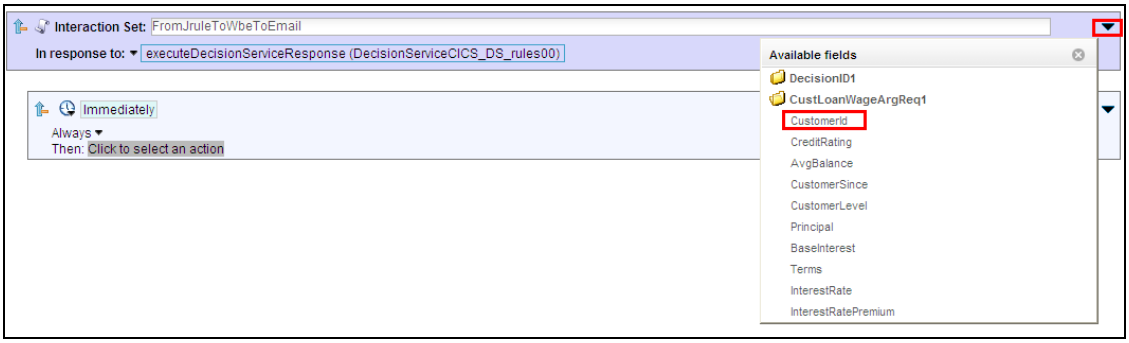
Select Available Fields ==> CICSData ==> Customer_Name as the field to us to define the Context of relationship for this Interaction Set



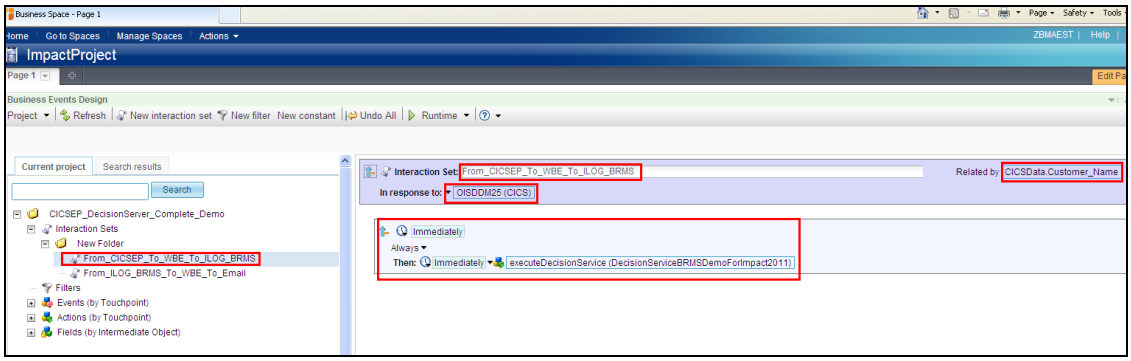
Repeat the same process for the other Interaction set *"From_ILOG_BRME_To_WBE_To_Email"*

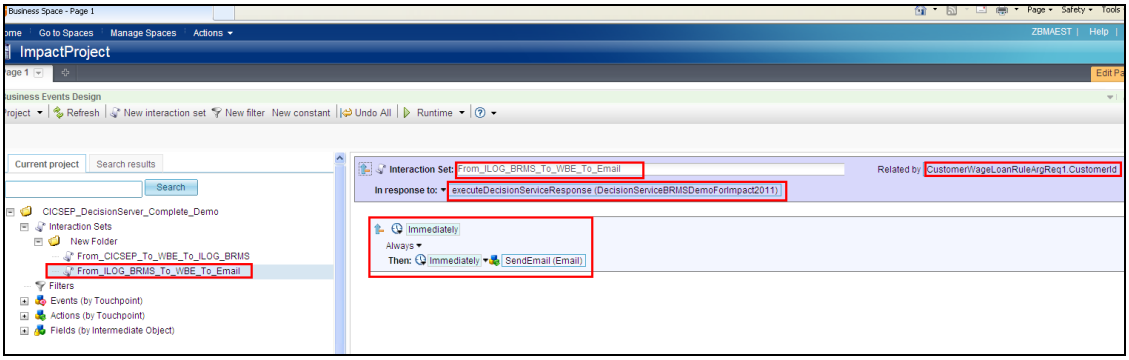


Select Available Fields ==> CustomerLoanWageArgReq1 ==> CustomerId as the field to us to define the Context of relationship for this Interaction Set

