



IBM Integration Bus

Implementing Analytics with the R Node using an Integration Service

Featuring:

Analytics using R
Integration service

June 2015

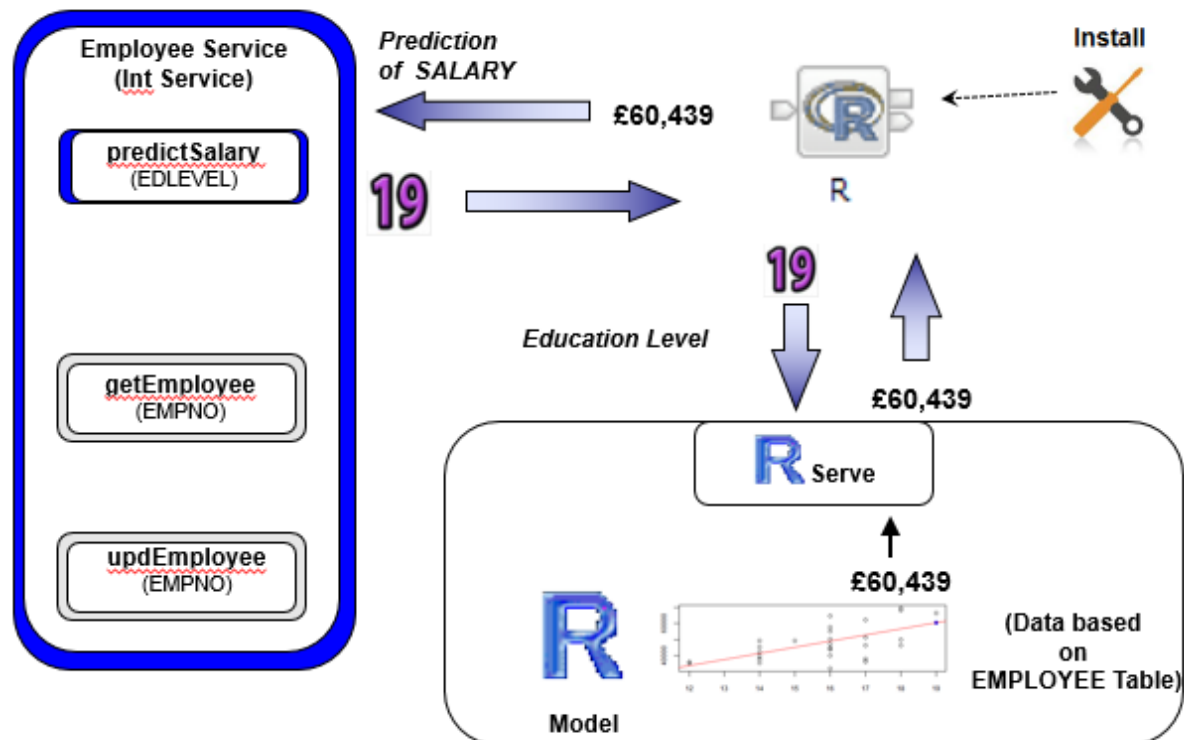
Hands-on lab built at product
Version 10.0.0.0

Contents

1. INTRODUCTION	3
2. PREPARE THE ENVIRONMENT	4
2.1 INSTALL THE IIB R NODE	4
2.2 OPEN THE WINDOWS LOG MONITOR FOR IIB.....	6
2.3 START R SERVER	7
2.4 IMPORT PRE-REQUISITES	9
3. UPDATE EMPLOYEE SERVICE APPLICATION TO CALL R	10
3.1 ADD OPERATION TO EMPLOYEESERVICE	10
3.2 CONFIGURE AN R NODE	13
3.2.1 <i>Configure the Basic properties</i>	13
3.2.2 <i>Configure Variables</i>	15
3.3 ADD MAPPING NODES	23
4. TESTING	27
4.1 DEPLOY THE SERVICE	27
4.2 TEST THE SERVICE WITH THE IIB FLOW EXERCISER	28
4.3 UNDERSTANDING THE VALUE RETURNED FROM R	35
4.4 ANALYZE THE MODEL BEFORE AND AFTER RUNNING THE APPLICATION	36
END OF LAB GUIDE	38

1. Introduction

This lab guide provides instructions on how to set up an R node in IIB and use it in an integration service. Data was extracted from the EMPLOYEE table in the DB2 SAMPLE database and an R model was created with the data. The simple application that you will create, will predict the value of Salary given a value for Education level. The following diagram outlines at a high level what you will do:



This lab guide shows you how to do the following tasks:

- 1) How to add R node support to your environment using the R node IIB Extension.
- 2) How to configure a simple message flow containing an R node using an integration service.
- 3) How to configure an R node in a message flow.
- 4) How to test a message flow with an R node and interface with R serve.
- 5) How to understand the number returned from R.

2. Prepare the Environment

2.1 Install the IIB R node

The Analytics node “R” is not part of the standard IIB V10 toolkit and runtime installation. The R node is being made available as an IIB extension from GitHub and is available from here:

<https://github.com/ot4i/integrate-R>.

In order to do this lab, you will need to install the R node. Two files need to be copied into the IBM Integration Bus installation. They are in `C:\student10\Analytics\Resources` and are:

- A toolkit component (currently `RNodeToolkit_1.0.0.20150417-1641.jar`): to be copied to `C:\IBM\IIB\10.0.0.0\tools\plugins`. (Once this has happened, the Integration Toolkit must be restarted).
- A runtime component (currently `RNodeRuntime-1.0.0.20150417-1641.par`): to be copied to `C:\IBM\IIB\10.0.0.0\server\bin`. (Once this has happened, any integration servers to which the R node will be deployed, must be restarted).

You can do this manually, if you wish. Alternatively, run a script we written for you, as follows:

1. Open an IIB Console and navigate to “`C:\student10\Analytics\Commands`” and run the command `RnodeInstall.cmd`
2. The script defaults to copying the files named above from `C:\student10\Analytics\Resources` to their respective destinations. In addition, it stops and starts IB10NODE by default. Accept these defaults by pressing Enter.

3. The output from the script should be like this:

```
C:\student10\Analytics\Commands>RnodeInstall.cmd
This command file must be run within an Integration Bus Command Console.
Betaworks Analytics Lab guide Rnode setup.
Enter IIB Node name (default is IB10NODE):
Enter R Node runtime file name (default is RNodeRuntime-1.0.0.20150417-1641.par)
:
Enter R Node runtime from folder (default is C:\student10\Analytics\Resources):

Enter R Node runtime to folder (default is C:\IBM\IIB\10.0.0.0\server\bin):
Enter R Node toolkit file name (default is RNodeToolkit_1.0.0.20150417-1641.jar)
:
Enter R Node toolkit from folder (default is C:\student10\Analytics\Resources):

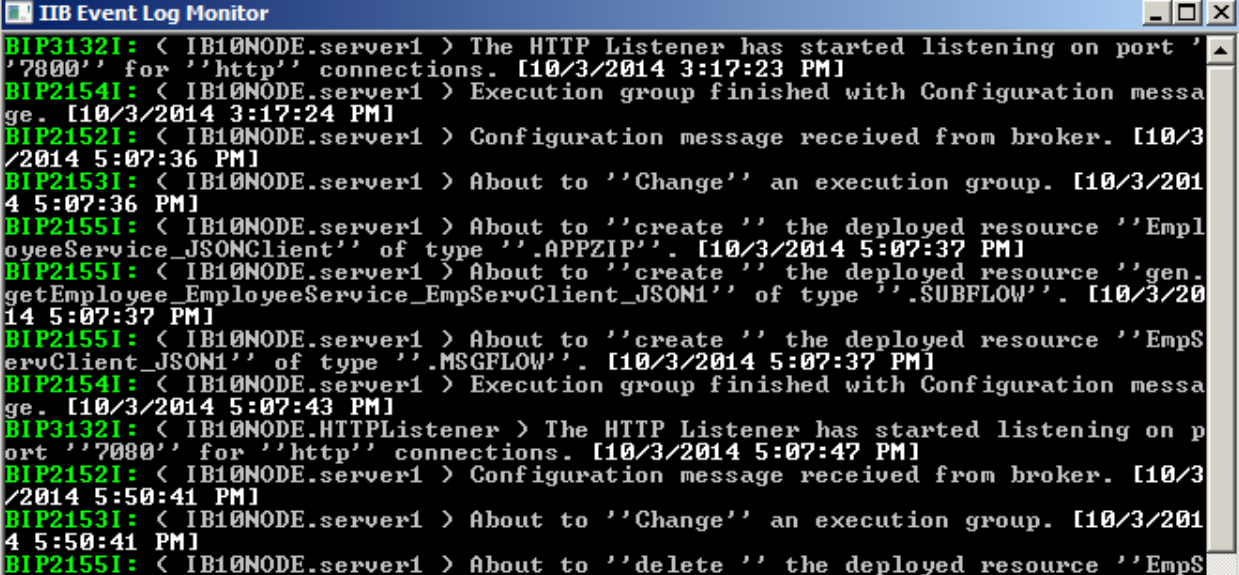
Enter R Node toolkit to folder (default is C:\IBM\IIB\10.0.0.0\tools\plugins):
-
Thankyou, using values "IB10NODE", "RNodeRuntime-1.0.0.20150417-1641.par", "C:\s
tudent10\Analytics\Resources", "C:\IBM\IIB\10.0.0.0\server\bin", "RNodeToolkit_1
.0.0.20150417-1641.jar", "C:\student10\Analytics\Resources", "C:\IBM\IIB\10.0.0
0\tools\plugins"
Ok to proceed? Use Ctrl-C to terminate.
Press any key to continue . . .
-
Stopping node "IB10NODE"
BIP8019E: Integration node 'IB10NODE' stopped.
This integration node is stopped; the command you issued cannot be processed whe
n an integration node is stopped.
A previous command has been issued to stop this integration node, or this integr
ation node has never been started.
This integration node can be started, changed, or deleted.
-
About to remove R Node code from IIB. Failure messages can be ignored.
Could Not Find C:\IBM\IIB\10.0.0.0\server\bin\RNodeRuntime-1.0.0.20150417-1641.p
ar
Could Not Find C:\IBM\IIB\10.0.0.0\tools\plugins\RNodeToolkit_1.0.0.20150417-164
1.jar
About to copy across R Node code.
Ok to proceed? Use Ctrl-C to terminate.
    1 file(s) copied.
    1 file(s) copied.
-
About to Start "IB10NODE"
BIP8096I: Successful command initiation, check the system log to ensure that the
component started without problem and that it continues to run without problem.
```

4. Check that the files have been copied across and the integration node has been started.

2.2 Open the Windows Log Monitor for IIB

A useful tool for IIB development on Windows is the IIB Log Viewer. This tool continuously monitors the Windows Event Log, and all messages from the log are displayed immediately.

From the Start menu, click IIB Event Log Monitor. The Monitor will open; it is useful to have this always open in the background.



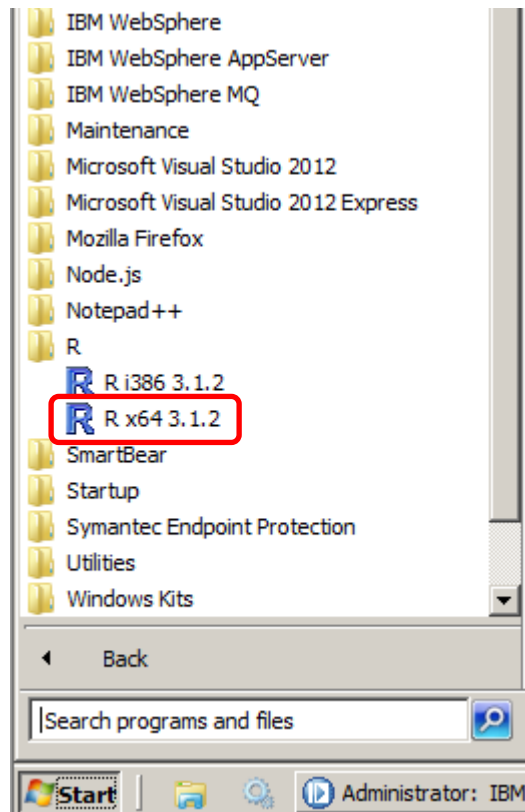
```
IIB Event Log Monitor
BIP31321: < IB10NODE.server1 > The HTTP Listener has started listening on port '
'7800' for 'http' connections. [10/3/2014 3:17:23 PM]
BIP21541: < IB10NODE.server1 > Execution group finished with Configuration messa
ge. [10/3/2014 3:17:24 PM]
BIP21521: < IB10NODE.server1 > Configuration message received from broker. [10/3
/2014 5:07:36 PM]
BIP21531: < IB10NODE.server1 > About to 'Change' an execution group. [10/3/201
4 5:07:36 PM]
BIP21551: < IB10NODE.server1 > About to 'create' the deployed resource 'Empl
oyeeService_JSONClient' of type '.APPZIP'. [10/3/2014 5:07:37 PM]
BIP21551: < IB10NODE.server1 > About to 'create' the deployed resource 'gen.
getEmployee_EmployeeService_EmpServClient_JSON1' of type '.SUBFLOW'. [10/3/20
14 5:07:37 PM]
BIP21551: < IB10NODE.server1 > About to 'create' the deployed resource 'EmpS
ervClient_JSON1' of type '.MSGFLOW'. [10/3/2014 5:07:37 PM]
BIP21541: < IB10NODE.server1 > Execution group finished with Configuration messa
ge. [10/3/2014 5:07:43 PM]
BIP31321: < IB10NODE.HTTPListener > The HTTP Listener has started listening on p
ort '7080' for 'http' connections. [10/3/2014 5:07:47 PM]
BIP21521: < IB10NODE.server1 > Configuration message received from broker. [10/3
/2014 5:50:41 PM]
BIP21531: < IB10NODE.server1 > About to 'Change' an execution group. [10/3/201
4 5:50:41 PM]
BIP21551: < IB10NODE.server1 > About to 'delete' the deployed resource 'EmpS
```

This tool is not shipped as part of the IIB product; please contact us directly if you would like a copy.

