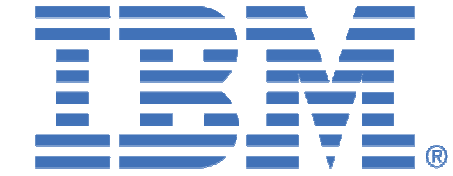


IBM Integration Bus

Using Callable Flows

Featuring:

CallableInput and Callable Reply nodes
CallableFlowInvoke node

April 2016
Hands-on lab built at product
Version 10.0.0.4

1. Introduction

IIB V10.0.0.4 introduces Callable Flows. This enables a message flow (or integration service, or REST API) to invoke a separate message flow in a call/return (blocked wait) programming model.

The calling and called message flows operate on the same message tree, so there is no requirement for any protocol-related components. For example, the message flows simply access message tree elements, and there is not requirement for any knowledge of network location of invoked components.

1.1 Scenario Overview

In this lab, you will make the necessary configurations in SQL Server and IIB to enable an IIB application (integration service) to connect to SQL Server, and retrieve some rows from a specific SQL Server table.

The integration service is provided for you, and is the solution of the integration service that was developed in Lab 1 (Create Integration Service) in this series of labs.

1.2 Outline of tasks

The tasks to complete in this lab are the following:

1. Import a partially-built shared library and applications
2. Complete the development of a new map and two message flows
3. Deploy and test.

1.3 Configure Integration Bus node to work with DB2

If you have already done Lab 1 in this series (create an Integration Service), you can skip straight to Develop the Applications on the next page.

To run this lab, the Integration Bus node must be enabled to allow a JDBC connection to the HRDB database.

1. Open an IIB Command Console (from the Start menu), and navigate to

   `c:\student10\Create_HR_database`

2. Run the command

   `3_Create_JDBC_for_HRDB`

   Accept the defaults presented in the script. This will create the required JDBC configurable service for the HRDB database.

3. Run the command

   `4_Create_HRDB_SecurityID`

4. Stop and restart the node to enable the above definitions to be activated

   `mqsistop TESTNODE_iibuser`

   `mqsistart TESTNODE_iibuser`

This will create the necessary security credentials enabling TESTNODE_iibuser to connect to the database.

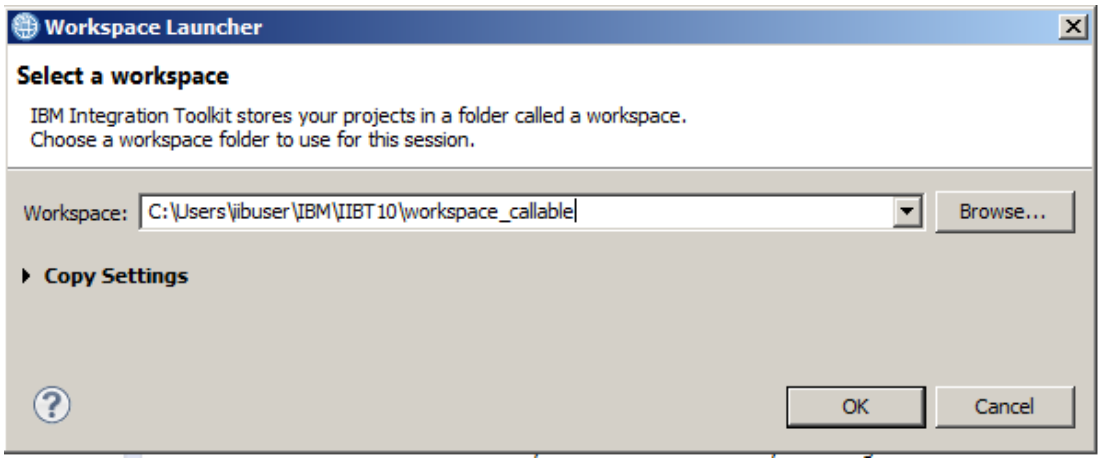
Recreating the HRDB database and tables
The HRDB database, and the EMPLOYEE and DEPARTMENT tables have already been created on the supplied VMWare image. If you wish to recreate your own instance of this database, the command **`1_Create_HRDB_database.cmd`** and **`2_Create_HRDB_Tables.cmd`** are provided for this. If used in conjunction with the VM image, these commands must be run under the user "iibadmin". Appropriate database permissions are included in the scripts to GRANT access to the user iibuser.

2. Develop the Applications

2.1 Import the provided resources

1. Login to Windows with the user **iibadmin**, password = **passw0rd**.

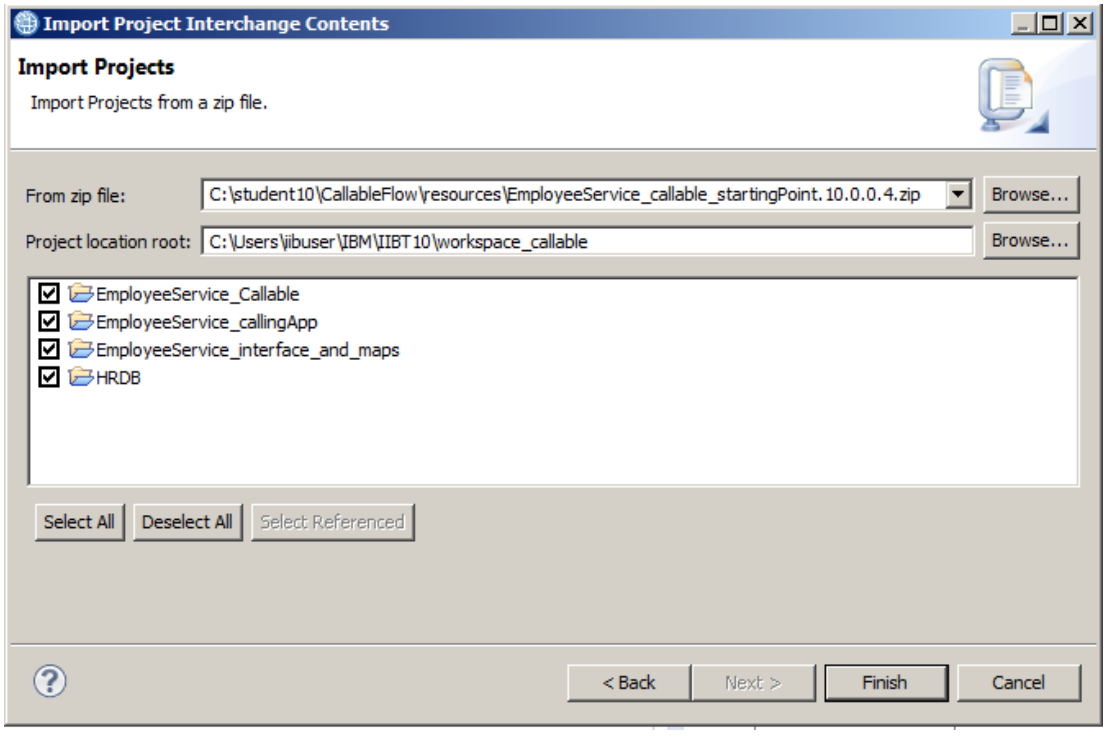
2. To ensure development artefacts do not overlap, create a new workspace with the name \workspace_callable.



3. Import the PI file

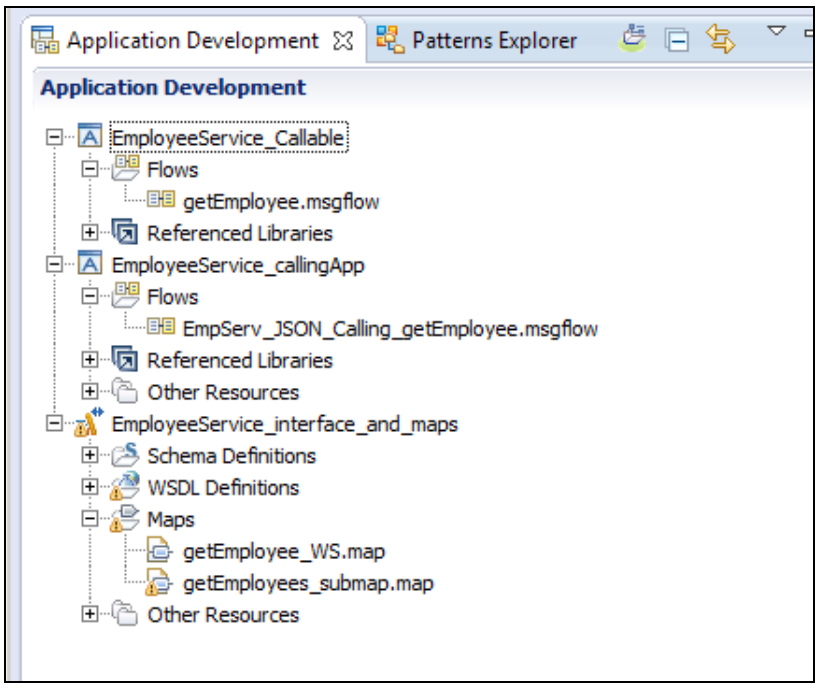
```
c:\student10\CallableFlow\resources\
EmployeeService_callable_startingPoint.10.0.0.4.zip
```

Ensure all projects are selected, and click Finish.



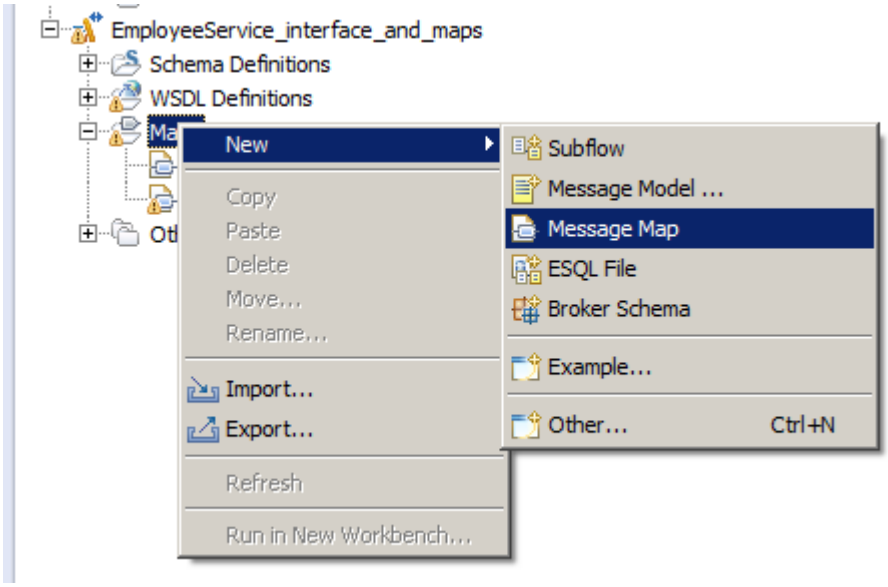
4. The project EmployeeService_interface_and_maps is the project that you have used in earlier labs. It contains the schemas that are required for the employeeNumber and EmployeeResponse elements, and the submap that will be used to retrieve data from the HRDB database.