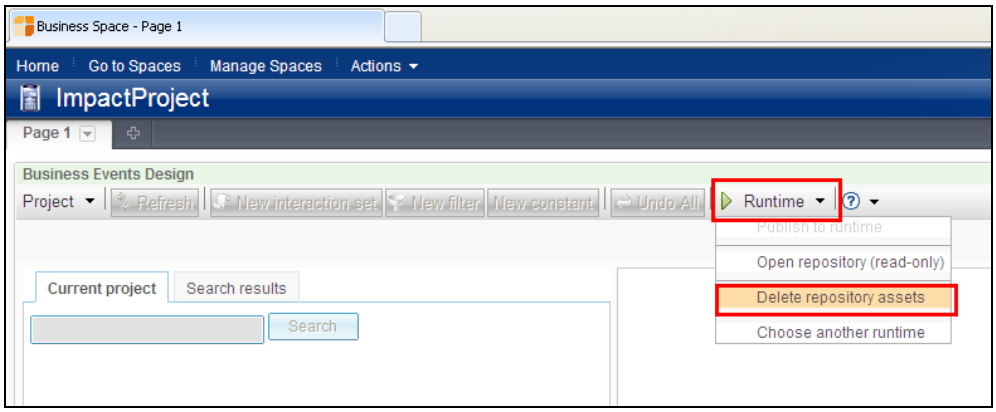


You business space will look like these figures

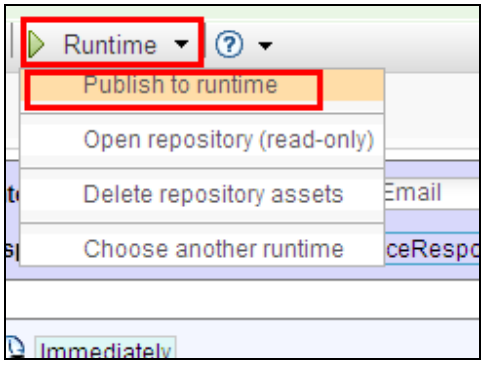




23. Select **Runtime > Delete repository assets** to clear any previous deployments



24. Select **Runtime > Publish to runtime** to deploy the Event project to the runtime.



25. Before you test these interactions, you must make sure that the connectors are running.

Go to TSO ==> SDSF and check job prefix ZB*

```
 Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF DA ZT01     ZT01     PAG  0  CPU/L/Z   1/  2/  2   DATA SET DISPLAYED
COMMAND INPUT ===>                                        SCROLL ===> CSR
NP   JOBNAME  StepName ProcStep JobID    Owner    C Pos DP Real Paging    SIO
_    ZBCONN   CONNECT           JOB08648 F058971  A LO  FF  290    0.00   0.09
_    ZBCONN3  STEP1             STC05976 F058971    LO  FF  326    0.00   0.00
     ZBCONN2  *OMVSEX           STC05508 F058971    IN  F0 160T    0.01  37.83
     ZBSR11AS ZBSR11AS BBOPASR  STC08645 ZBASRU     IN  F4 265T    0.00   0.01
     ZBDEMNA  ZBDEMNA  BBODAEMN STC08642 ZBACRU     NS  FE 7488    0.00   0.00
     ZBSR11A  ZBSR11A  BBOPACR  STC08641 ZBACRU     NS  FE  79T    0.00   0.00
     ZBCONN1  *OMVSEX           STC08034 F058971    LO  FF  388    0.00   0.00
     ZBSR11AA ZBSR11AA BBOPCRA  STC08644 ZBACRU     IN  F4 119T    0.00   0.07
```