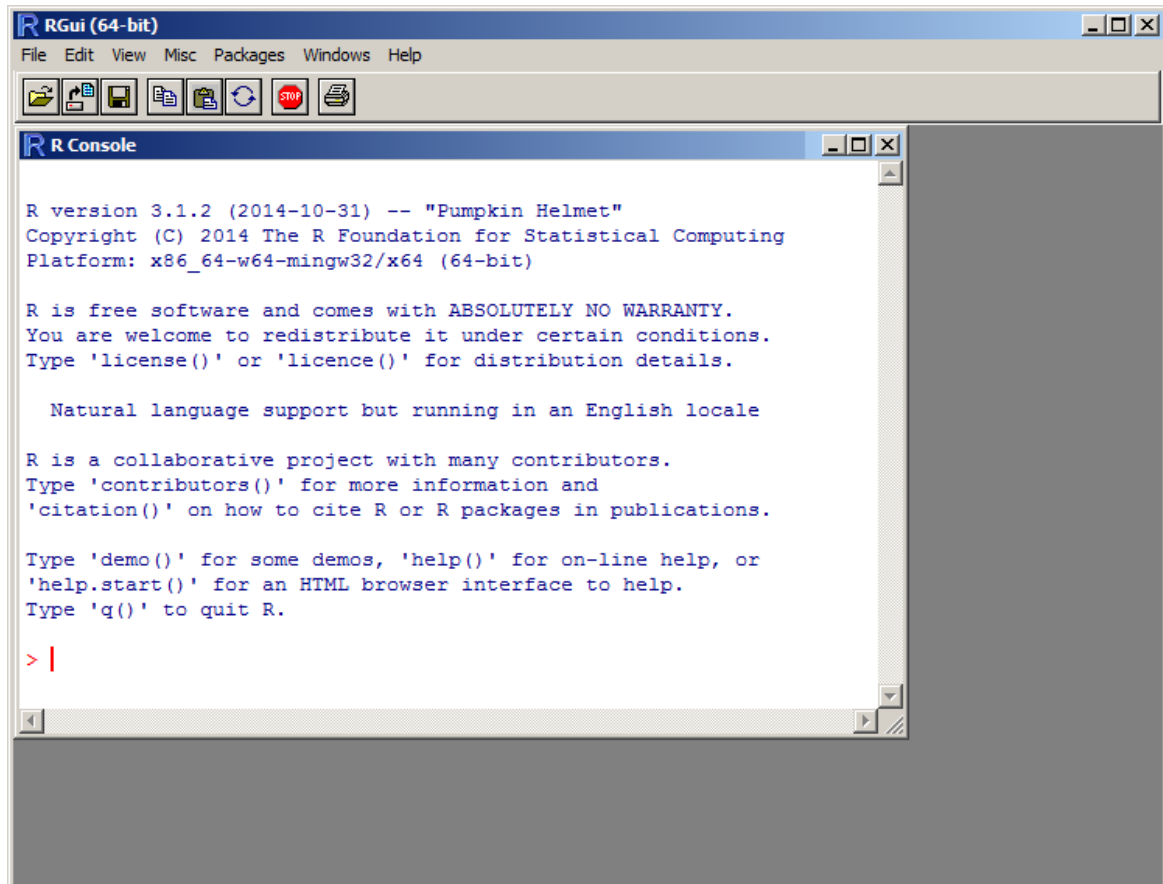
2.3 Start R server

In order to run a message flow with an R node, the R server needs to be started. In this section you will start R Server from the R GUI.

1. From the Windows Start menu start the R Gui, (click **Start > All programs > R > “R x64 3.1.2”**)



- The **RGui(64-bit)** with an **R Console** will open:



- In the "R console" type:

```
library(Rserve)
```

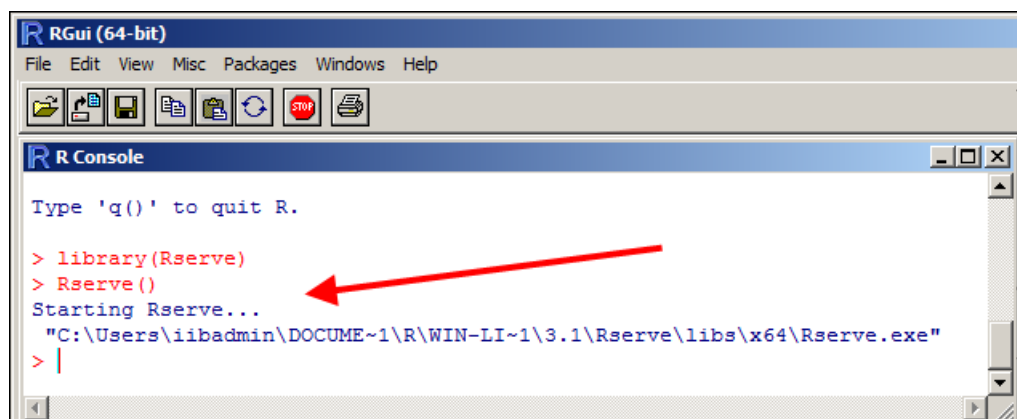
and press enter.

- In the same console type:

```
Rserve()
```

and press enter.

The R Console will look like this:



The IIB R node can now use Rserve to interface with R.

2.4 Import pre-requisites

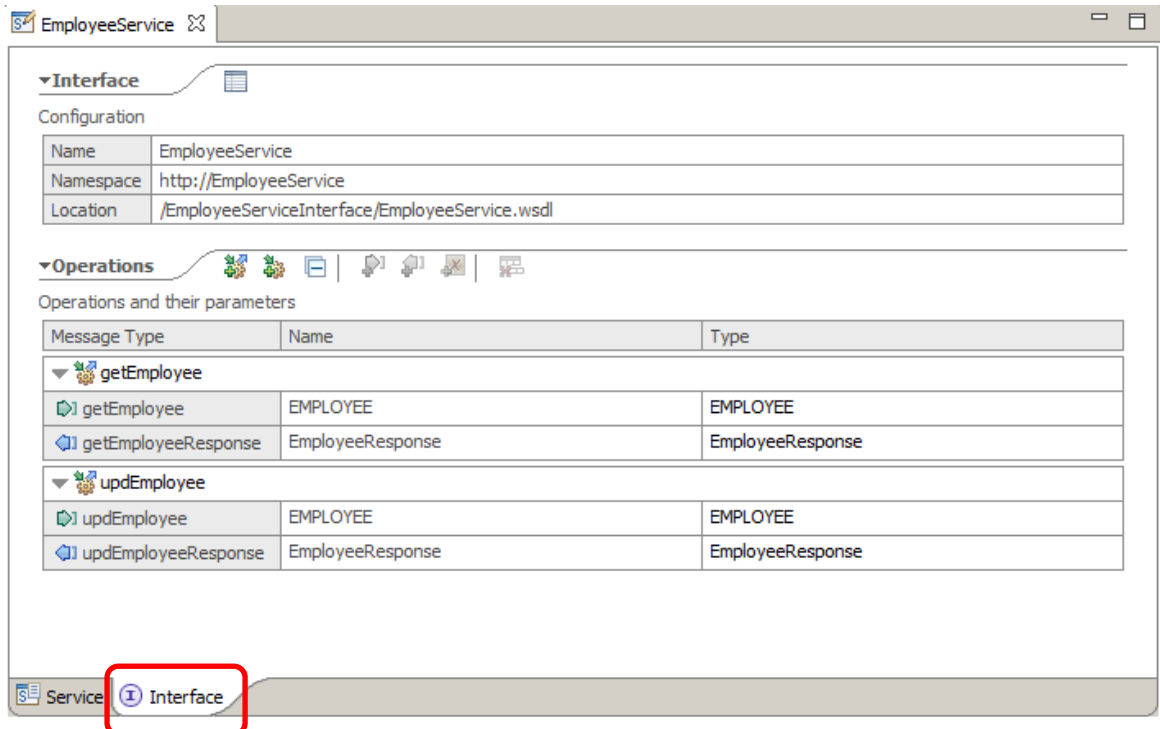
1. First, to ensure there are no conflicts with other components, switch to a new IIB workspace in the IIB Toolkit, for example name it c:\workspaces\IIB_R.
2. In the IIB Toolkit, import these Project Interchange files:
c:\student10\Integration_service\solution\EmployeeServiceInterface.V10.zip and
c:\student10\Integration_service\solution\ EmployeeService.V10.zip

3. Update Employee Service application to call R

3.1 Add Operation to EmployeeService

1. Go to the EmployeeService integration service and double-click the Integration Service Description in the navigator.

Select the Interface tab. You will see operations, getEmployee and updEmployee.



The screenshot displays the IBM Integration Bus interface for the EmployeeService. The window title is "EmployeeService". The main content area is divided into two sections: "Interface" and "Operations".

Interface Configuration:

Property	Value
Name	EmployeeService
Namespace	http://EmployeeService
Location	/EmployeeServiceInterface/EmployeeService.wsdl

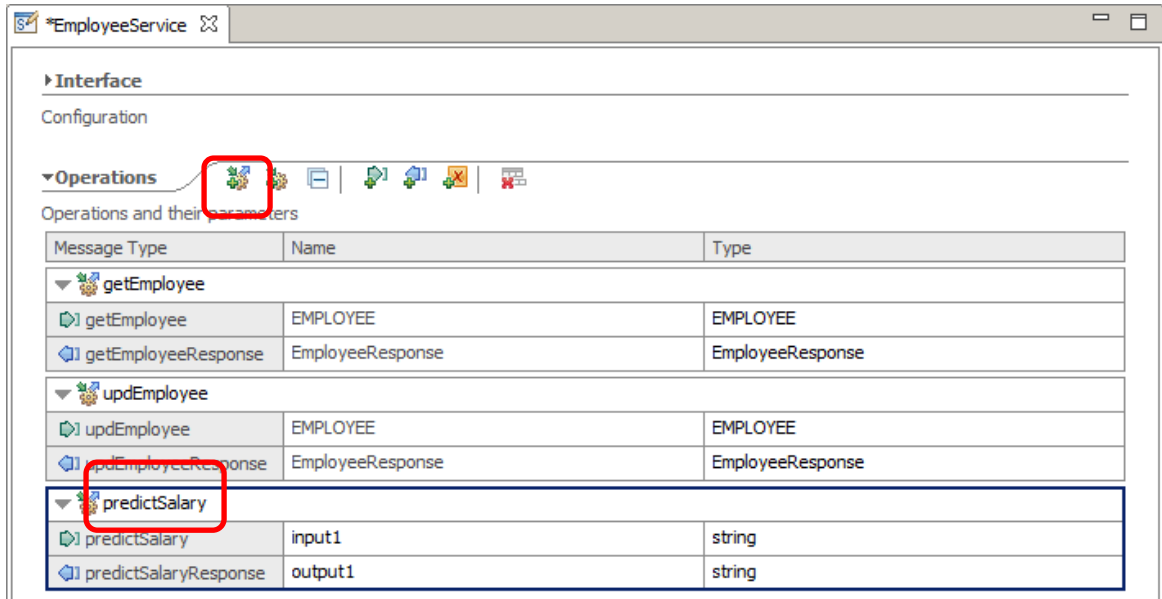
Operations and their parameters:

Message Type	Name	Type
getEmployee		
getEmployee	EMPLOYEE	EMPLOYEE
getEmployeeResponse	EmployeeResponse	EmployeeResponse
updEmployee		
updEmployee	EMPLOYEE	EMPLOYEE
updEmployeeResponse	EmployeeResponse	EmployeeResponse

The "Interface" tab is selected and highlighted with a red box in the bottom-left corner of the window.