   The applications **EmployeeService_callable** and **EmployeeService_callingApp** are pre-defined applications to save time during the course of this lab. You will extend the provided message flows in these two applications.

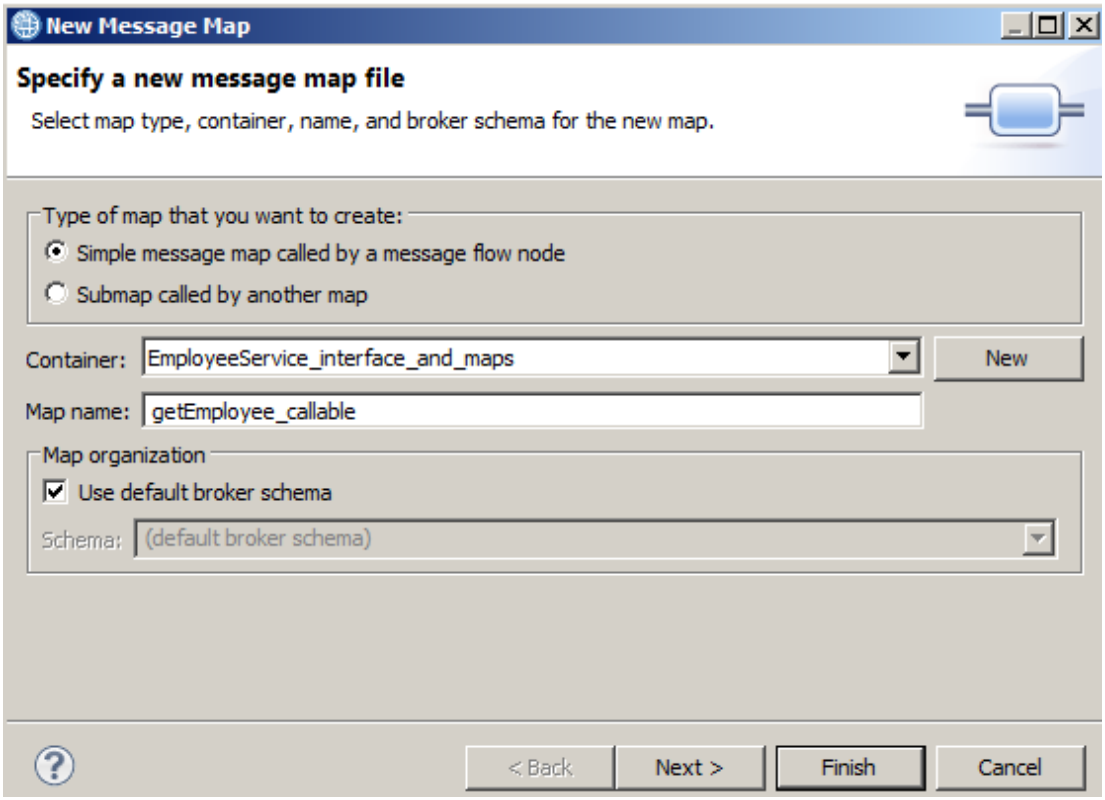


2.2 Create a new Message Map

1. In the EmployeeService_interface_and_maps library, right-click the Maps folder, and select New, Message Map.



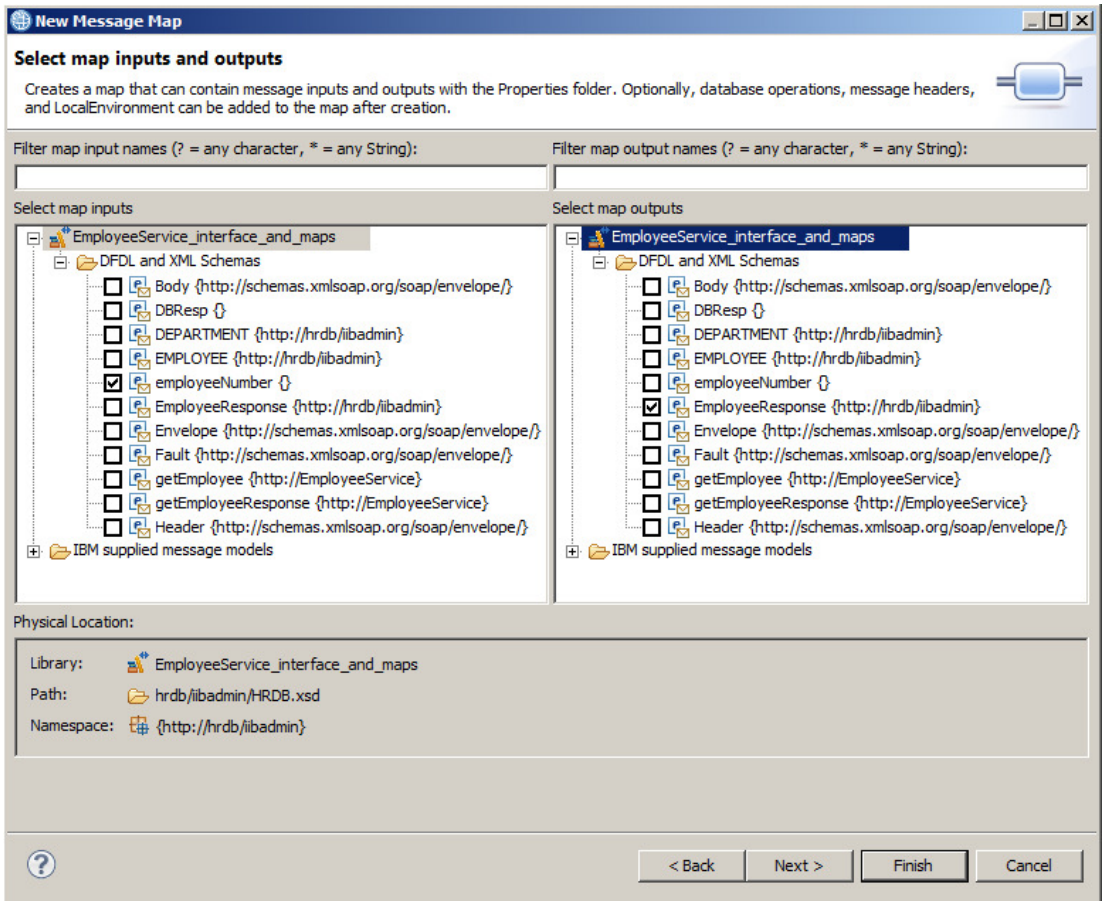
2. Name the new map getEmployee_callable, and click Next.



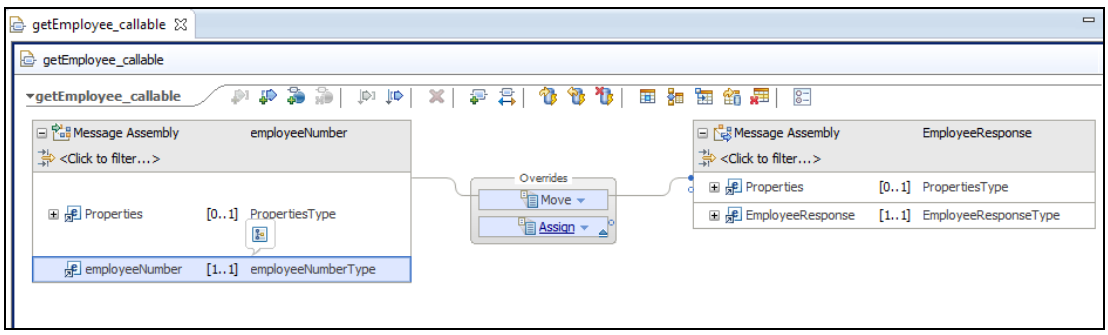
3. The input to this map will be employeeNumber, which will be a single element in the IIB message tree. Similarly, the output from the map will be the complex element EmployeeResponse. No protocol-related elements will be required to be processed by this map.

   - Set the map input to DFDL and XML schemas, employeeNumber.
   - Set the map output to DFDL and XML schemas, EmployeeResponse.

   Click Finish.

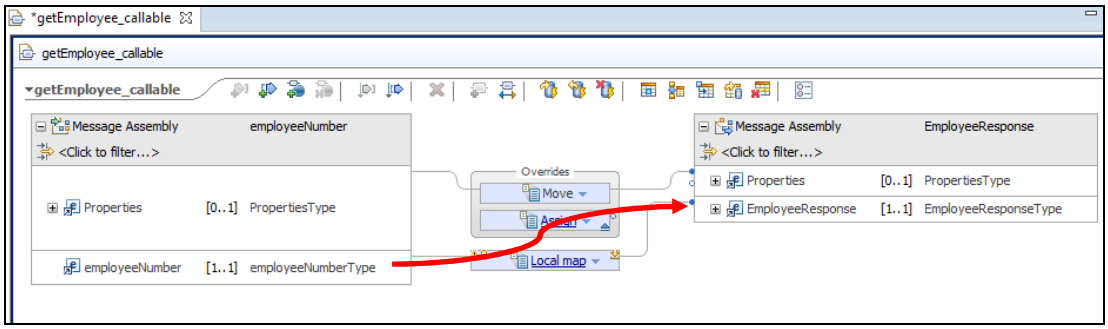


4. The input and output assemblies will be created as shown.

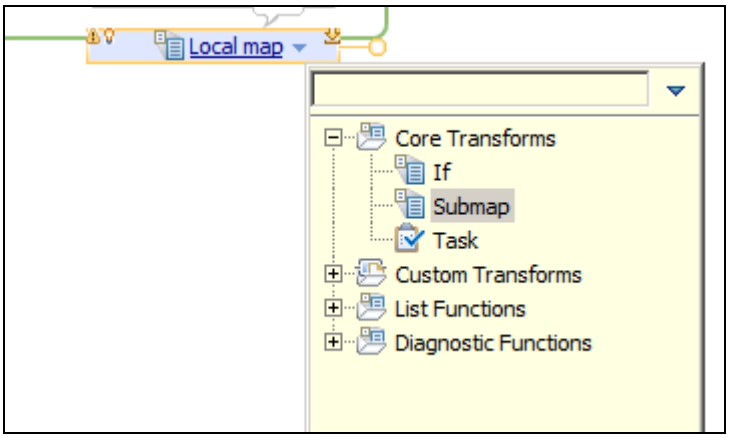


5. Connect the employeeNumber element to the EmployeeResponse element.

   This will create a Local map transformation.