Ask for job ZBCONN:

```
 Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF DA ZT01     ZT01     PAG  0  CPU/L/Z   1/  2/  1  LINE 1-8 (8)
COMMAND INPUT ===>                                        SCROLL ===> CSR
NP   JOBNAME  StepName ProcStep JobID    Owner    C Pos DP Real Paging    SIO
?_   ZBCONN   CONNECT           JOB08648 F058971  A LO  FF  290    0.00   0.00
     ZBCONN3  STEP1             STC05976 F058971    LO  FF  326    0.00   0.00
     ZBCONN2  *OMVSEX           STC05508 F058971    IN  F0 160T    0.00   2.24
     ZBSR11AS ZBSR11AS BBOPASR  STC08645 ZBASRU     IN  F4 265T    0.00   0.00
     ZBDEMNA  ZBDEMNA  BBODAEMN STC08642 ZBACRU     NS  FE 7488    0.00   0.00
     ZBSR11A  ZBSR11A  BBOPACR  STC08641 ZBACRU     NS  FE  79T    0.00   0.00
     ZBCONN1  *OMVSEX           STC08034 F058971    LO  FF  388    0.00   0.00
     ZBSR11AA ZBSR11AA BBOPCRA  STC08644 ZBACRU     IN  F4 119T    0.00   0.00
```

Then select STDOUT to verify the Connector availability

```
 Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF JOB DATA SET DISPLAY - JOB ZBCONN   (JOB08648)    LINE 1-8 (8)
COMMAND INPUT ===>                                        SCROLL ===> CSR
NP   DDNAME   StepName ProcStep DSID Owner    C Dest              Rec-Cnt Page
     JESJCLIN                    1 F058971  H                        14
     JESMSGLG JES2               2 F058971  H                         4
     JESJCL   JES2               3 F058971  H                        15
     JESYSMSG JES2               4 F058971  H                         0
     $INTTEXT JES2               5 F058971  A                         6
     $JOURNAL                    6 F058971  A                        31
s_   STDOUT   CONNECT          101 F058971  H                         0
     STDERR   CONNECT          102 F058971  H                         0
```

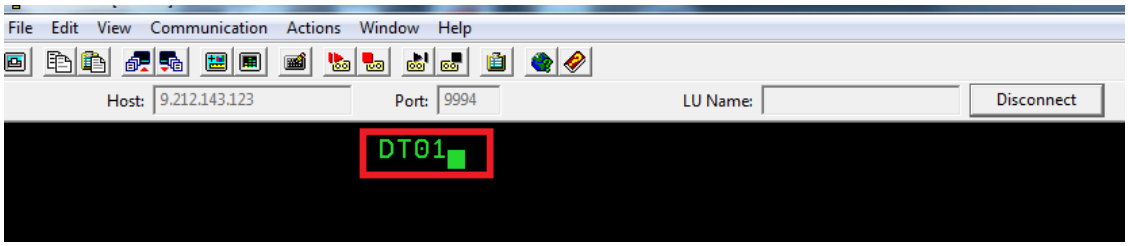
You will see these messages showing the connector ready to monitor ILOG Rule execution method “executeDecisionService” and the eMail action sending.

```
 SDSF OUTPUT DISPLAY ZBCONN   JOB08648  DSID   101 LINE 319     COLS 02- 81
 COMMAND INPUT ===> _                                          SCROLL ===> CSR
BEER4675I: Recovering the transaction log from job queue executeDecisionServiceQ
BEER4675I: Recovering the transaction log from job queue executeDecisionServiceQ
BEER4674I: Reading log file /wasv70config/zbcell/zbwbe1a/queues//executeDecision
BEER0637I: Starting one action worker
BEER0632I: The SOAP module is monitoring action: executeDecisionService
BEER0631I: The SMTP module is monitoring action: SendEmail
BEER0616I: The JMS module is monitoring event: OISDDM25
BEER0639I: The Connector reload is complete
```