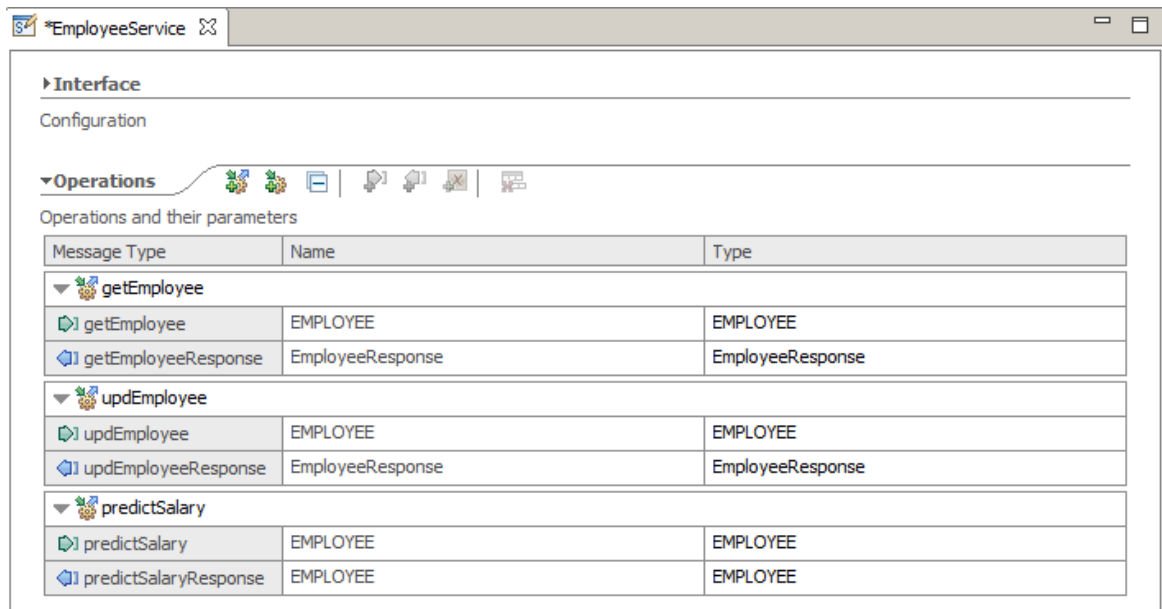
- Click the Add Request Response Operation button which will create another operation called "operation1".
Overtpe this to name the new operation "predictSalary".

Press return, which will automatically change the names of the input and output message types (predictSalary and predictSalaryResponse).

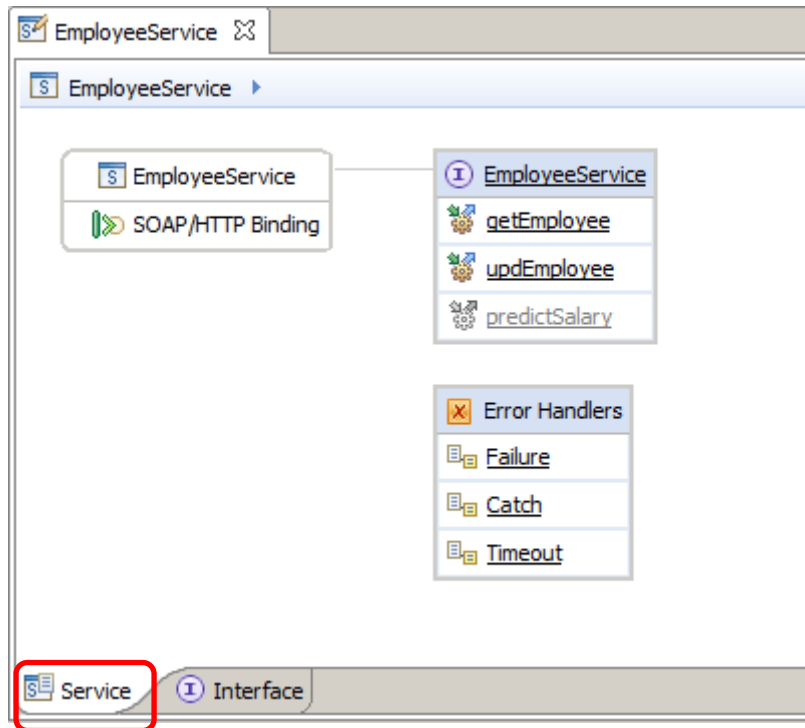


- Click the word "string" to change the type of both the input and output messages to EMPLOYEE (use the Browse button when the dialogue window opens, and type "e" to show all available types that beginning with "e").

Save the changes to the service.



- Click the Service tab to show the implementation. Note that predictSalary is greyed out, since it has not yet been implemented.

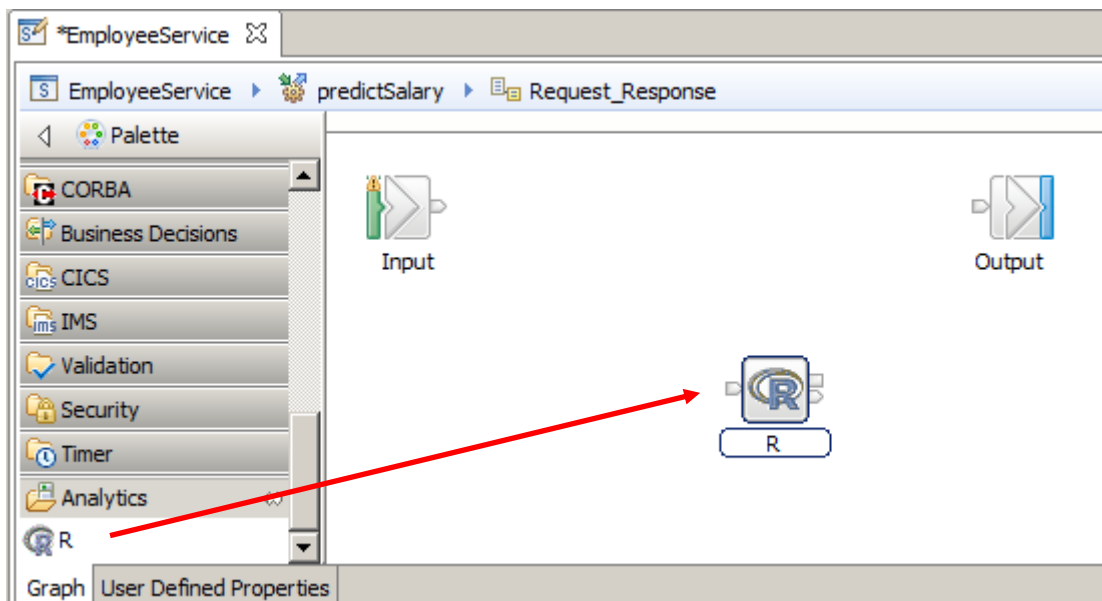


- Click predictSalary, which will open the subflow editor.

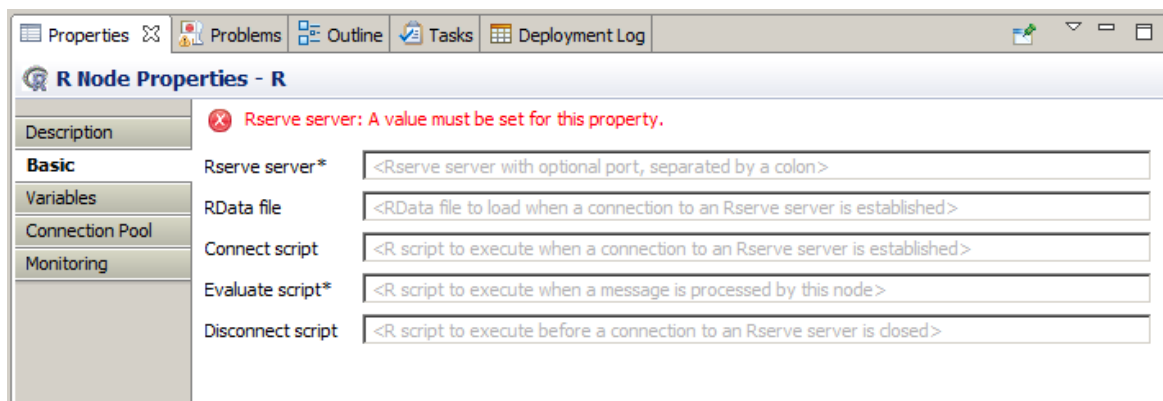
3.2 Configure an R Node

3.2.1 Configure the Basic properties

1. Open the Analytics folder in the message flow palette, and drag the R node onto the canvas:



2. Highlight the R node so that the properties tab shows the R node properties:



- Configure the R node properties as follows:

Basic Tab:

Rserve server : localhost:6311

RData file: C:\student10\Analytics\RWork\employeedata.RData

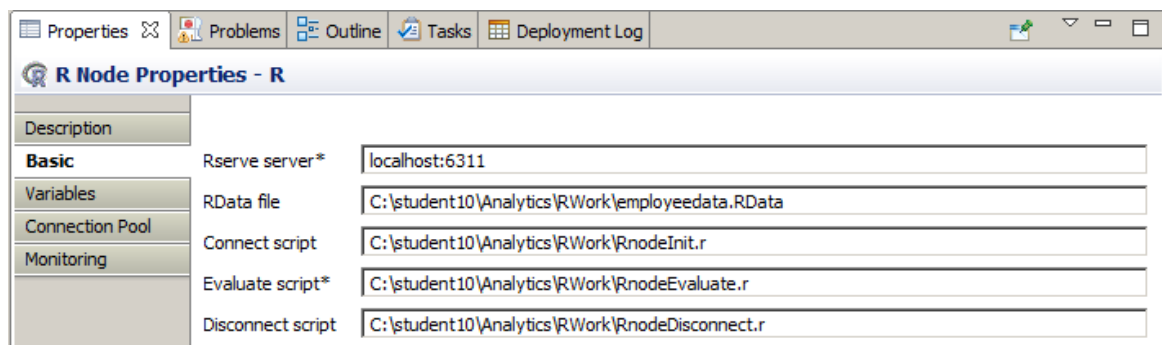
(note: in a distributed environment - the RData file needs to exist on the same machine as IIB toolkit)

Connect script: C:\student10\Analytics\RWork\RnodeInit.r

Evaluate script: C:\student10\Analytics\RWork\RnodeEvaluate.r

Disconnect script: C:\student10\Analytics\RWork\RnodeDisconnect.r

(note: in a distributed environment- the scripts need to exist on the same machine as IIB Run time environment, they also reference locations on the server where R is installed)



3.2.2 Configure Variables

The Variables tab in the R node is where you configure the variables to be used with your R environment. A Data Frame, in R, is a used to store a collection of data in rows and columns, (similar to the concept of a matrix in mathematics, however the types of a data frame do not necessarily need to be numeric).

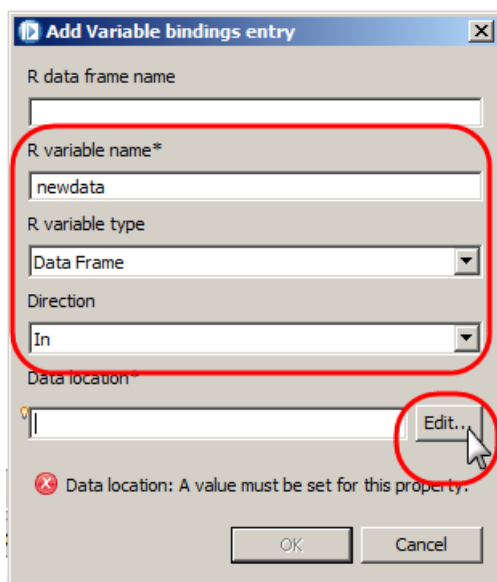
In the following section you will define the variables that will be used in R and how IIB will use them.