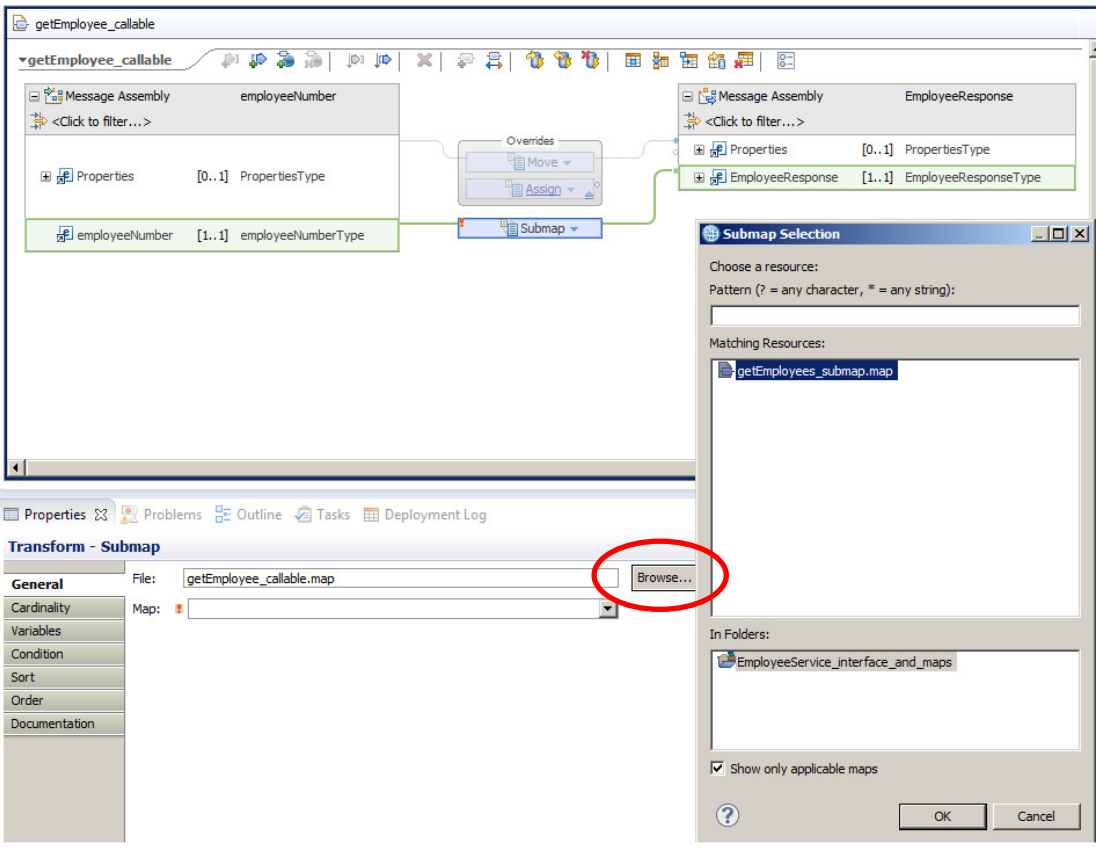


6. Change the Local map transformation to a Submap (click the blue down-arrow).



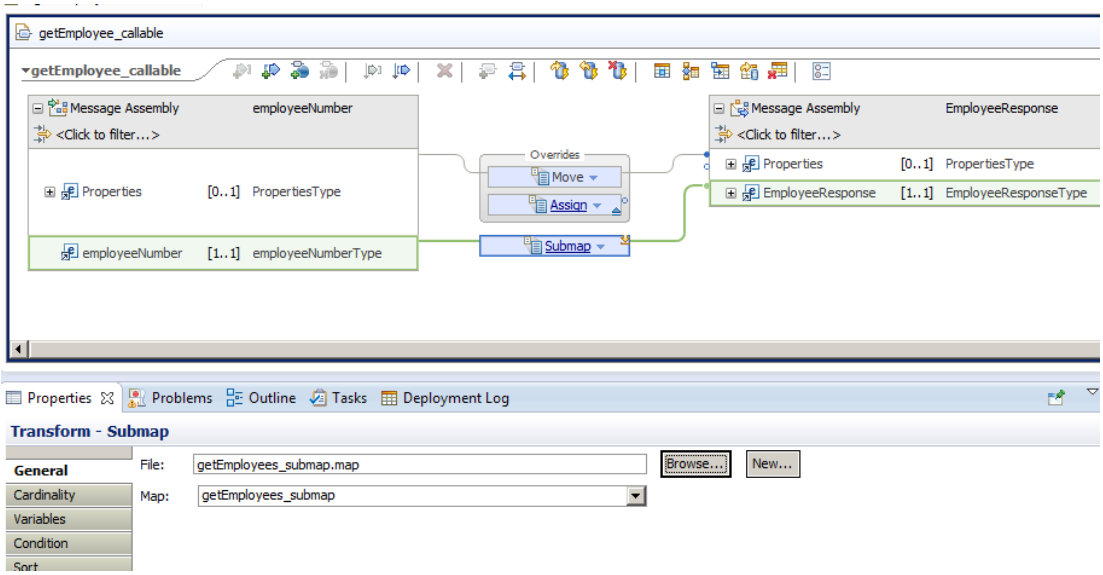
7. The map editor will detect that the Submap transformation requires the name of the submap file to be specified. In the Properties of the Submap transform, on the General tab, click the Browse button.

   The list of available submaps will be shown. Only one submap will be available; this is the submap that has been used in several of the labs in this series, so select **getEmployee_submap** and click OK.



8. The error on the Submap transform will have been resolved, and the General tab will show the name of the submap file and mapname that will be invoked by the submap transform.

   Save (Ctrl-S) and close the map.



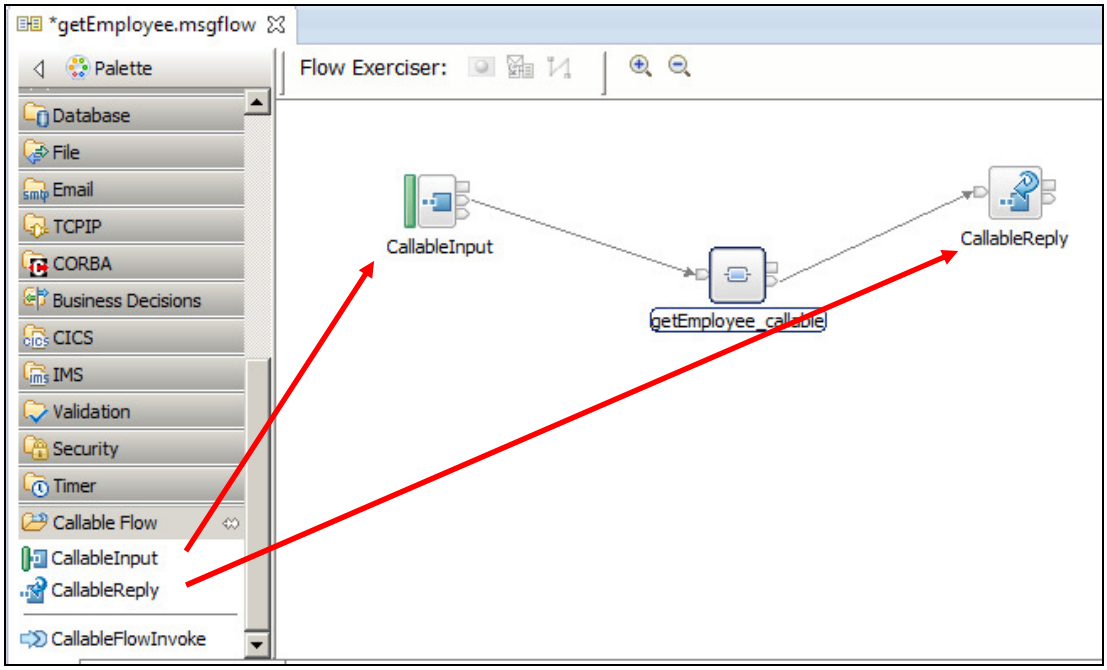
2.3 Create the Callable Message Flow

1. Expand the application EmployeeService_Callable, and open the message flow getEmployee.

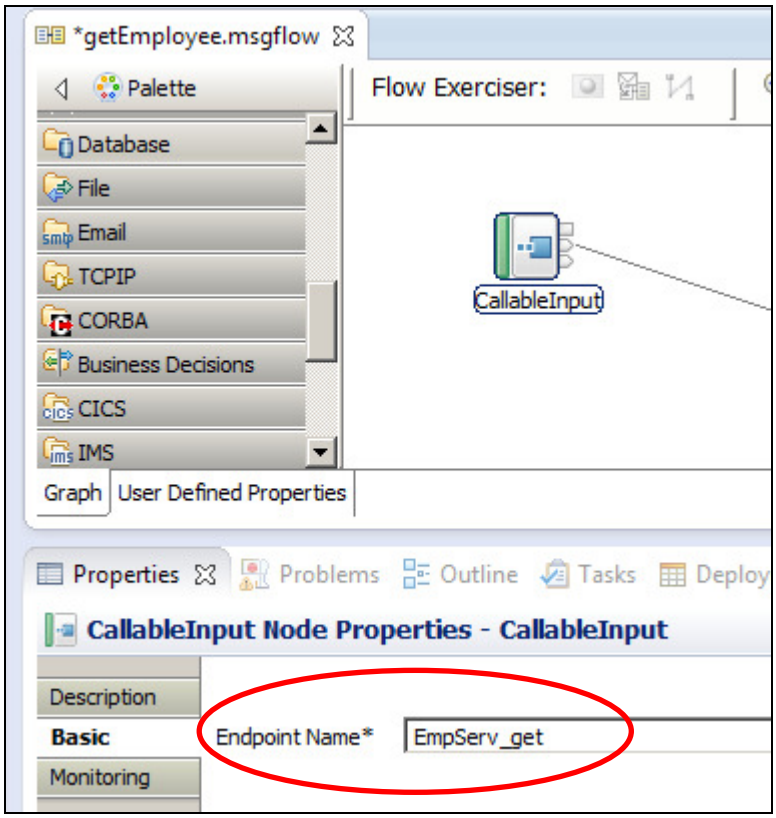
   The flow has been created for you, but has not been populated with any nodes.