5.2.3 Test the all scenario from CICS to Email

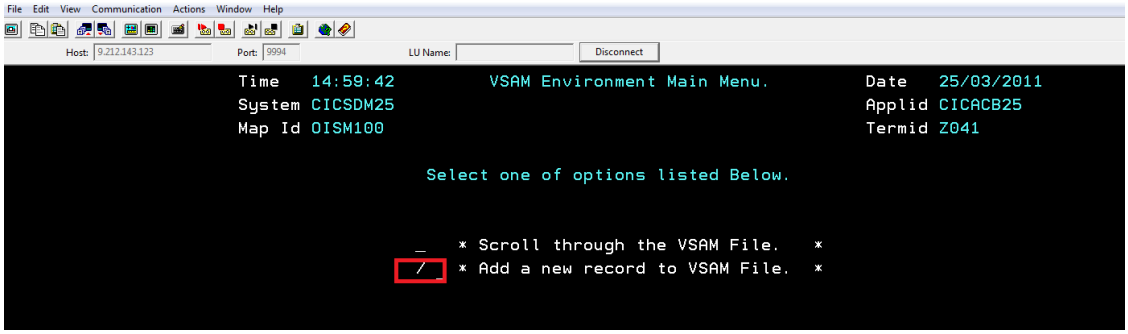
In this step you will test an event arriving from CICS Application, Capturing the event, invoking the decision service and sending an eMail as described in the previous step.

8. Connect to CICS Application and run DT01 transaction.



9. You will obtain this menu.

   Set **/** on “Add a new record to VSAM File”