1. Variables Tab:

Click the “Add...” button to add a variable, the “Add Variable bindings entry” window will appear: Specify the following (case sensitive):

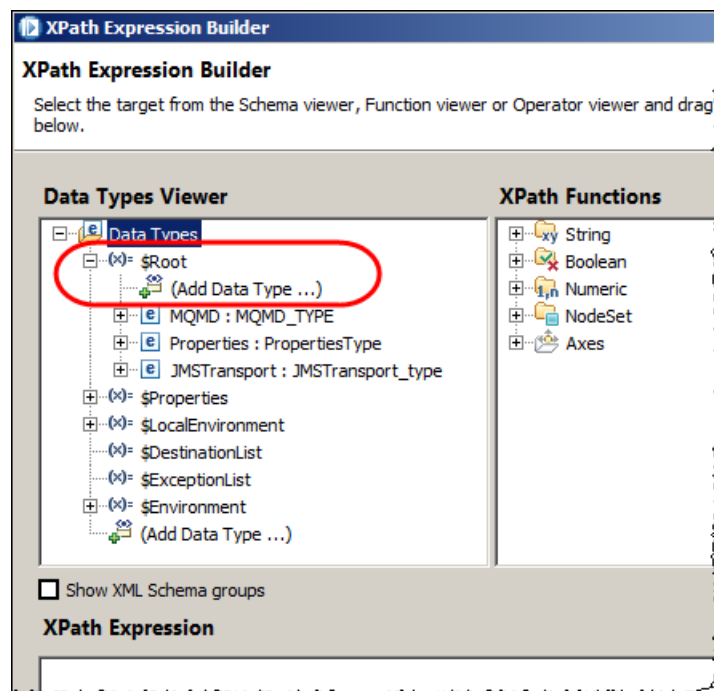
R variable name: **newdata**
R variable type : **Data Frame**
Direction : **In**

For Data location click the Edit button:

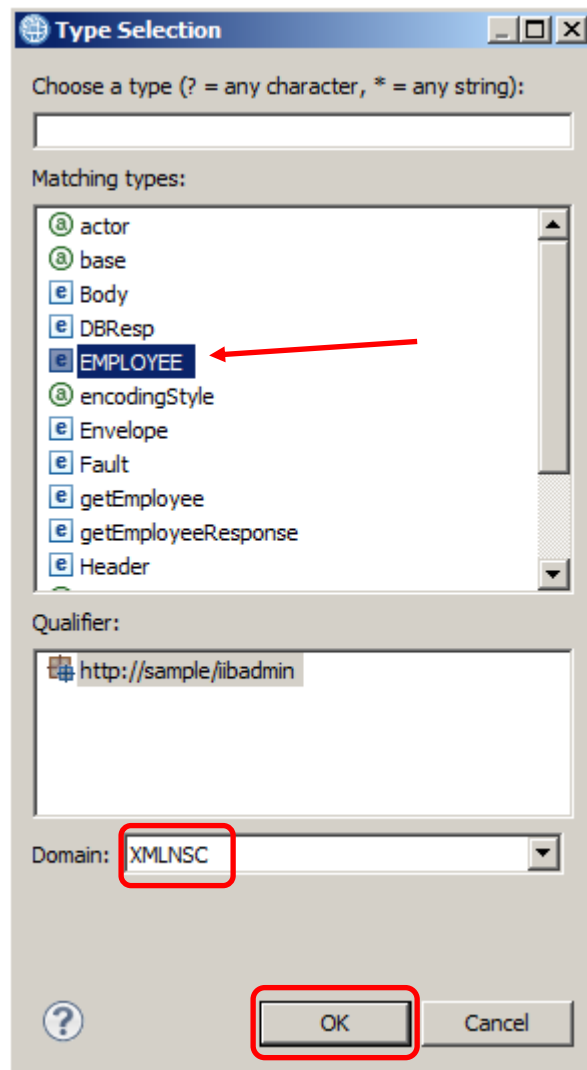


2. The XPath Expression Builder will open.

In the **Data Types Viewer** section, expand **\$Root** and click “**(Add Data Type)**”



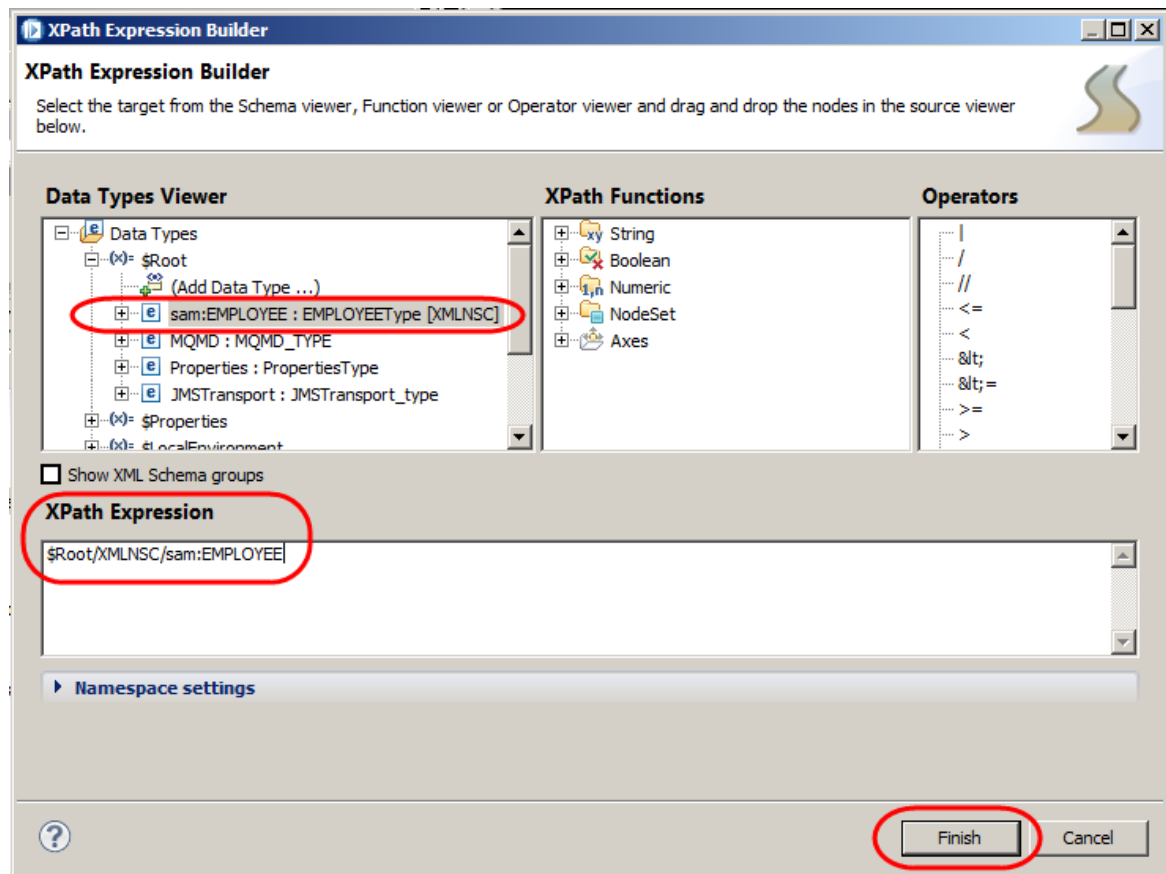
3. In the “**Type Selection**” window,
 - 1) highlight **EMPLOYEE**
 - 2) Change the Domain to **XMLNSC**
 - 3) and click **OK**:



- The EMPLOYEE data type appears under \$Root.

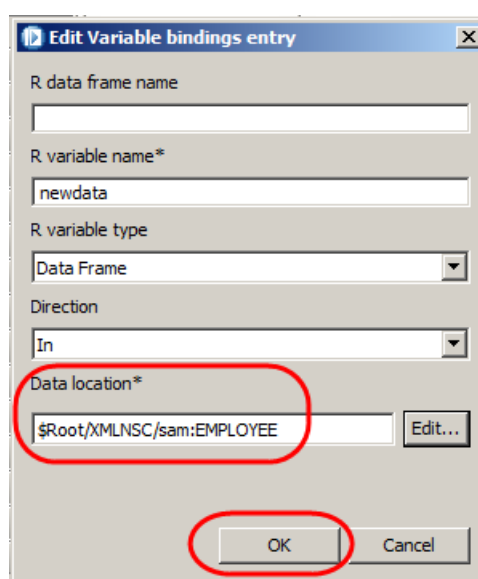
Double click on **sam:EMPLOYEE:EMPLOYEEType [XMLNSC]** to set the value in the XPath Expression

Click Finish to save the expression:

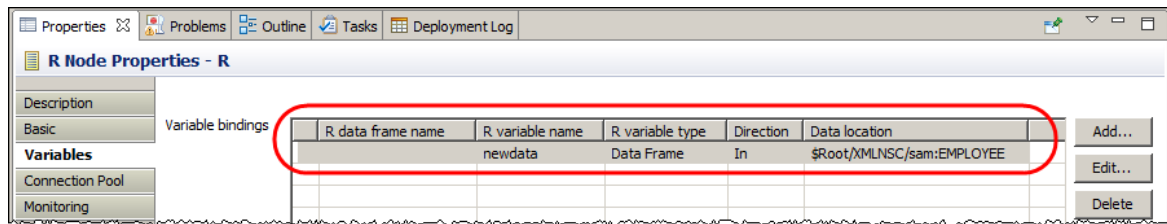


- The Data location field will be automatically completed with the XPath expression.

Click OK to add the variable:



6. The variable bindings table will be updated with the definition for the newdata “Data Frame”:



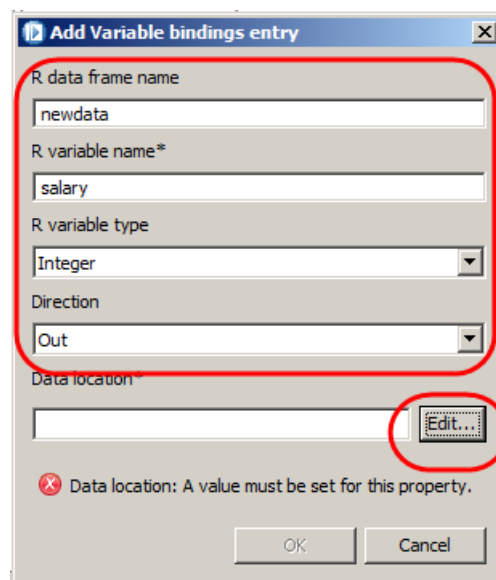
7. Click the Add button to add a second variable:

Click the “Add...” button to add a **second variable**, the “Add Variable bindings entry” window will appear.

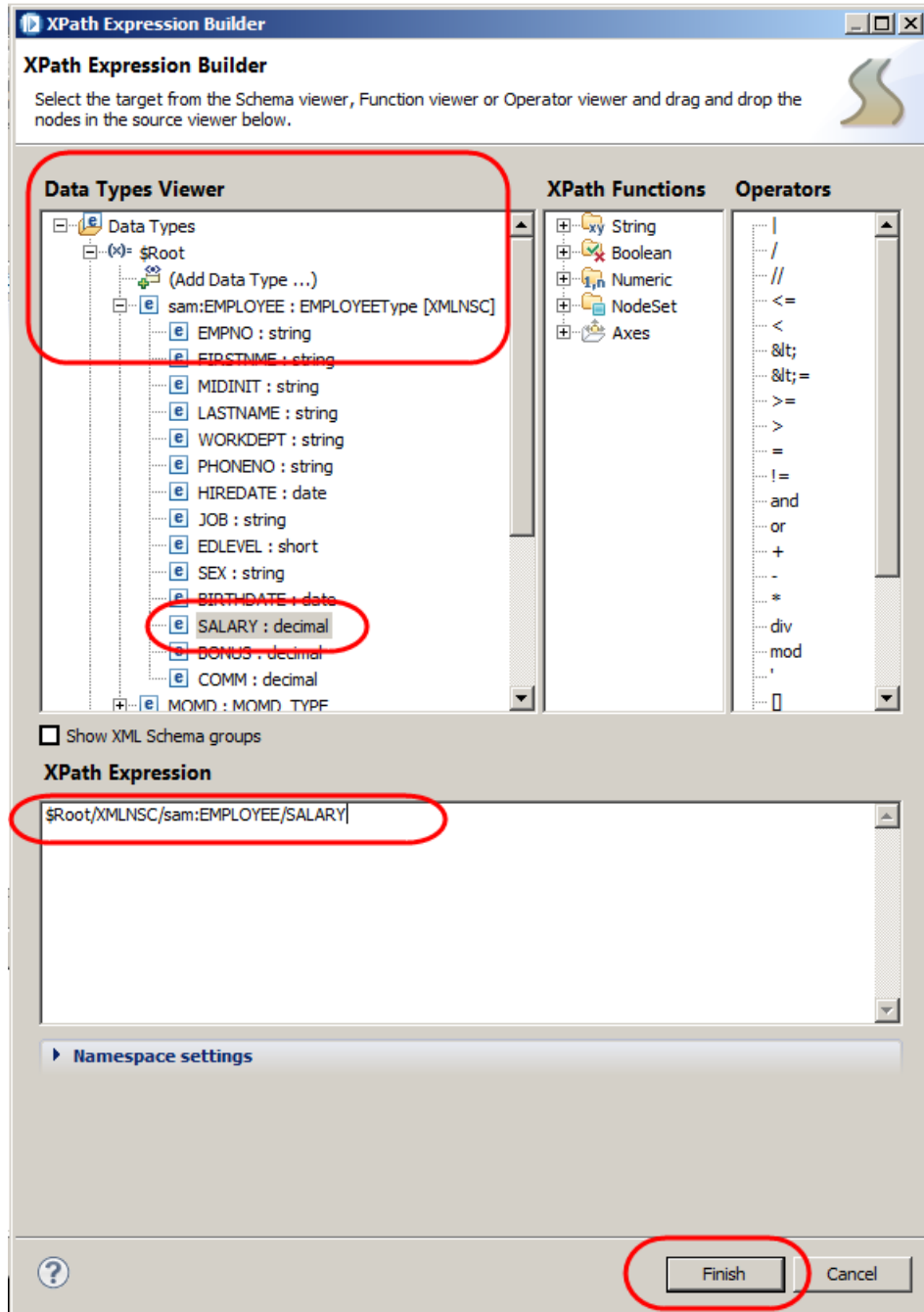
Specify the following (case sensitive):

R data frame name : **newdata**
 R variable name : **salary**
 R variable type : **Integer**
 Direction : **Out**

For Data location click the Edit button:



- 8. In the XPath Expression Builder expand \$Root until you can select SALARY. Double click on Salary to formulate the expression. Click Finish:



9. The Data location will automatically be filled with the location of SALARY in the message tree, Press OK to save the variable:

The screenshot shows the 'Add Variable bindings entry' dialog box with the following fields:

- R data frame name: newdata
- R variable name*: salary
- R variable type: Integer
- Direction: Out
- Data location*: \$Root/XMLNSC/sam:EMPLOYEE/SALARY

The 'OK' button is circled in red.