   Drop three nodes onto the flow editor, and connect as shown:

   - CallableInput - from the Callable Flow folder
   - Mapping node - named it getEmployee_callable
   - CallableReply - from the Callable Flow folder

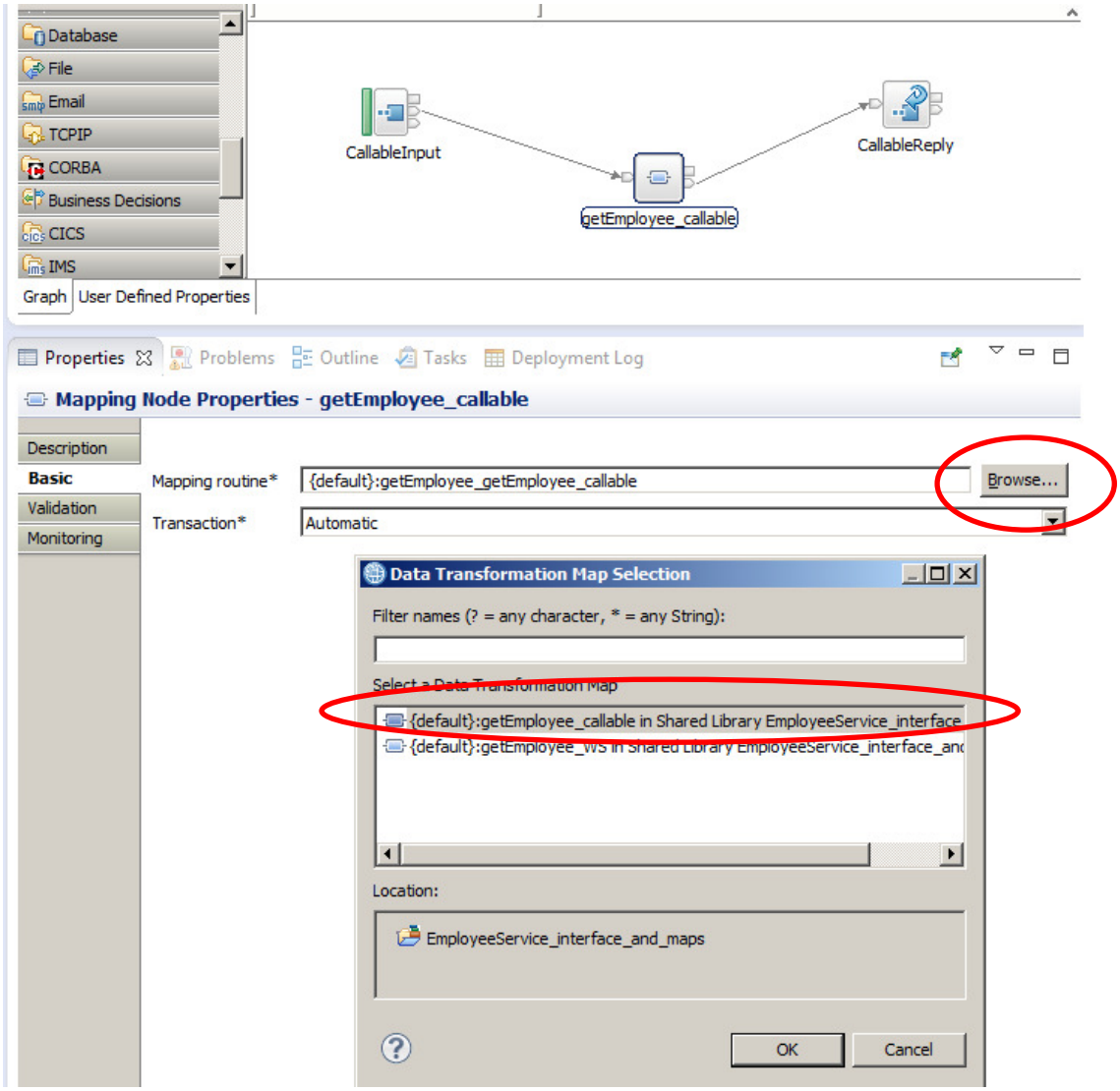


2. Highlight the CallableInput node and on the Basic Properties, set the Endpoint Name to **EmpServ_get**.

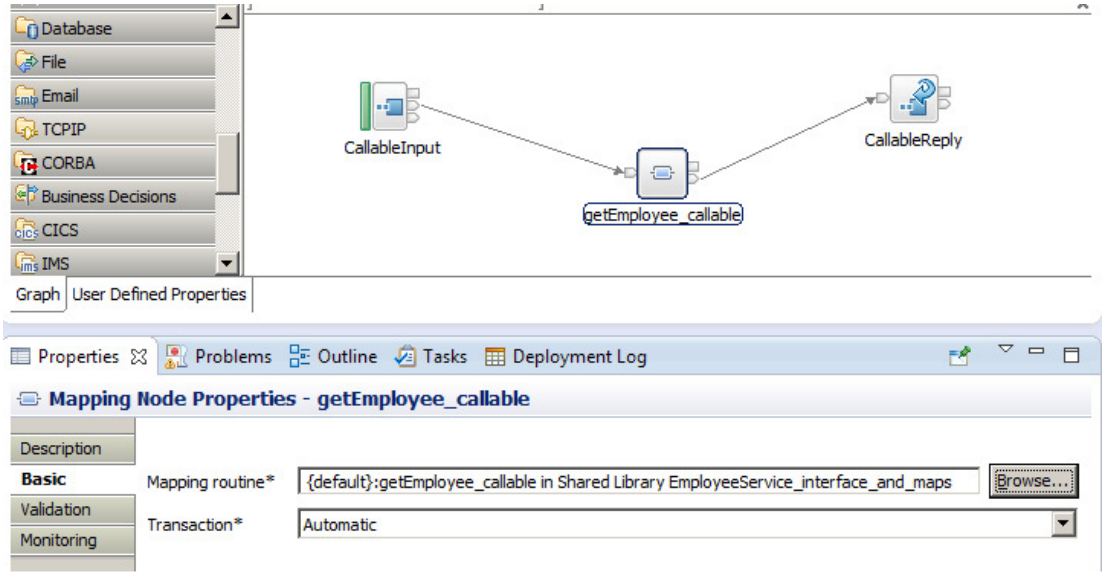


3. Highlight the Mapping node getEmployee_callable.

   In the properties of the map node, use the Browse button to set the name of the map file to **getEmployee_callable** in the Shared Library. (This is the map that you created earlier).



4. The flow is now complete, so save and close the flow.

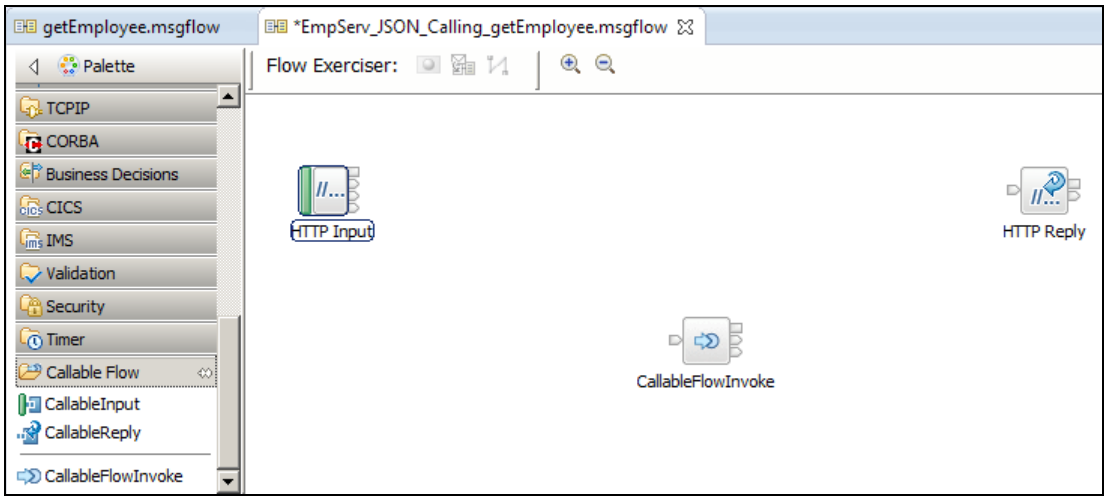


2.4 Create the Calling Message Flow

1. Expand the EmployeeService_callingApp application, and open the **EmpServ_JSON_Calling_getEmployee** message flow.

   The flow does not have any nodes at this point. Drop the following nodes onto the flow:

   - HTTP Input
   - CallableFlowInvoke
   - HTTP Reply

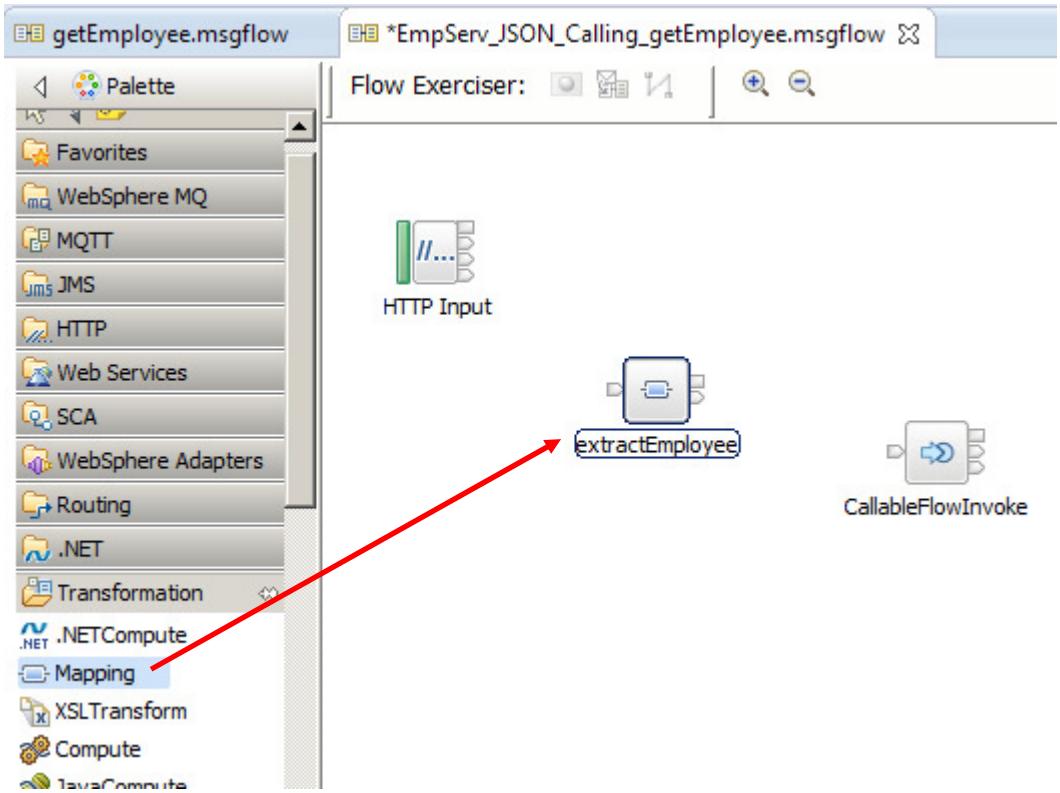


2. Set the following node properties:

   - HTTP Input
     - Basic, Path suffix URL: /empServClient_Calling_getEmployee
     - Input Message Parsing, Message Domain: JSON