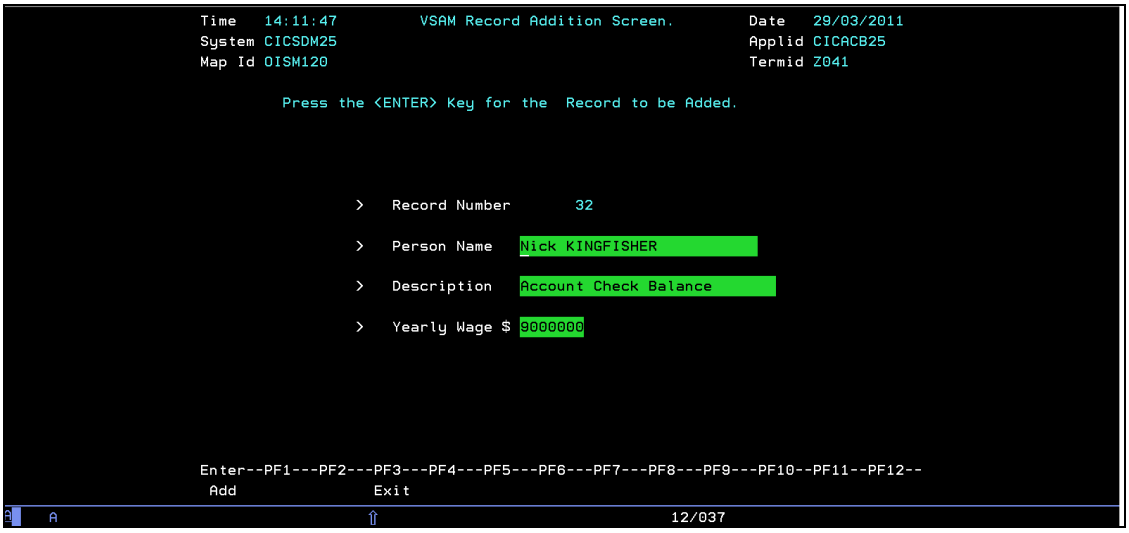


10. Fill values for fields :

    Person Name

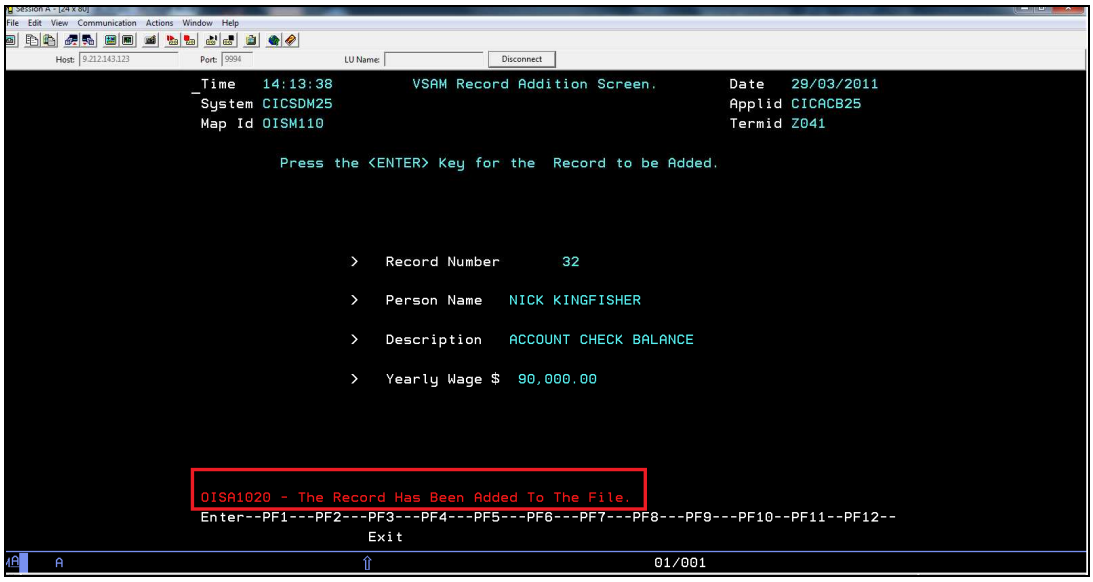
    Description

    YearlyWage



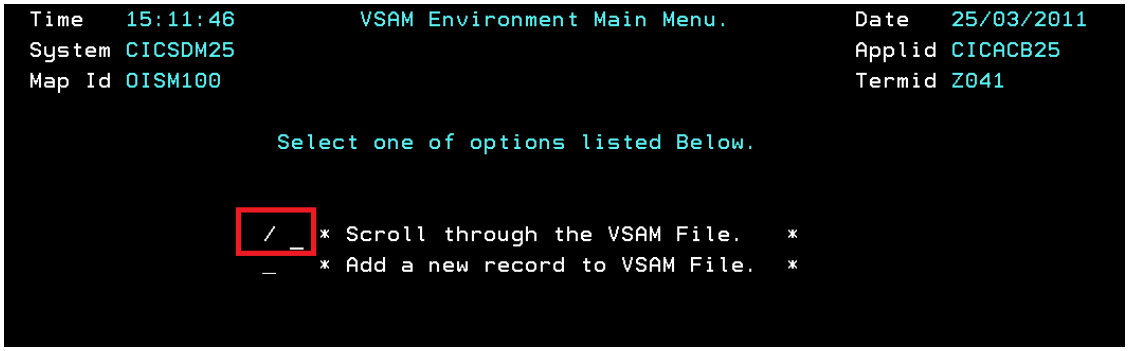
And hit "***Enter***"

You will obtain this message "*OISA1020 - The Record Has been Added To The File*"