10. Click the “Add...” button to add a **third variable**, using the same method you used for the previous two variables.

Specify the following (case sensitive):

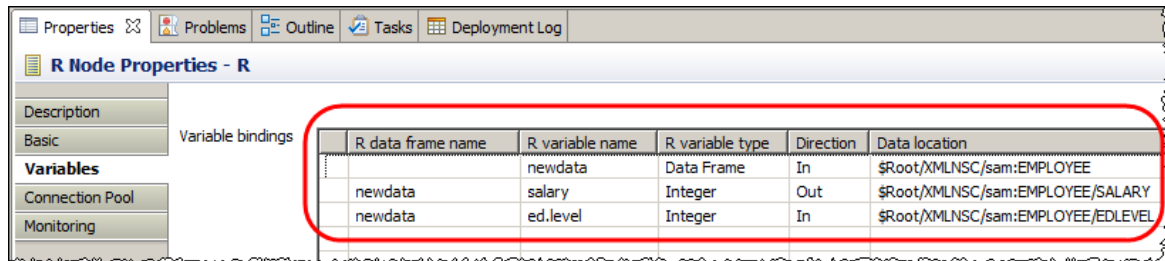
R data frame name: **newdata**
 R variable name : **ed.level**
 R variable type : **Integer**
 Direction : **In**
 Data location : **\$Root/XMLNSC/sam:EMPLOYEE/EDLEVEL**

The screenshot shows the 'Add Variable bindings entry' dialog box with the following fields:

- R data frame name: newdata
- R variable name*: ed.level
- R variable type: Integer
- Direction: In
- Data location*: \$Root/XMLNSC/sam:EMPLOYEE/EDLEVEL

The 'OK' button is circled in red.

11. The variables table will look like the following when all required variables for the model we are using are defined:



R data frame name	R variable name	R variable type	Direction	Data location
newdata	salary	Integer	Out	\$Root/XMLNSC/sam:EMPLOYEE/SALARY
newdata	ed.level	Integer	In	\$Root/XMLNSC/sam:EMPLOYEE/EDLEVEL

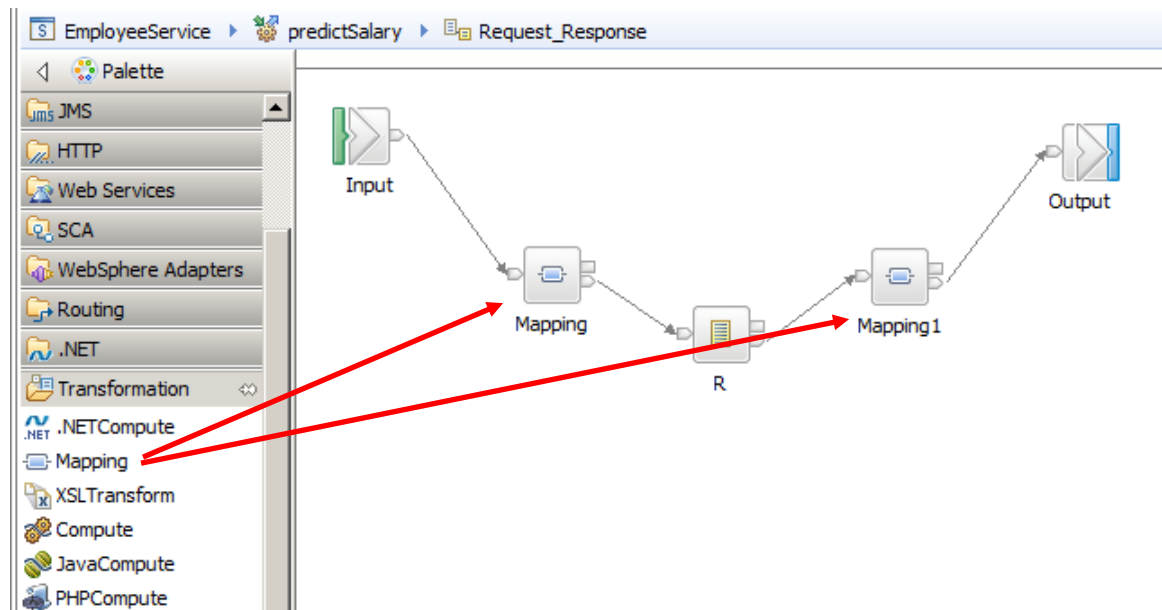
12. Save the message flow <ctrl s>, keeping the message flow open in the Integration Toolkit.

3.3 Add Mapping Nodes

1. The new operation must make sure that the data that is sent to the R server is in the correct format. The rule was defined to operate on the EMPLOYEE schema, so the predictSalary operation must send the data in this format.

To do this, a Mapping node will be added to transform the incoming message from the predictSalary SOAP message to the EMPLOYEE schema format. A second Mapping node will be added to perform the reverse transformation.

In the predictSalary operation, add two mapping nodes, and connect them as shown.

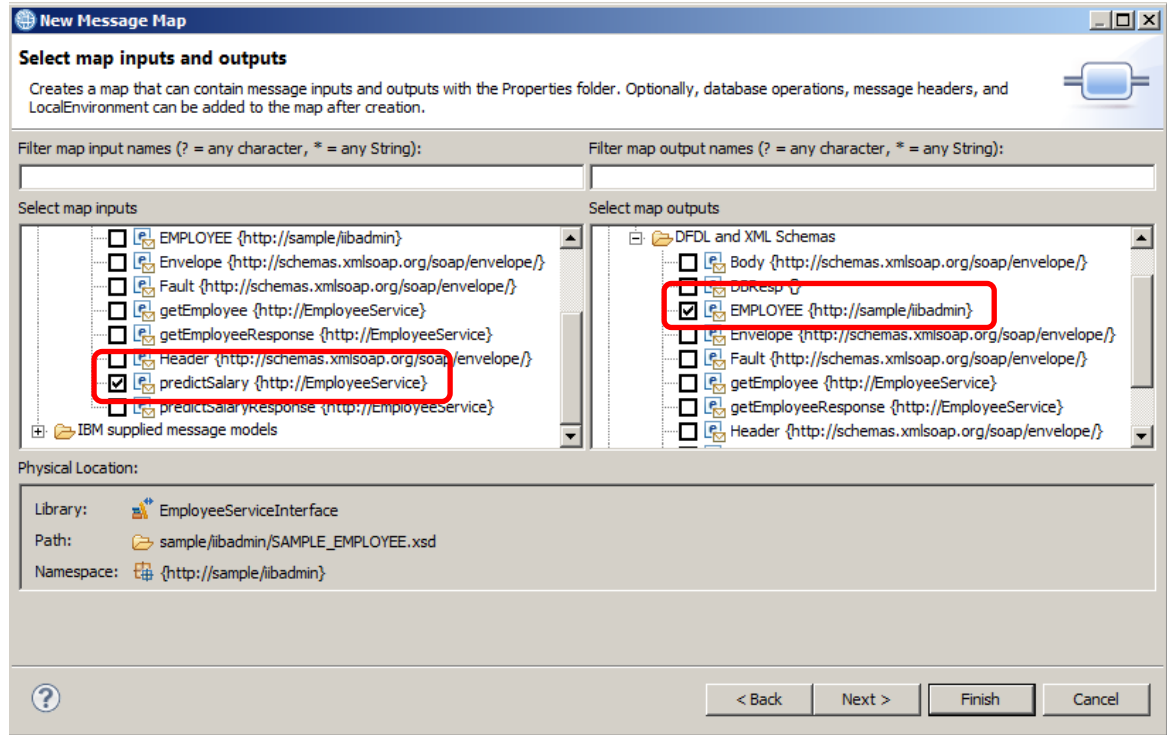


- Open the first mapping node, and click Next on the first dialogue window.

At the map inputs and outputs window:

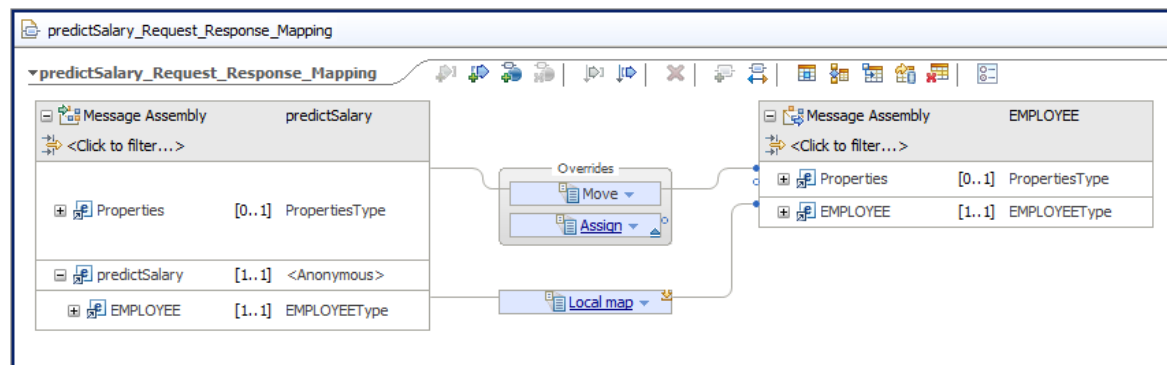
- Select the input message = predictSalary
- Select the output message = EMPLOYEE

Click Finish.



- Expand predictSalary, and map the input EMPLOYEE to the output EMPLOYEE.

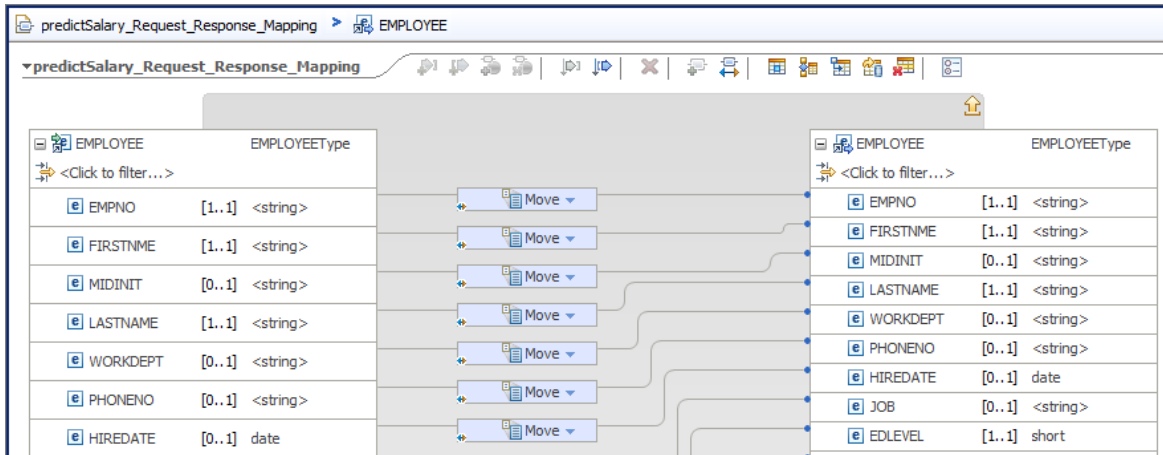
Change the transform type to "Local map".



- Click "Local map" to define the individual mappings.

Use the automap feature to map each input element to the same output element.

Save and close the map.

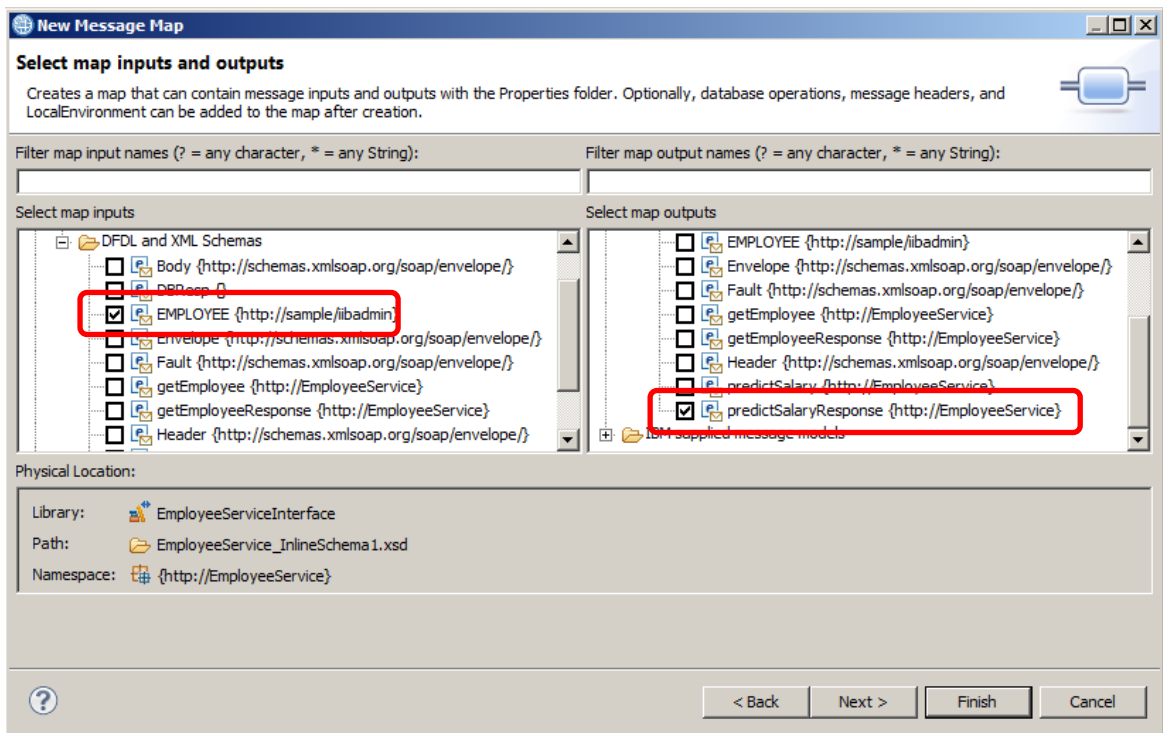


- Open the second map, and click Next on the first dialogue window.

