



OpenESB Enterprise Edition V3.x

Hello World

Document identifier:

Pymma document: 770-002

Location:

www.pymma.com

Editor:

Pymma Services: contact@pymma.com

Abstract:

This document provides a complete guide to create a simple “Hello world” application with OpenESB Enterprise EditionV3.x. You will learn how to install shared libraries, components and deploy composite applications, connect OE Studio and OE Instance, create BPEL and a composite application project. Then you will understand how to deploy and test your project.

Status:

This Document is in a beta state

ABOUT PYMMA CONSULTING

Pymma Services is a technical architect bureau founded in 1999 and headquartered in London, United Kingdom. It provides expertise in service-oriented integration systems design and implementation. As leader of the OpenESB project, Pymma is recognised as one of the main actors in the integration landscape. To reply to its customer's requirement, Pymma develops OpenESB Enterprise Edition which implements Enterprise features such as a Horizontal Scalability, Monitoring, IoT, Big Data Access and takes advantage of professional technical support.

Pymma is a European company based in London with regional offices in France, Belgium and Canada. (contact@pymma.com or visit our website at www.pymma.com)

Copyright

Copyright © 2015, Pymma Services LTD. All rights reserved. No part of this publication may be copied or distributed, transmitted, transcribed, stored in a retrieval system, or translated into any human or computer language, in any form or by any means, electronic, mechanical, magnetic, manual, optical, chemical or otherwise; or disclosed to third parties without the express written permission of Pymma Services LTD

Disclaimer

Information in this document is subject to change without notice and does not represent a commitment on the part of Pymma Services LTD. This manual is provided “as is” and Pymma is not responsible for disclaims all warranties of any kind with respect to third-party content, products, and services. Pymma will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Trademark Notice

Pymma is a trademarks of Pymma Engineering LTD. Java is trademarks of Oracle and its affiliates. Other names may be trademarks of their respective owners.

Contents

| | | |
|-------|--|----|
| 1 | Introduction | 6 |
| 1.1 | Before starting | 6 |
| 1.2 | What you learn in this document | 6 |
| 2 | Start OpenESB instance | 7 |
| 3 | Install OE shared libraries and component..... | 9 |
| 4 | Install the components..... | 12 |
| 5 | Connect OE-Studio to an OE-Instance..... | 16 |
| 6 | Create a BPEL project..... | 19 |
| 6.1 | Create a WSDL..... | 20 |
| 6.2 | Create a BPEL | 22 |
| 6.2.1 | Add BPEL activity | 24 |
| 6.2.2 | Link the activities with the PartnerLink(s)..... | 27 |
| 6.2.3 | Setup Assign activity..... | 30 |
| 7 | Create a Composite Application..... | 34 |
| 7.1 | Add the BPEL Module in a composite application | 35 |
| 7.2 | Add a Binding Component | 36 |
| 8 | Deployment | 37 |
| 8.1 | Deploy the composite application | 37 |
| 9 | Test the application. | 38 |
| 10 | Next steps | 41 |
| 11 | Help and support | 42 |

1 Introduction

When we start working with a product, it is customary to start with a ‘hello word exercise’. Through this simple exercise easily understandable by anyone, we develop a hello word application and get our first thrill and enjoyment by getting Hello + Name.

We go a bit further in this tutorial by explaining how to install OpenESB shared libraries and components and how to connect OpenESB Studio with an OpenESB instance. At the end of this tutorial, you deploy and test your application without leaving OE-Studio.

OpenESB Enterprise Edition (OE EE) offers the lightest and most efficient service-based integration tools on the market. Powerful, prompt, scalable with a very low memory footprint, OE EE is perfect for virtualisation and cloud deployment.

1.1 Before starting

We suppose you have already installed OpenESB on your machine. If not, please have a look at our document: **770-001-OE EE Installation Guide**.