Hit "PF3" key to return to main menu

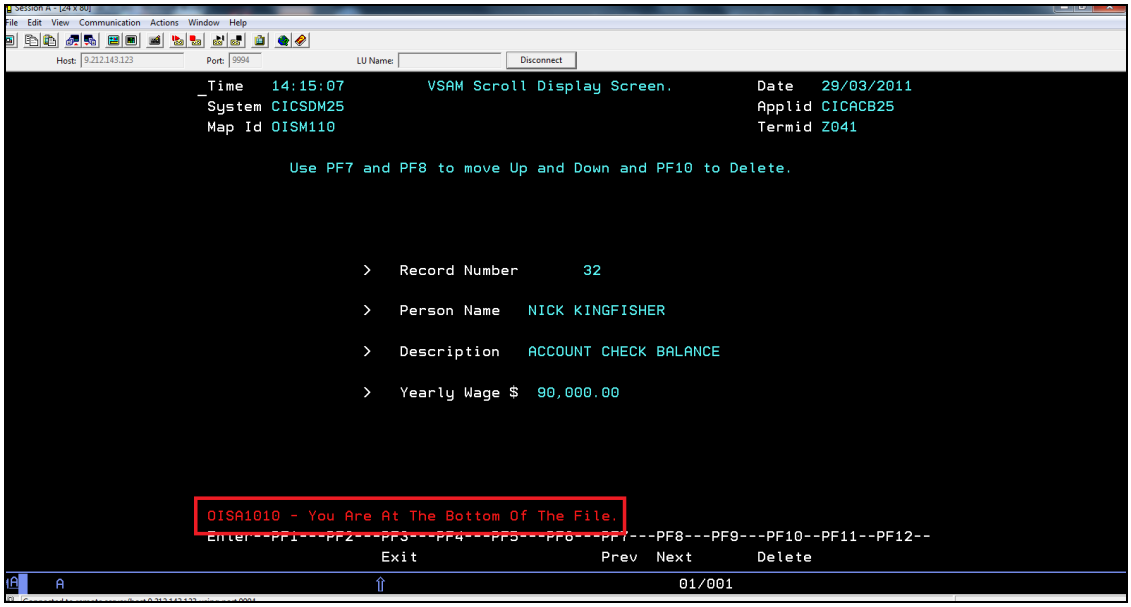
11. Now select "Scroll through the VSAM File" as shown below



12. Browse the record by hitting the "**PF8**" key until the end of the record on the VSAM.

At the end, you will obtain this message : "*OISA1010 - You Are At The Bottom Of The File*"

The current record will be the one you create.



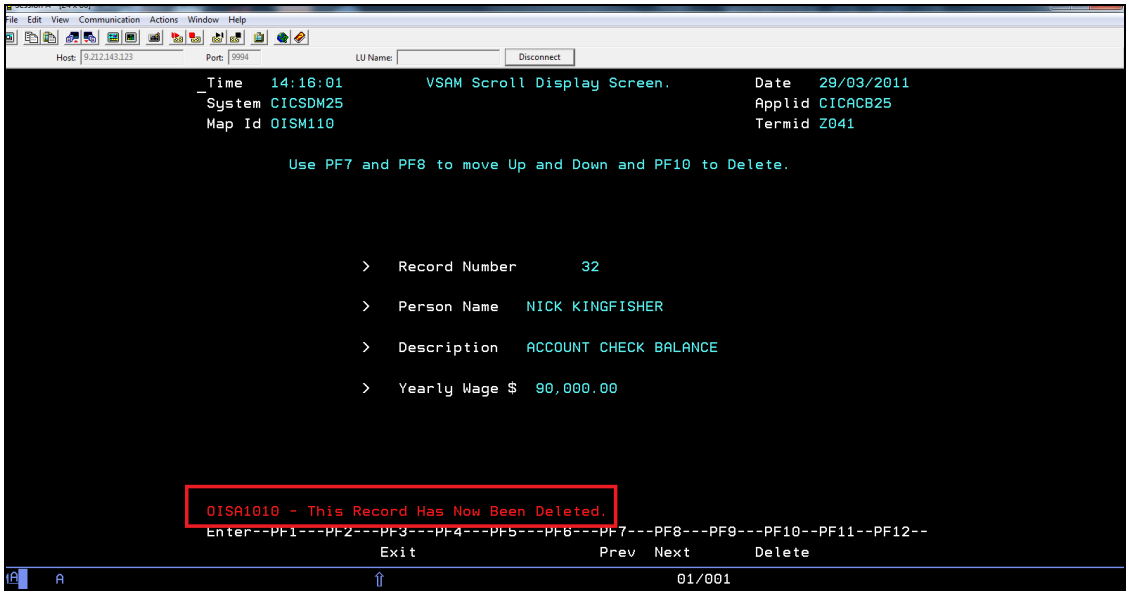
13. Delete the current record.