At the map inputs and outputs window:

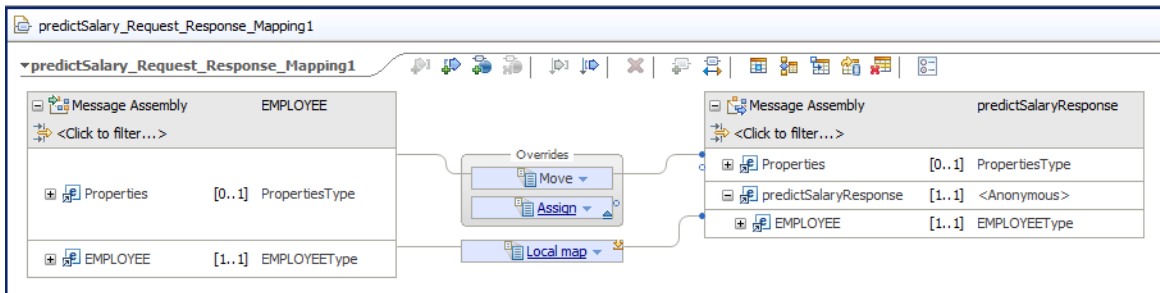
- Select the input message = EMPLOYEE
- Select the output message = predictSalaryResponse

Click Finish.



- Expand predictSalary, and map the input EMPLOYEE to the output EMPLOYEEE.

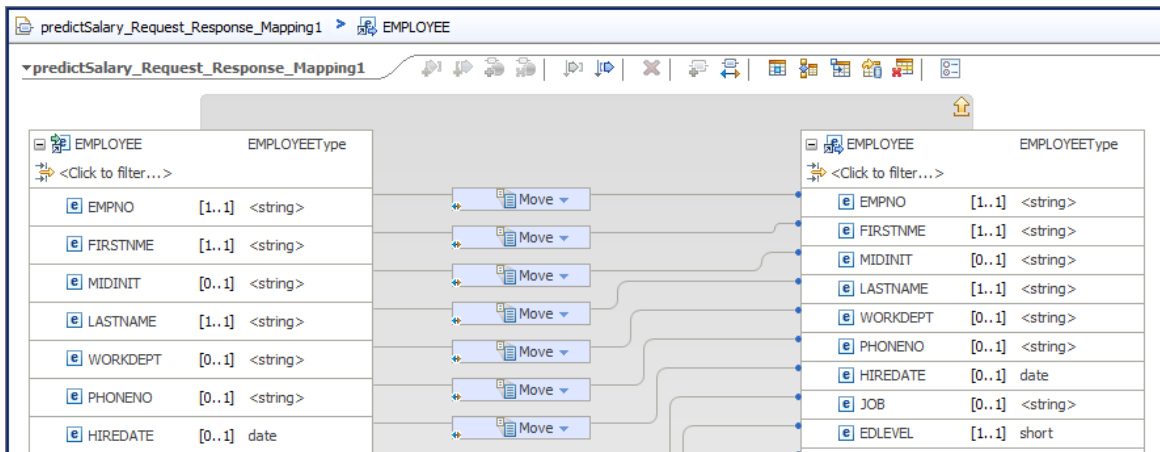
Change the transform type to "Local map".



- Click "Local map" to define the individual mappings.

Use the automap feature to map each input element to the same output element.

Save and close the map.

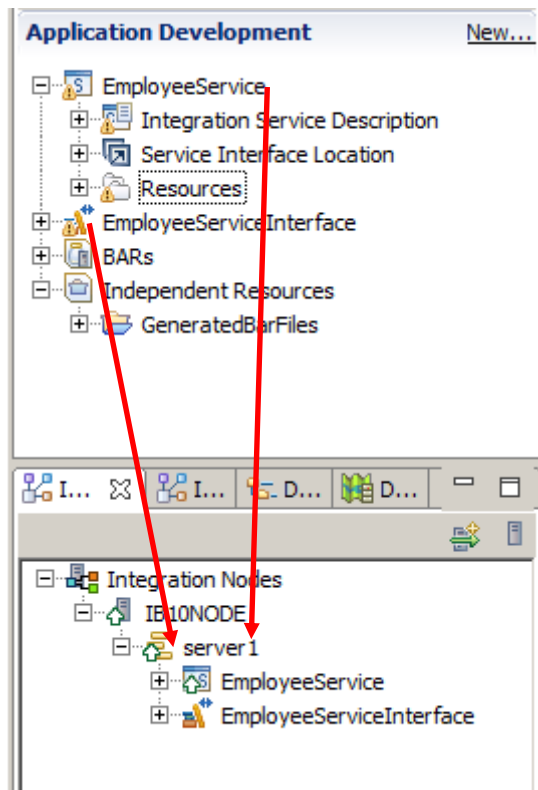


- Save the updated message flow

4. Testing

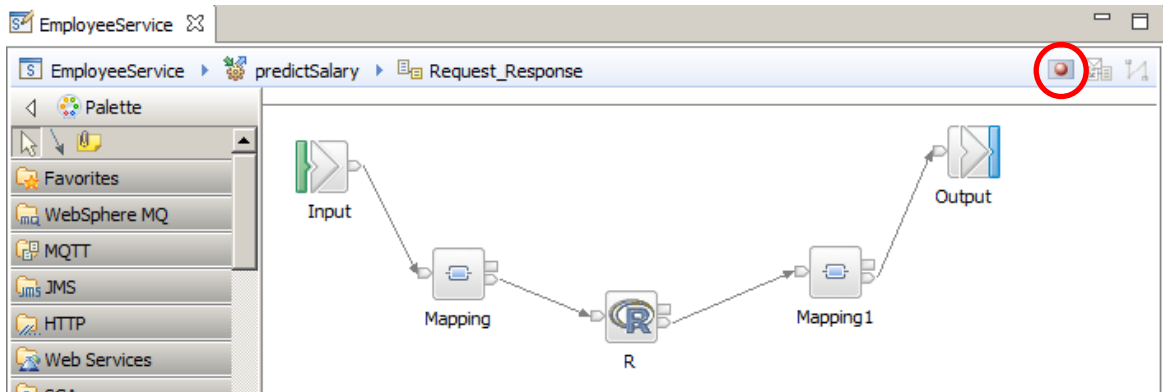
4.1 Deploy the service

1. Save the updated message flow and deploy **both** the EmployeeServiceInterface library and the EmployeeService service in the usual way.

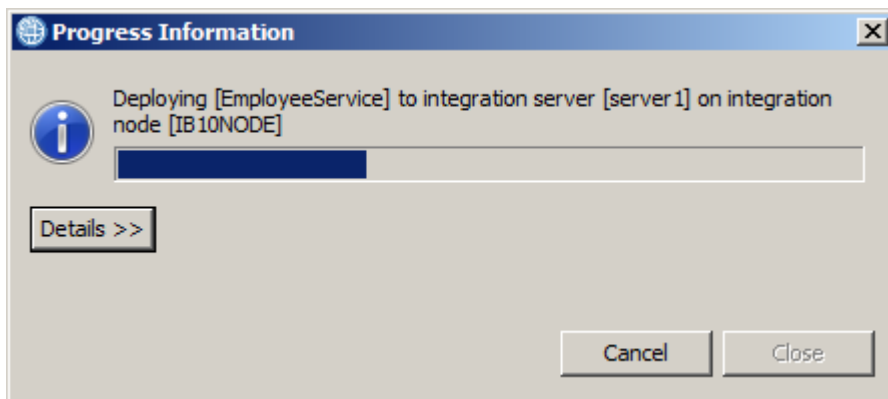


4.2 Test the service with the IIB flow exerciser

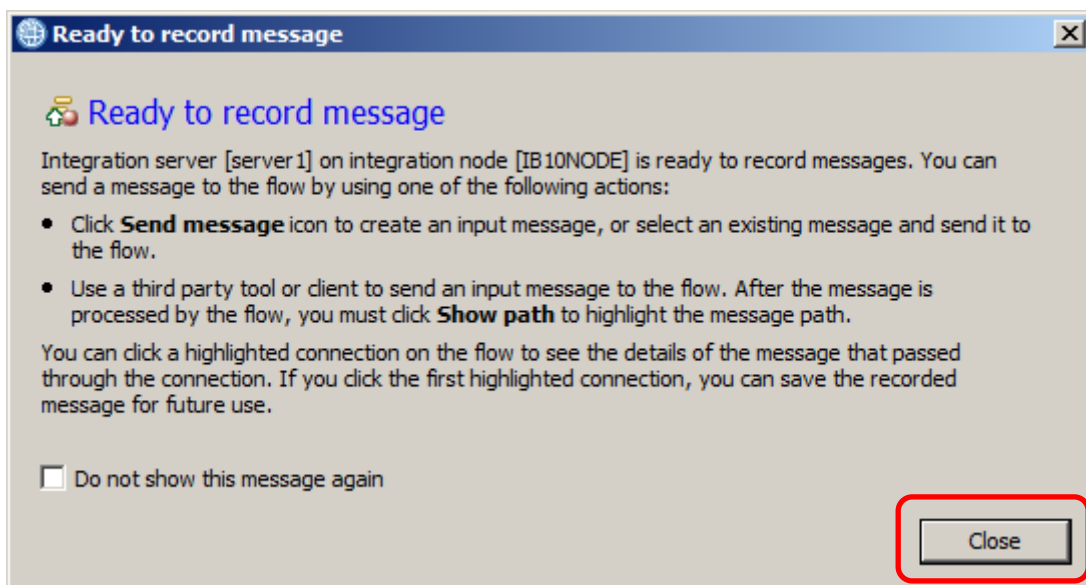
1. Open the **predictSalary Request_Response** subflow that you created earlier. Click the red button to start the flow exerciser,



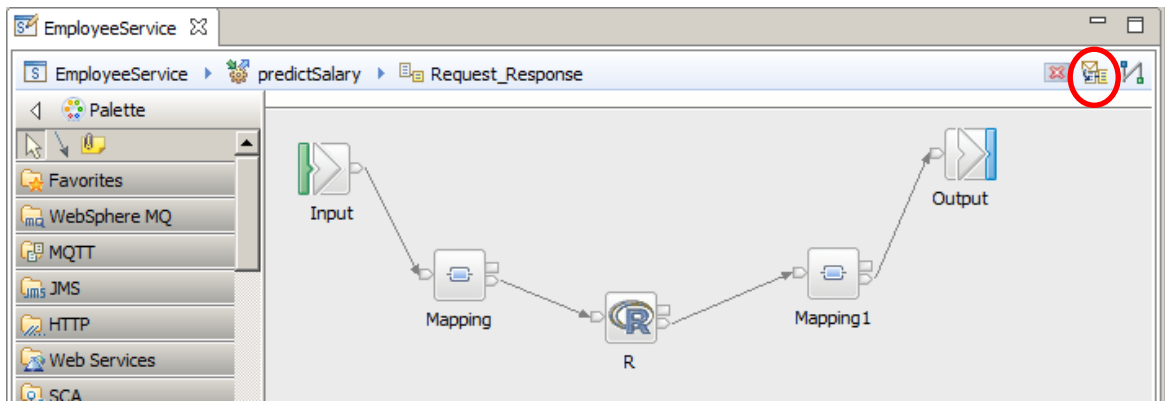
2. The service will be deployed.