1.2 What you learn in this document

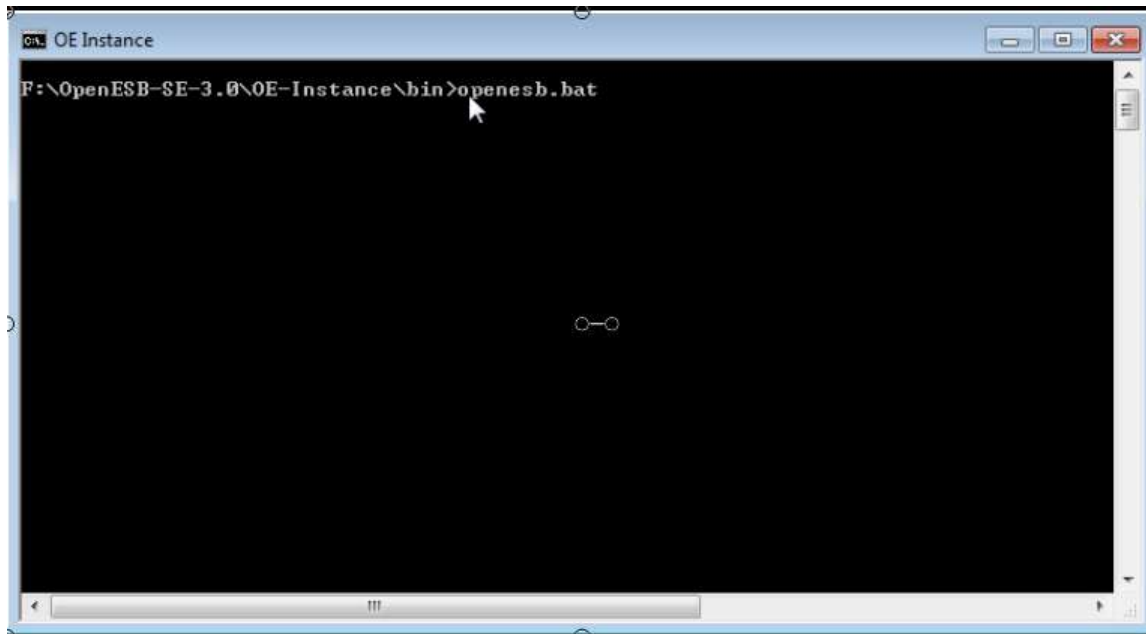
This tutorial agenda is:

- Install OE shared libraries and components
- Connect OE-Studio with an OE-Instance
- Create a BPEL project
- Create a composite application project
- Deploy a project
- Test a project

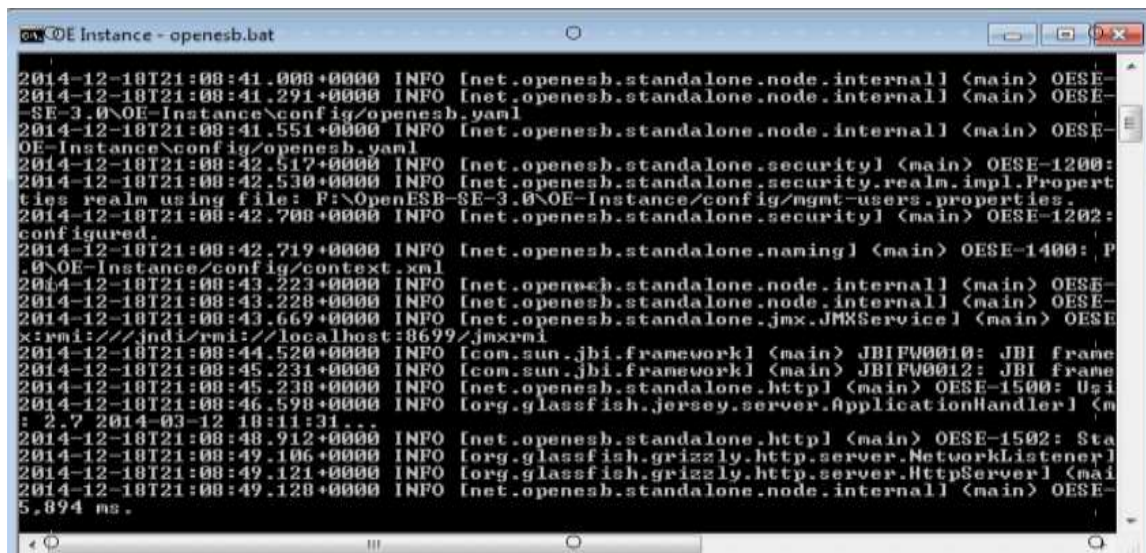
As much as possible we try to be clear and concise in our tutorial. Feel free to contact us to give us your feedback (contact@pymma.com).

2 Start OpenESB instance

First Open a console and go to the directory ...\\OpenESB-SE-3.0\\OE-Instance\\bin and run “openesb.bat” on Windows and “openesb.sh” on Linux or UNIX.



After a few seconds, OpenESB Instance is ready to run.



Open a browser (Firefox, Chrome or Else) and type the address: <http://localhost:4848/webui>.

OpenESB web admin console allows you to install shared libraries and components and deploy your project. It provides monitoring facilities as well.

Before starting, you have to enter a login and password. Default values are “admin” and “admin”.

Login



The login form consists of two input fields and a login button. The first field is for the username, containing the text 'admin'. The second field is for the password, represented by six dots. Below the fields is a green 'Login' button with a right-pointing arrow icon.

Then the web admin console opens.