    Hit the PF10 key to delete your record.

    By this action, CICS Event will capture and format the event of Record deletion to WBE.

    Then WBE will ask to Jrules to decide action.

    Return from Jrules, WBE will merge data and Send Email.

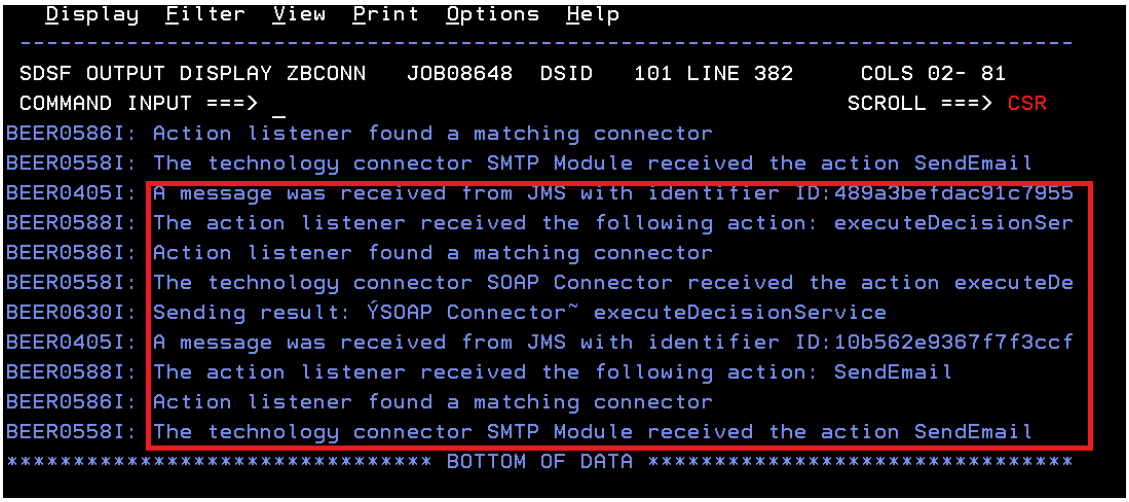


By hitting "**PF10**" key, you obtain this message : "*OISA10110 - This Record Has Now Been Deleted.*"