   - CallableFlowInvoke
     - Target Application: EmployeeService_Callable
     - Target Endpoint Name: EmpServ_get

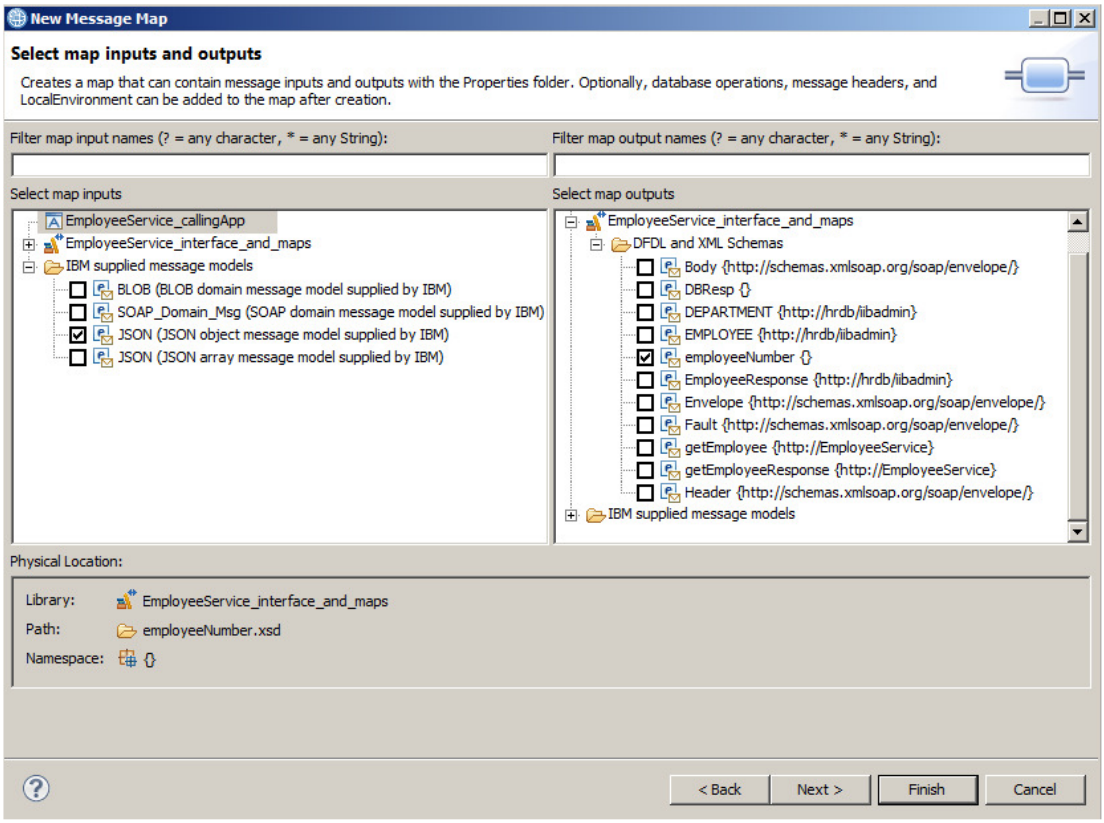
3. Drop a new mapping node onto the flow, and name as extractEmployee.



4. Double-click to open the map, and set the input and output as follows:

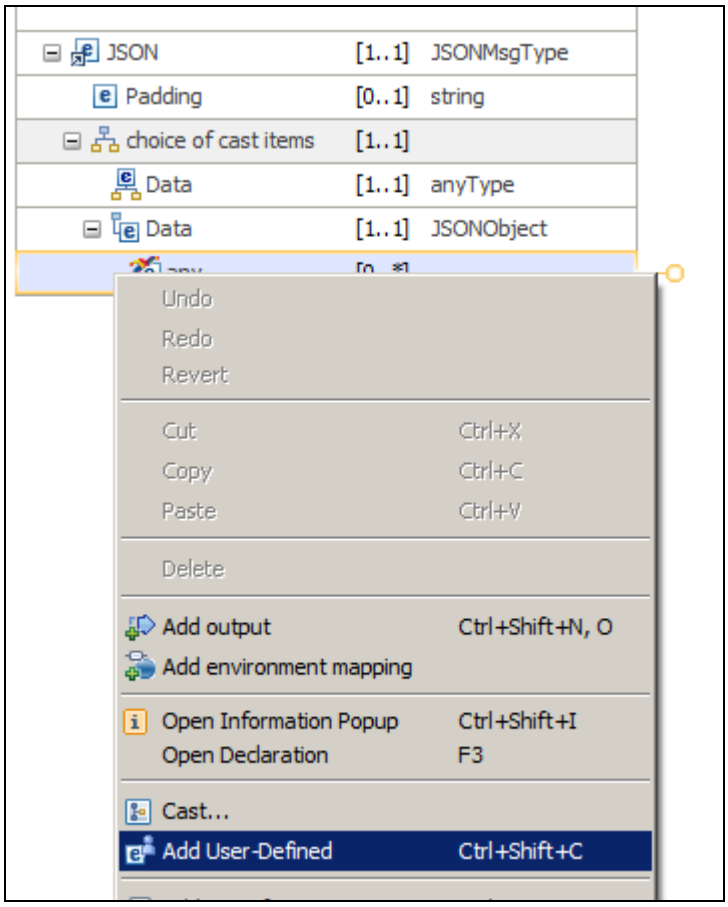
   - Input: IBM supplied message models, **JSON object**
   - Output: DFDL and XML schemas, **employeeNumber**

   Click Finish.



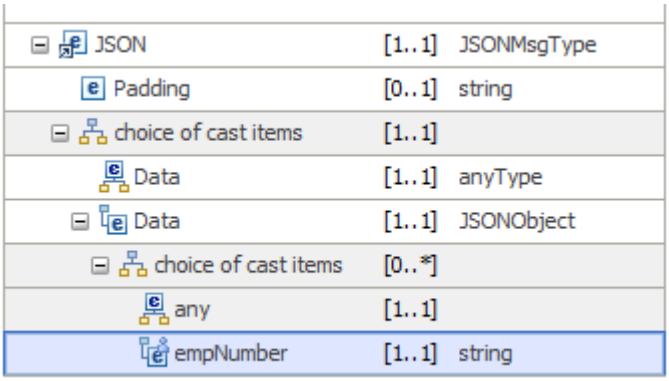
5. In the map editor, expand the JSON input message assembly.

   Right-click the "**any**" element, and select "Add User-Defined".

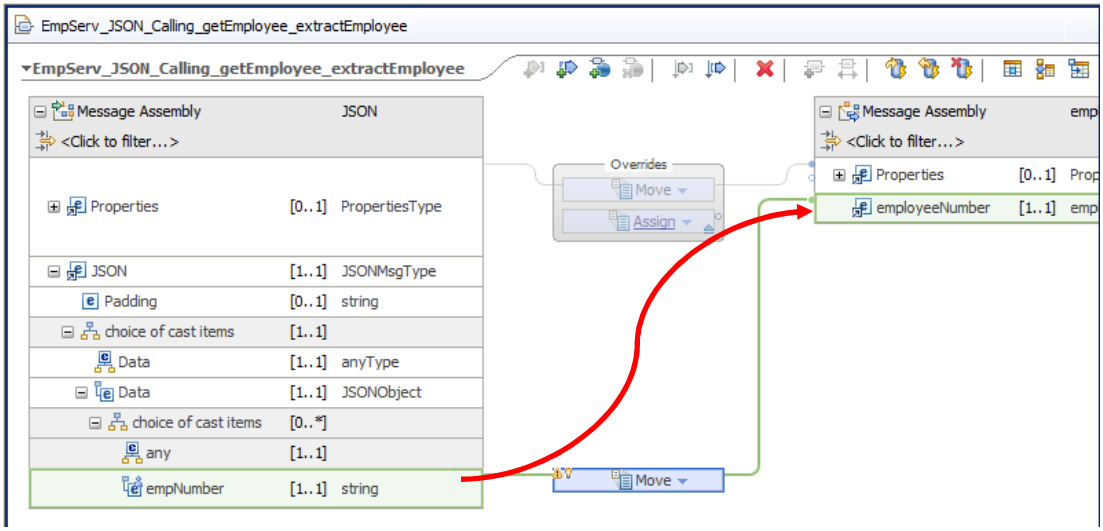


6. Set the name of the new element to **empNumber**. The type can be left as string.

   This enables the map to parse an incoming JSON message containing a single element named empNumber.