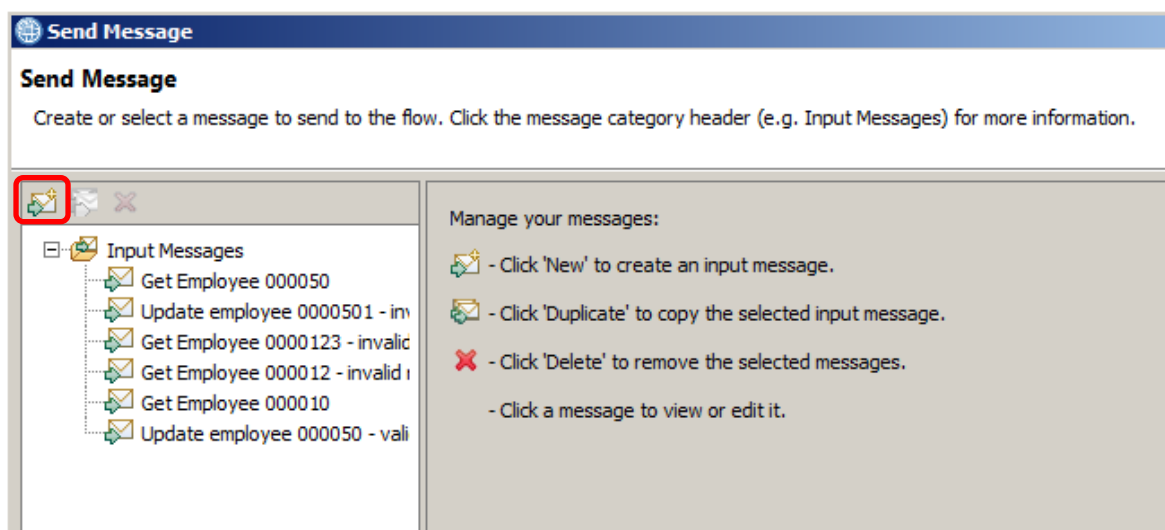
3. When the service has been deployed, a message box will appear as below, to show that the integration server is ready to record messages. Click on the **Close** button to continue.



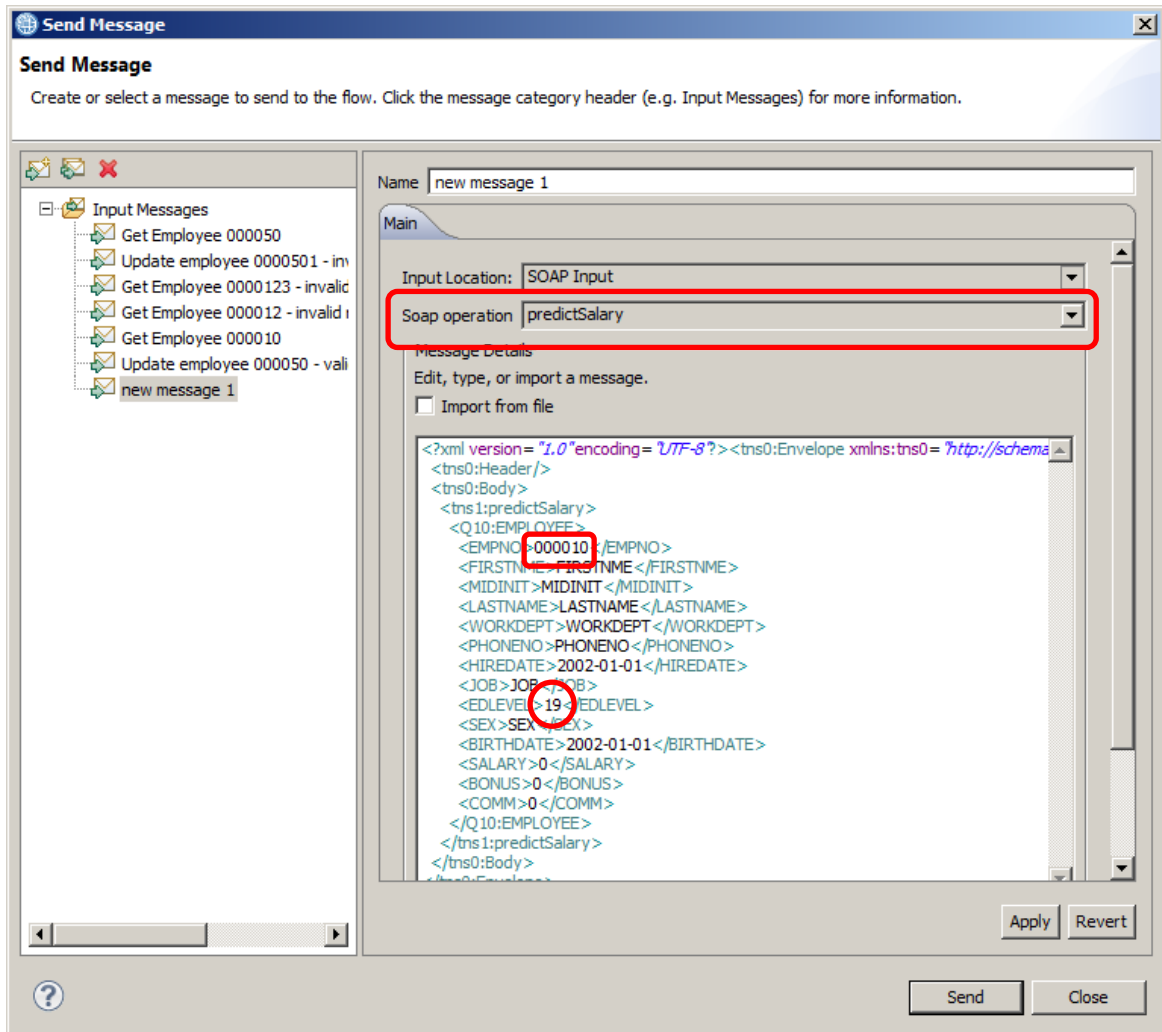
4. Click the **send message** icon



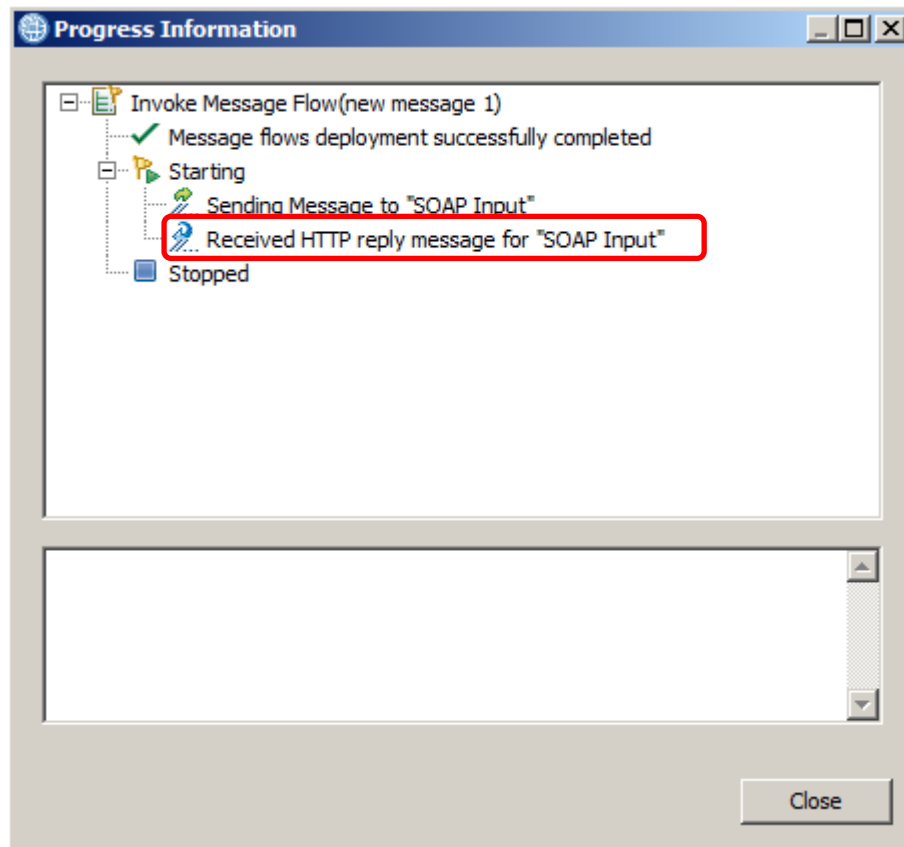
5. Click on **new message**



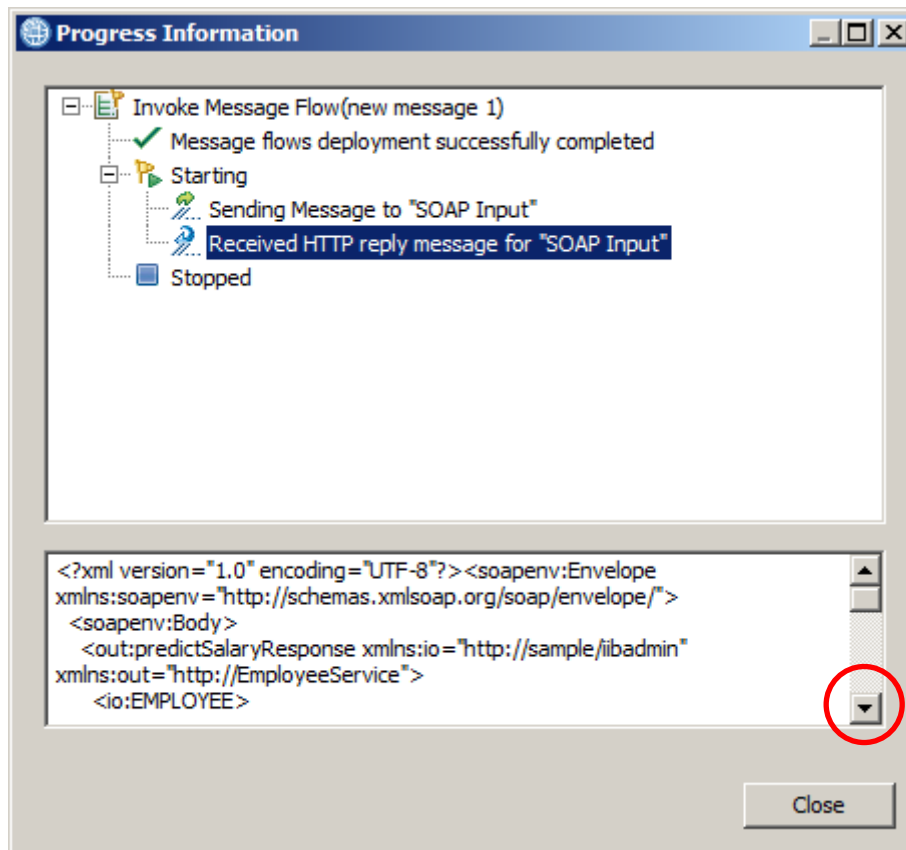
6. Select **predictSalary** under **Soap operation**, enter an employee number of '000010' and an education level of '19'. Then click **Send**. This will send a SOAP message to the predictSalary operation.



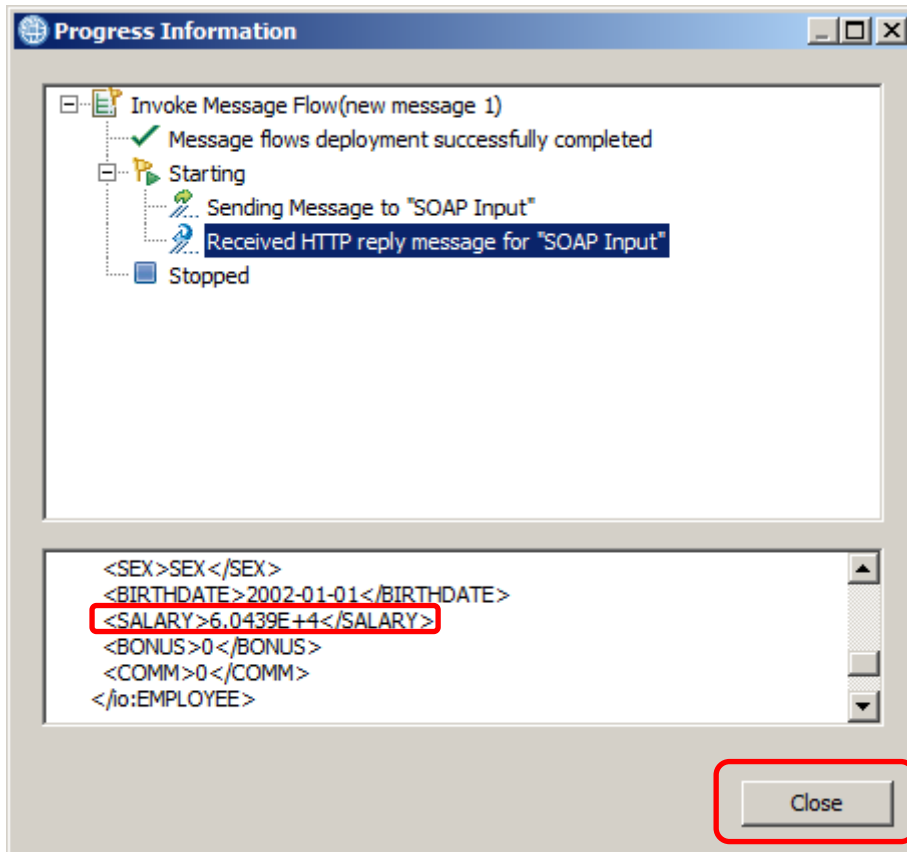
7. After a few seconds, an HTTP reply message should be received. Highlight and click on the notification.