Now , go to TSO SDSF to check the connector

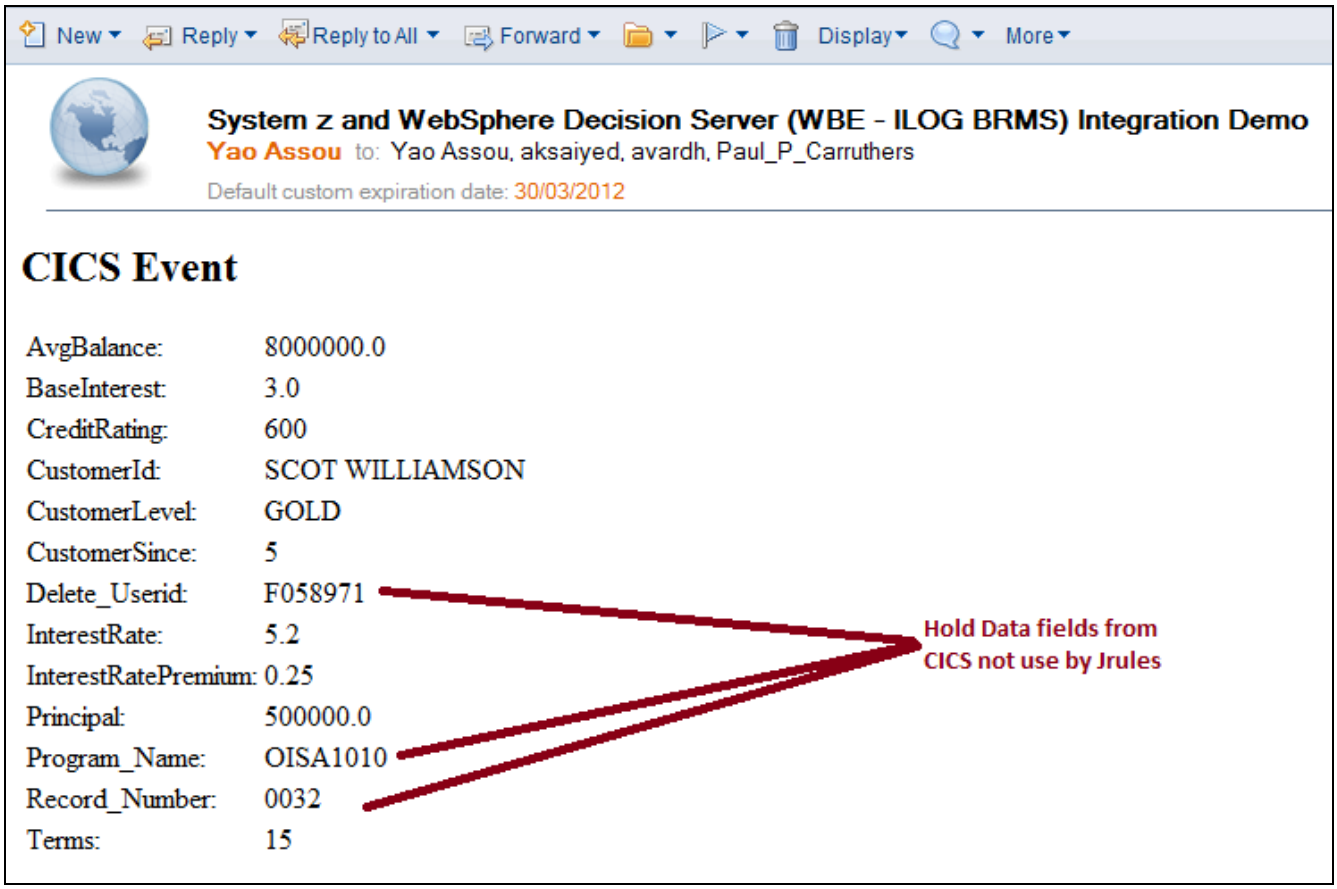
Connector will receive theses messages on outpout.

Message "*The technology connector SMTP Module receive the action SendEmail*" mean everything is ok.



14. Check now your email.

.



5.3 Summary

Congratulations! You have completed the IBM CICS Event Processing and WebSphere Decision Server integration scenario.

During this lab you became familiar with all the Decision Server modules:

- **Rule Studio** to design and develop the business rule application.
- **Design Data** to design and develop the business event application.
- **Rule Team Server** for business users to manage the Rule Based Decisions.
- **Rule Execution Server** to execute and monitor the business rules.

- **Business space** to manage your event project by creating a new business process

We hope that this lab helped you understand how you can use IBM WebSphere Decision Server to externalize the business logic from your own application and place it in the hands of the business users.

Appendix A. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have

been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

[This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental. All references to fictitious companies or individuals are used for illustration purposes only.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Appendix B. Trademarks and copyrights

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM	AIX	CICS	ClearCase	ClearQuest	Cloudscape
Cube Views	DB2	developerWorks	DRDA	IMS	IMS/ESA
Informix	Lotus	Lotus Workflow	MQSeries	OmniFind	
Rational	Redbooks	Red Brick	RequisitePro	System i	
System z	Tivoli	WebSphere	Workplace	System p	

Adobe, Acrobat, Portable Document Format (PDF), and PostScript are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

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

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

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

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

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

ITIL is a registered trademark and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

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

NOTES

NOTES

The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, these materials. Nothing contained in these materials is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software. References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. This information is based on current IBM product plans and strategy, which are subject to change by IBM without notice. Product release dates and/or capabilities referenced in these materials may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way.

Please Recycle