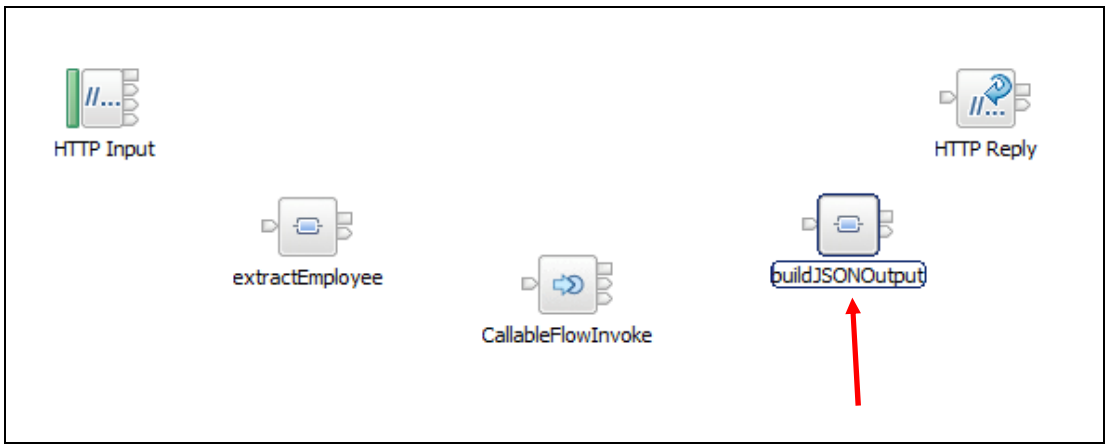


7. Connect the new element **empNumber** to the output **employeeNumber**. This will create a Move transform.



8. This is all that is required for this map, so save and close the map.

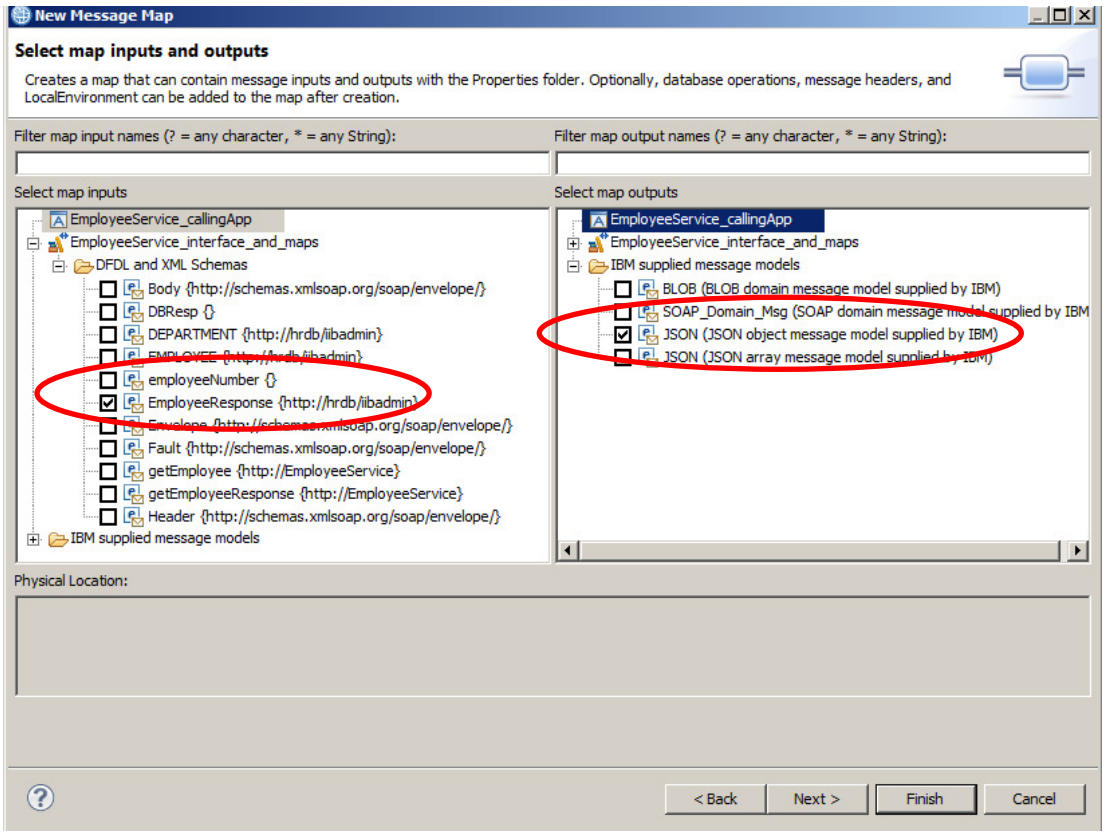
9. Drop a second map onto the flow, named **buildJSONOutput**.



10. Open the second map, and set the input and output as follows:

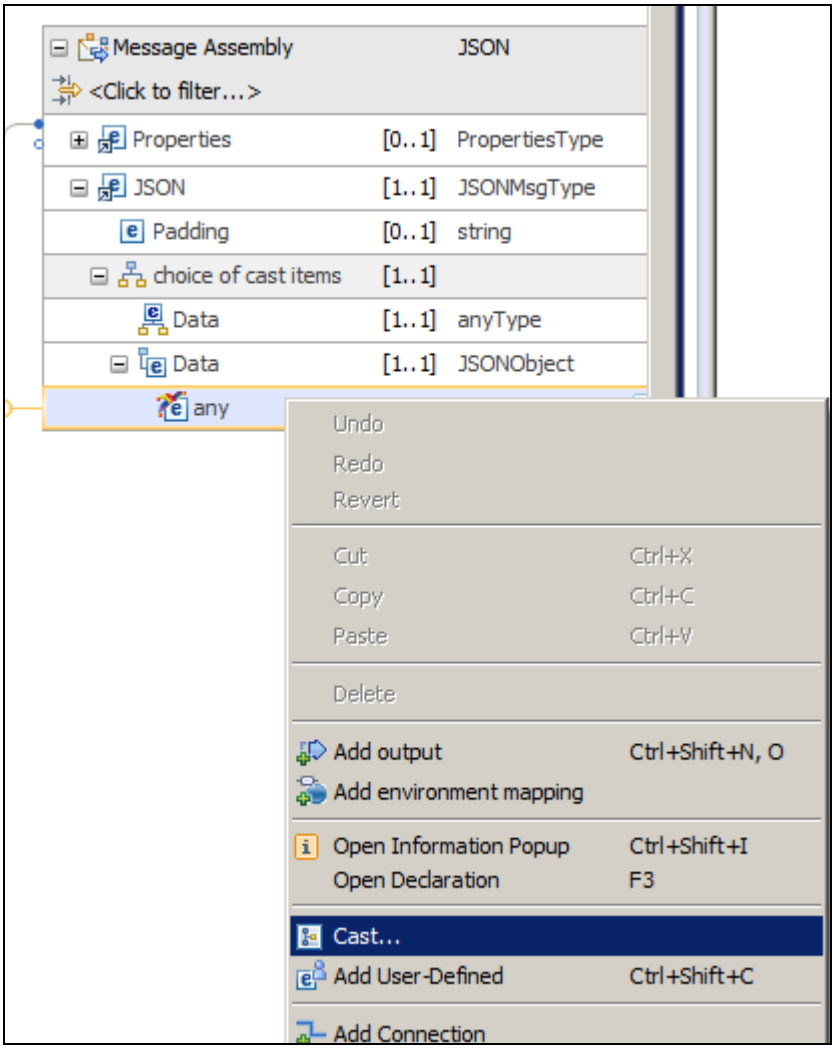
    - Input: DFDL and XML schemas, E**mployeeResponse**
    - Output: IBM supplied message models, **JSON object**

    Click Finish.

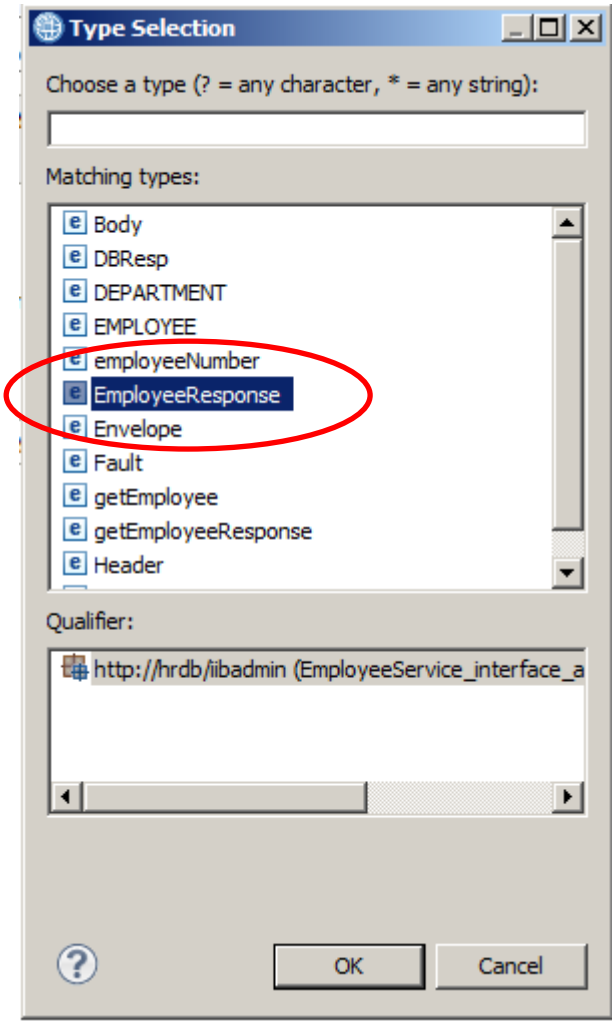


11. In the output message assembly, expand the JSON output, and right-click the "**any**" element.

    Select Cast.



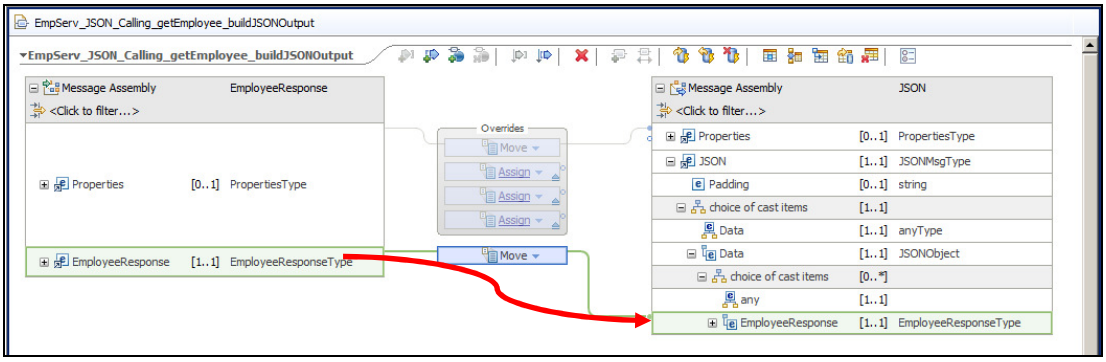
12. In the Type Selection window, select EmployeeResponse, and click OK.