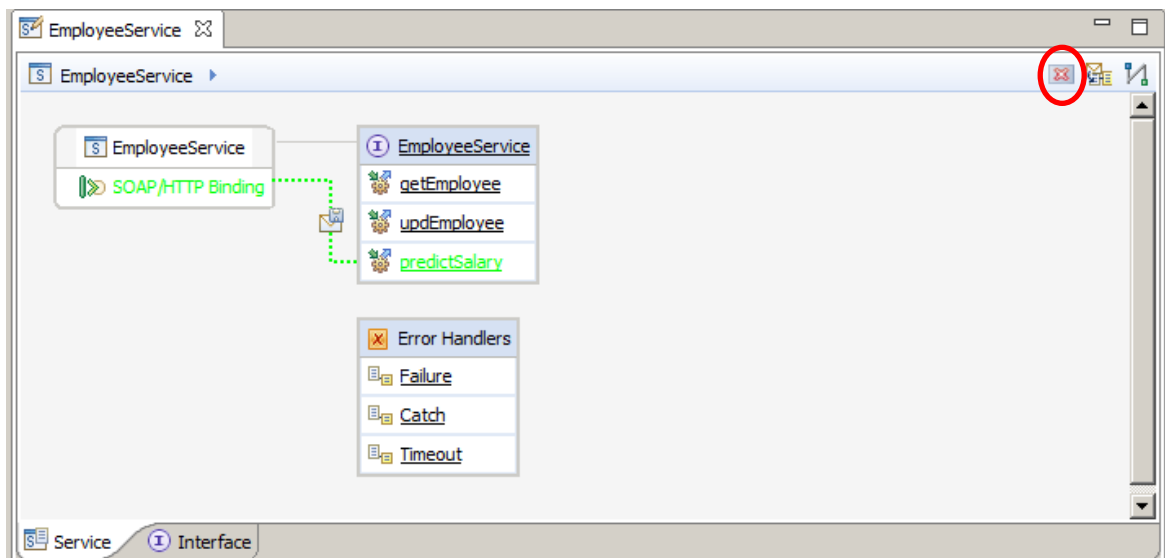
8. The HTTP reply details should be displayed in the lower pane. Scroll down to see the details.



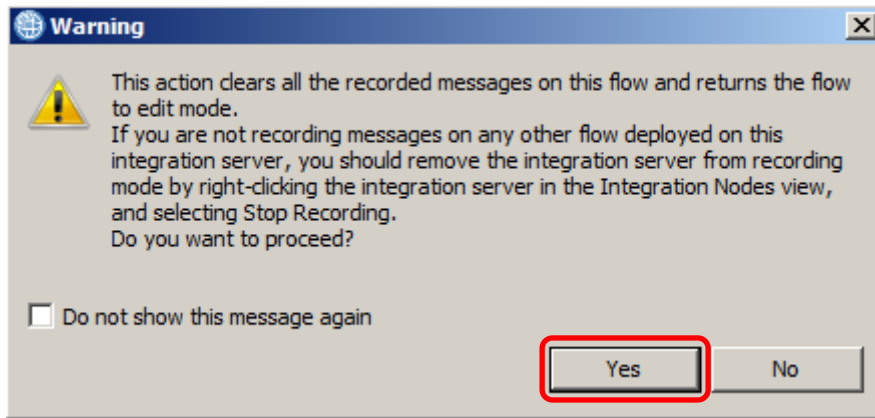
9. You should see that the record for employee number '000010' has been returned. However, the salary is the one predicted for the education level you supplied. Close the window when you are ready.



10. You will be returned to the EmployeeService view. Return the flow to edit mode by clicking on the red icon.



11. A warning message will pop up. Click on **Yes** to continue.



4.3 Understanding the value returned from R

The R node support in IIB V10 provides an interface for IIB to have the facility to call out to an R environment. Three scripting based interfaces are provided with the R node:

- 1) Connection script: Run when IIB initially connects to RServe.
- 2) Evaluate script: Run (per message) when a message reaches the R node.
- 3) Disconnect script: Run when the Integration Server stops or the application is redeployed.

We have used a very simple R model that predicts a value for SALARY based on a value for EDLEVEL. The data contained in the R model you used above, was based on an extract from the EMPLOYEE table in the DB2 SAMPLE database.

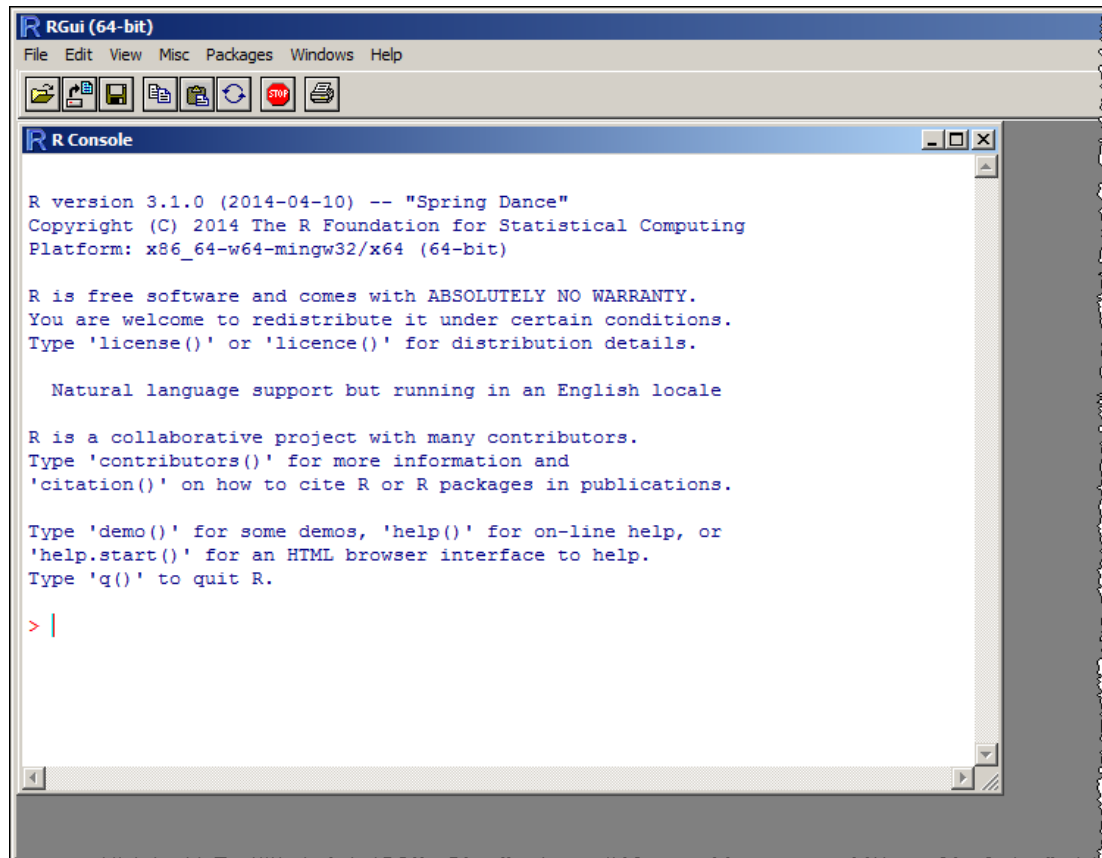
You will now use some provided R scripts to graphically represent how the model works and how the SALARY number was derived by the R model.

4.4 Analyze the model before and after running the application

You will now use R to show a graphical representation of the model before and after you run the application.

1. Open the R Gui interface (Start > All Programs > R > R x64 3.1.0).

RGui(64-bit) will open with an R Console.



2. In the console type `setwd("c:/student10/Analytics/RWork")`

(Note the **Forward slashes**).

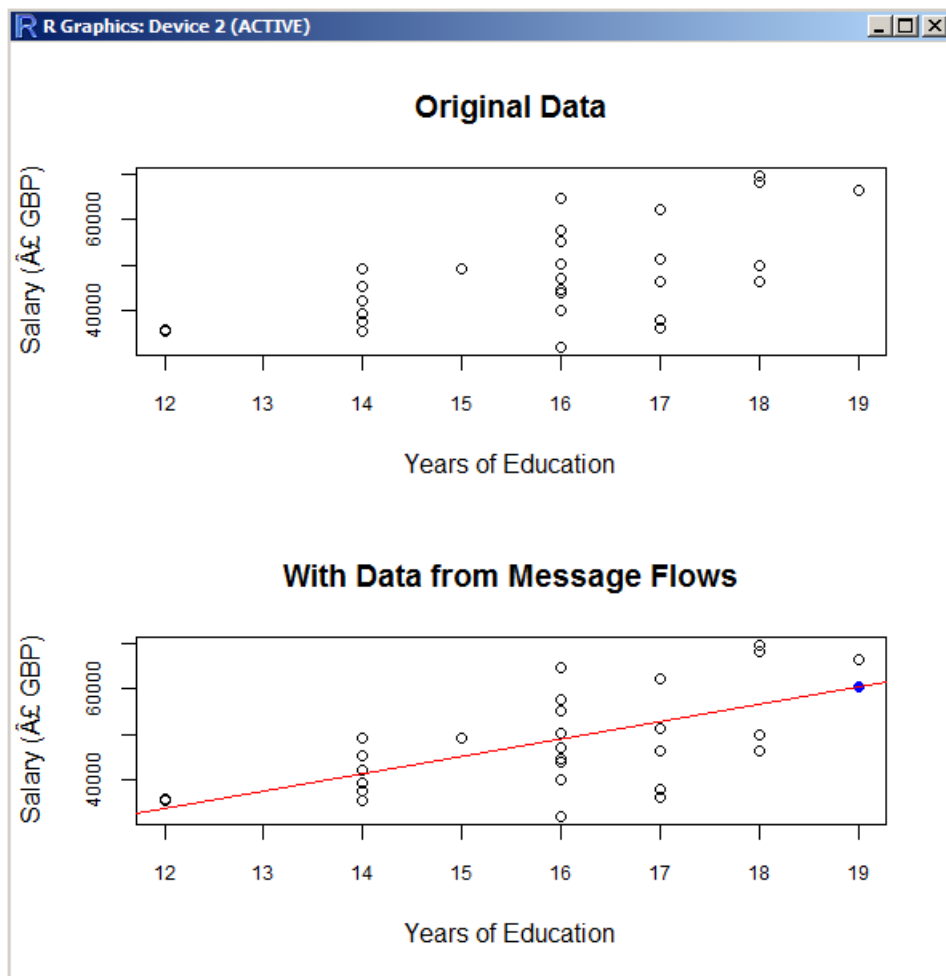
The command sets the working directory for the R console session, you are setting the value of this working directory to where it can find scripts to run in the student10 directory.

If the command works there will be no response:

```
> setwd("c:/student10/Analytics/RWork")
> |
```

3. Type source `source("DrawGraphs.r")` and press enter

4. A window with two graphs will open in RGui :



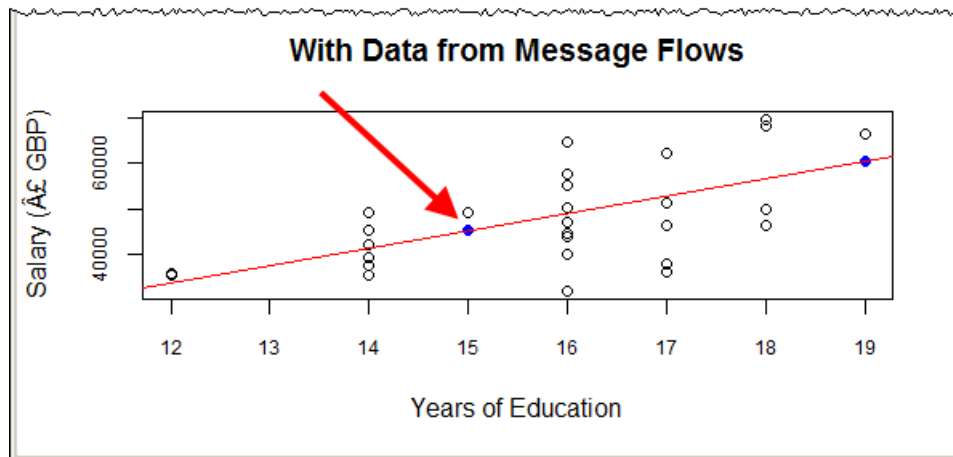
5. The first graph shows the original data that was built with the model.
The second graph shows a (Red) line. This is a (very simple) line of best fit to the data.
The (blue) mark at the red line shows how the value for SALARY was predicted given the EDLEVEL value of 19 (the SALARY value is taken from the Y (axis)).
6. Go back to the flow exerciser and send a message with EDLEVEL=15.
Click the green arrow to execute the predictSalary.
A value of 4.524E+4 will be returned in the SALARY field:

HIREDATE	2001-01-01
JOB	
EDLEVEL	15
SEX	F
BIRTHDATE	2001-01-01
SALARY	4.524E+4
BONUS	0
COMM	0

7. Switch back to the R console and re run the DrawGraphs script

The command is `source("DrawGraphs.r")`
(highlight the R console and press the up arrow to retrieve previous commands)

8. A blue mark will now appear on the line (above 15 on the X axis).



Again the value passed back to the message flow was taken from the Y axis of the graph.

END OF LAB GUIDE