13. In the map editor, connect **EmployeeResponse** to **EmployeeResponse**. This will create a Move transform.

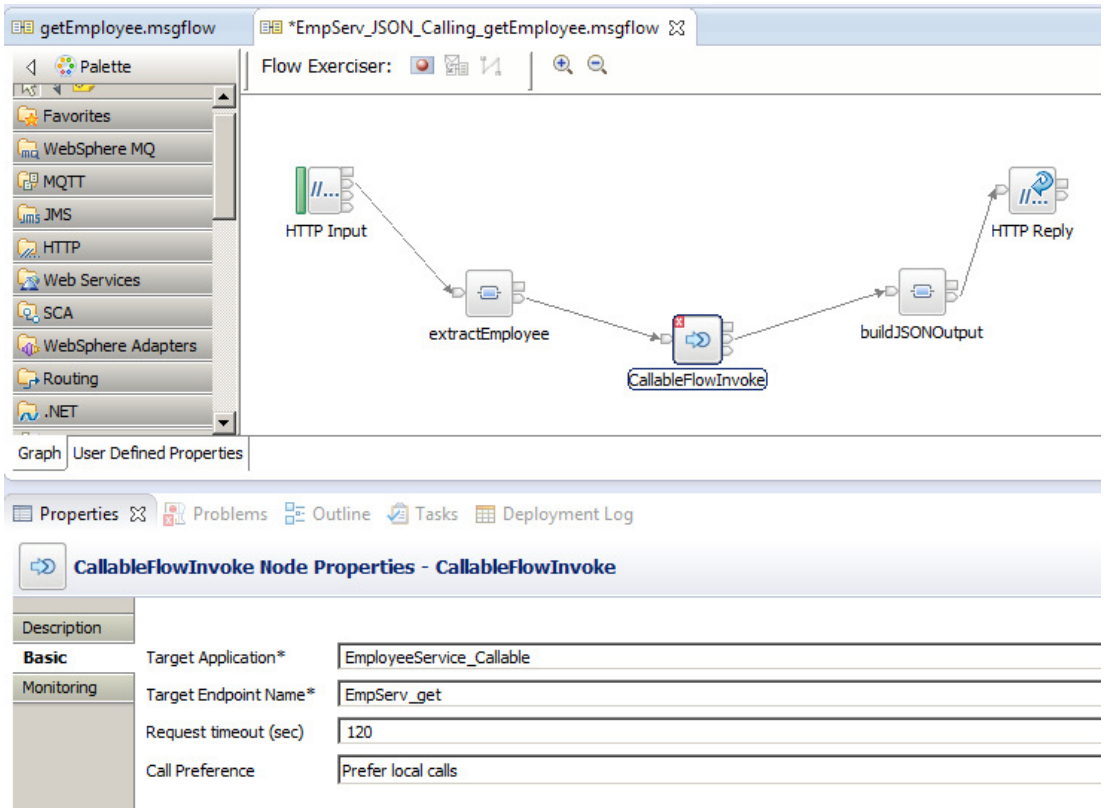
    This will transform the EmployeeResponse element into the same element, but wrapped in a JSON output message.

    Save and close the map.

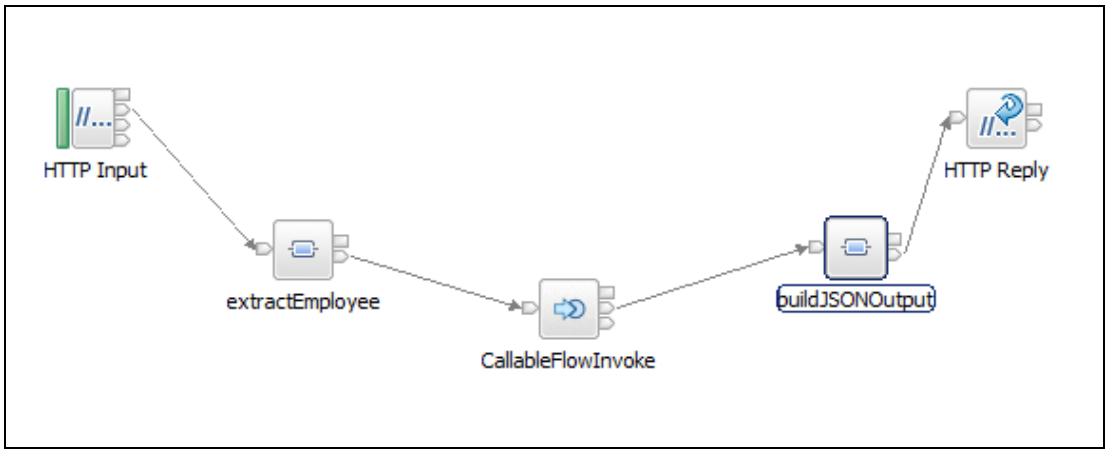


14. Highlight the CallableFlowInvoke node, and set the following properties:

    - Target Application: EmployeeService_Callable
    - Target Endpoint Name: EmpServ_get



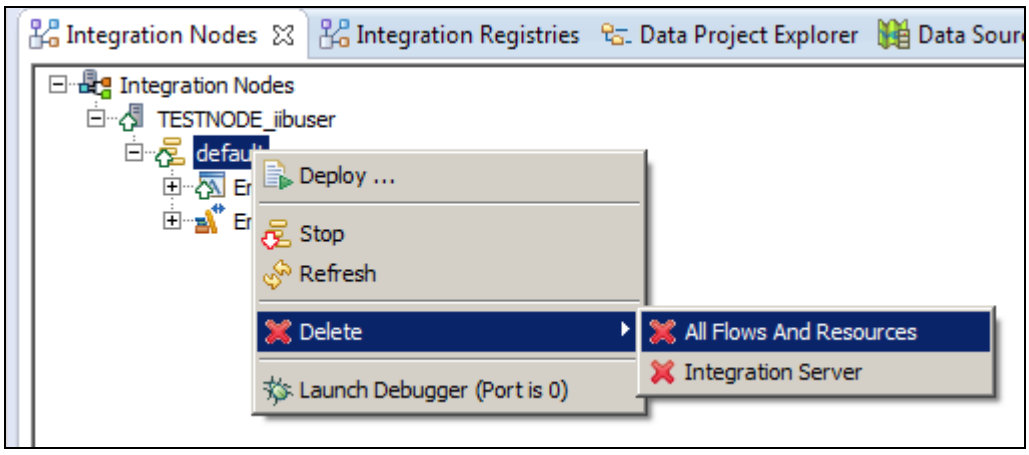
15. Finally, connect the nodes as shown, then save the message flow.



2.5 Deploy the shared library and callable application

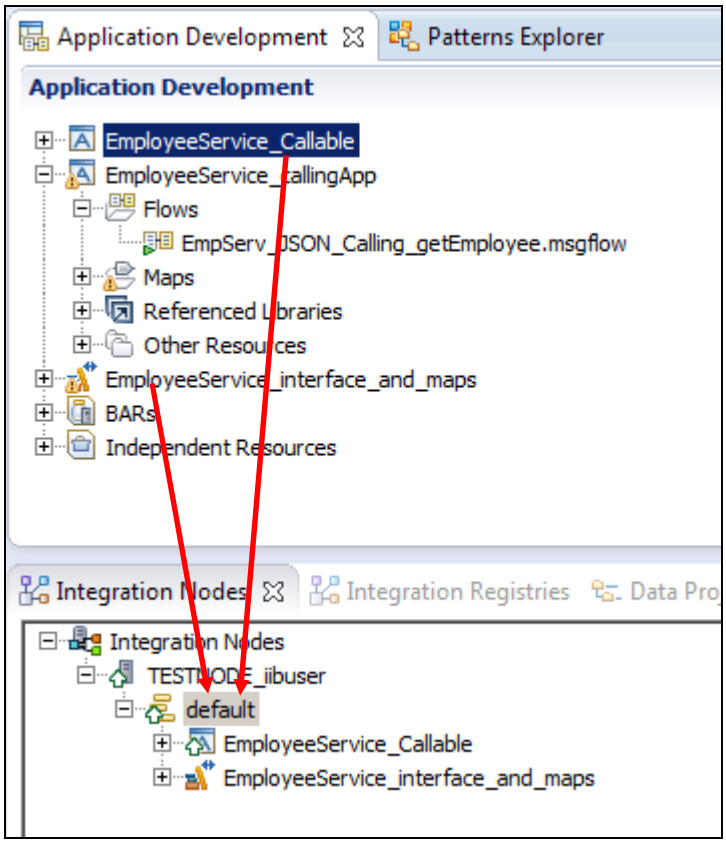
1. To make sure there are no conflicts with earlier scenarios, remove all currently deployed artefacts in the TESTNODE node.

   Right-click the default server, and select Delete, All Flows And Resources.



2. Deploy the shared library **EmployeeService_interface_and_maps**.

   Deploy the application **EmployeeService_Callable**.

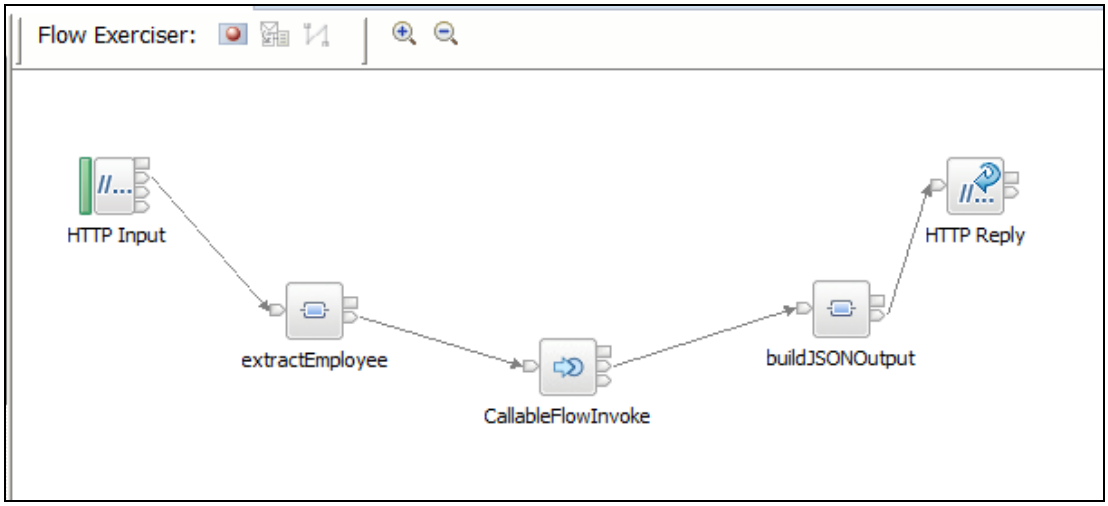


3. Test the Calling Application

3.1 Test with Flow Exerciser

1. Open the message flow EmpServ_JSON_Calling_getEmployee.

   Click the red button to invoke the Flow Exerciser.

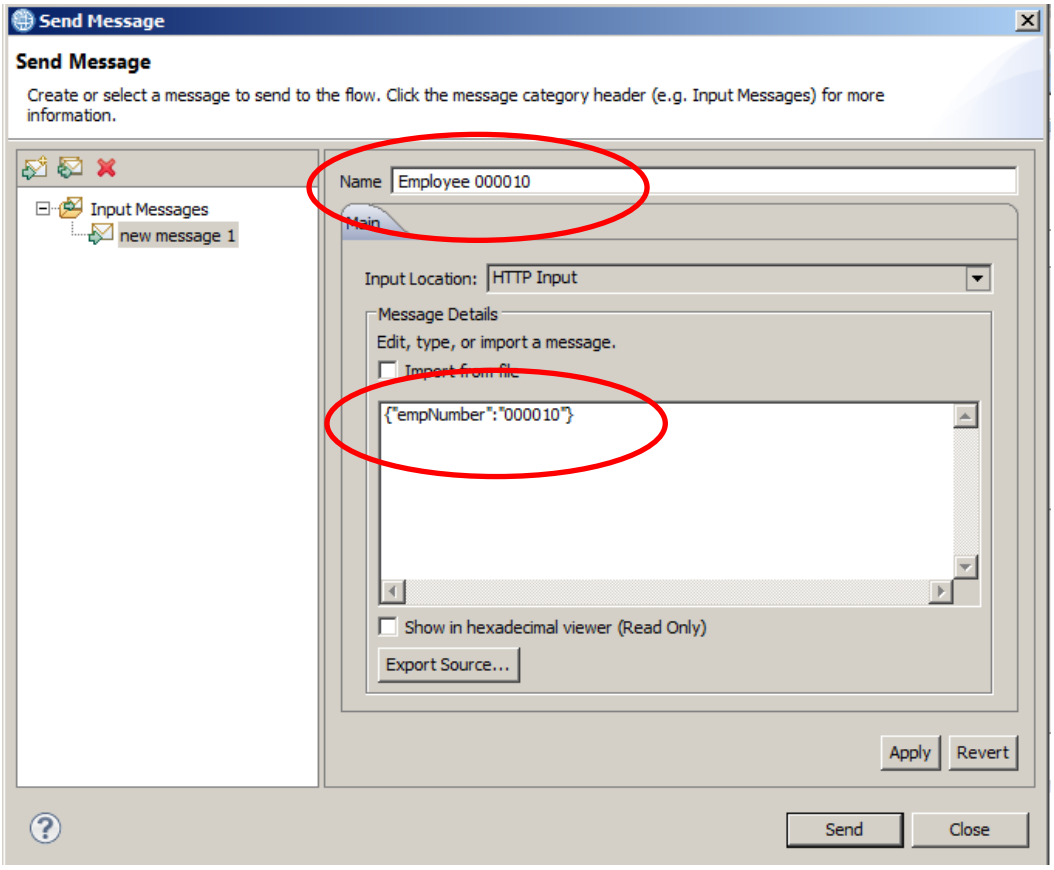


2. Name the new message "Employee 000010".

   Set the message payload to

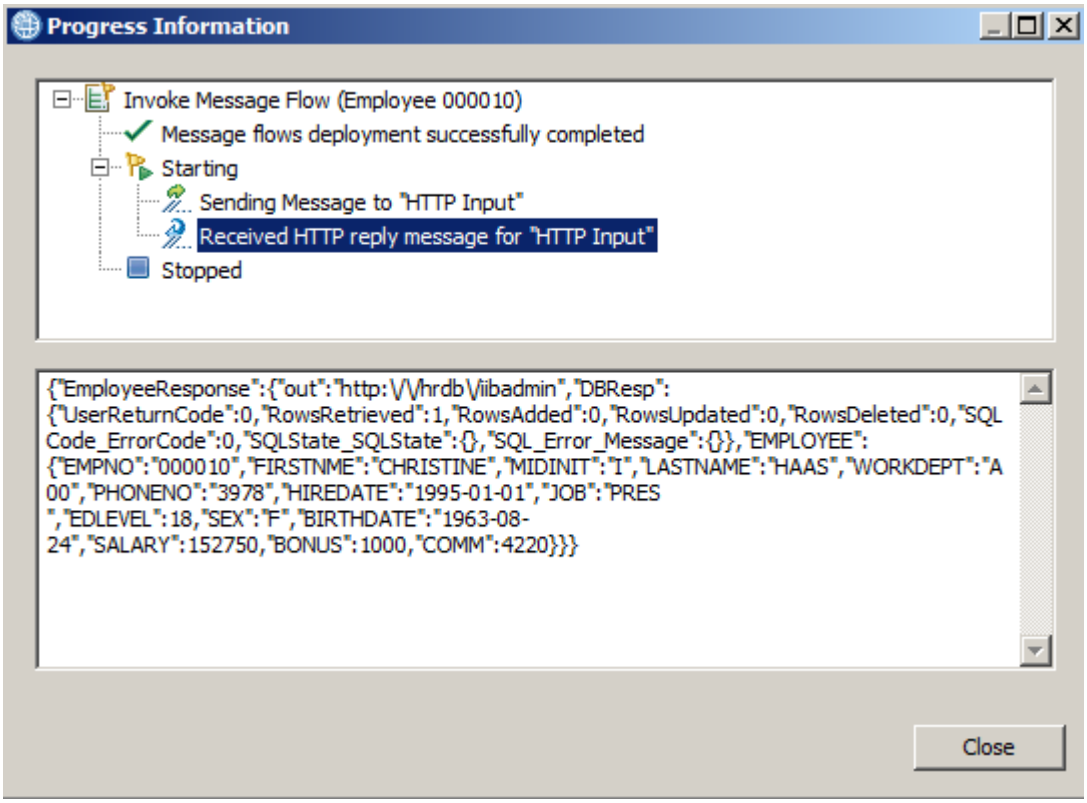
   ```
   {"empNumber":"000010"}
   ```

   Click Send.



3. When the flow has executed, highlight the "Received HTTP reply" message.

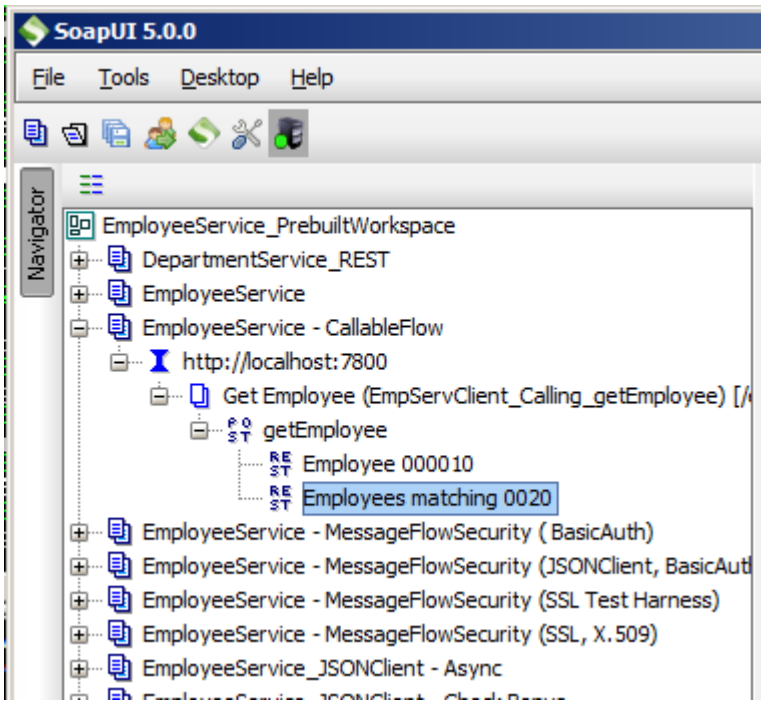
   The returned data should show the row returned from the HRDB database. Since the response will be in JSON format, this dialogue will not be able to format the data fully.



3.2 Test with SOAPUI

1. Open SOAPUI from the Start menu.

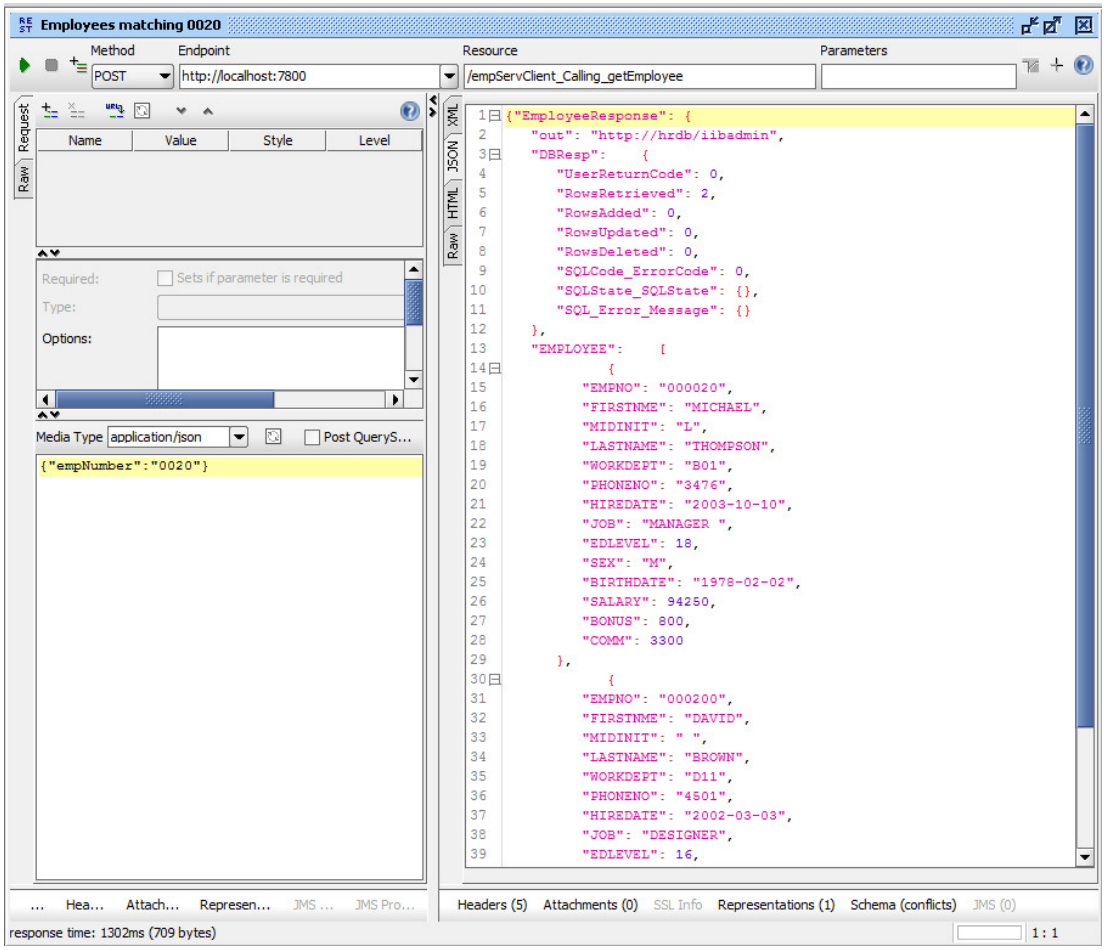
   Open the project EmployeeService - CallableFlow, fully expand and open the request "Employees matching 0020".



2. Click the green arrow to send the message.

   This message ("0020") tells the submap to retrieve all employees which match this number. Two employee records will be retrieved.

   Click the JSON tab to see the output in JSON format.



END OF LAB GUIDE