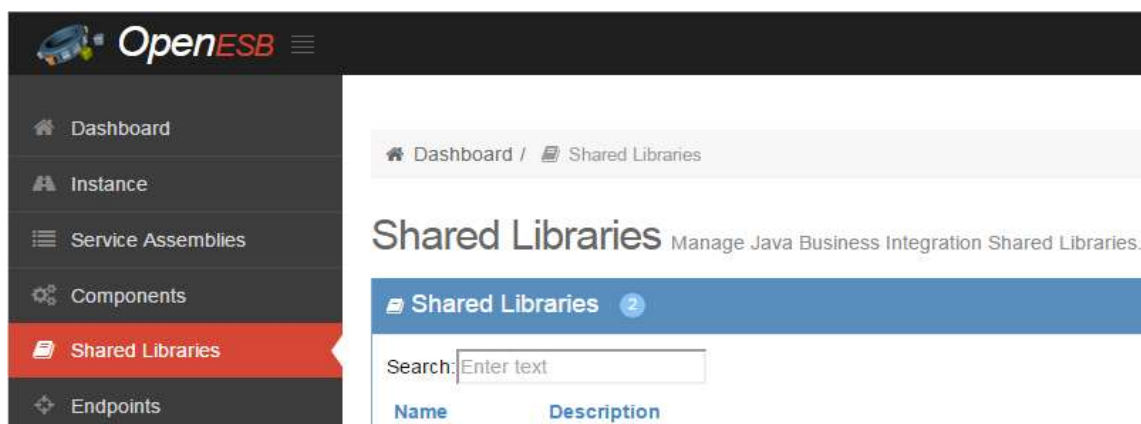
OpenESB is now ready to run.

3 Install OE shared libraries and component

OpenESB Enterprise installation comes without components; it is up to you to install the shared libraries and components to run your applications. This installation is much lighter and takes up less memory.

Shared libraries are jar files shared by components in OpenESB. For this tutorial, you must install two shared libraries: **wsdlexlib.jar** and **encoderlib.jar**. The purpose of each library is beyond the scope of this tutorial.

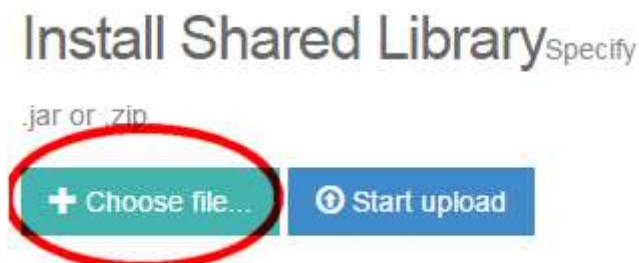
Select “**Shared Libraries**” in the main menu.



Click on the Install button on the left of the console.

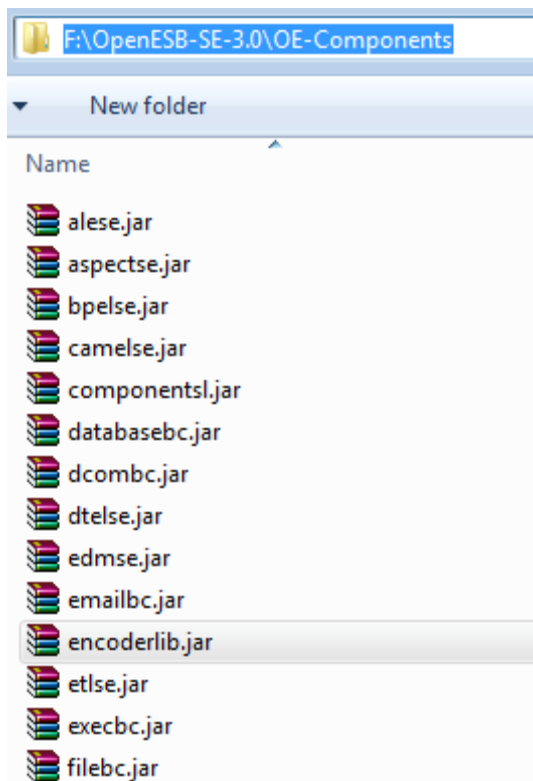


Click on the button “**Choose file**”

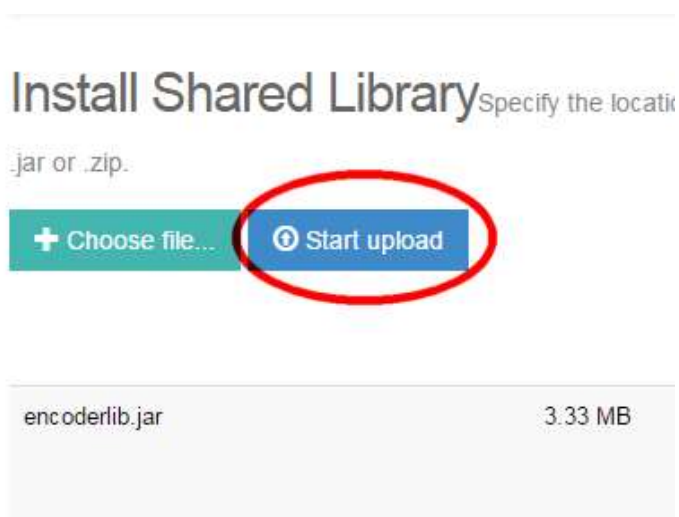


Where are the components located? Shared libraries and components are in .../OpenESB-SE-3.x\components.

Go to this directory and select the file “**encoderlib.jar**”.





Then select the button “**Start Upload**”.



After a few seconds, the library opens. Redo the same process with **wsdlxlib.jar**. You must get almost the same screen.

Shared Libraries Manage Java Business Integration Shared Libraries.

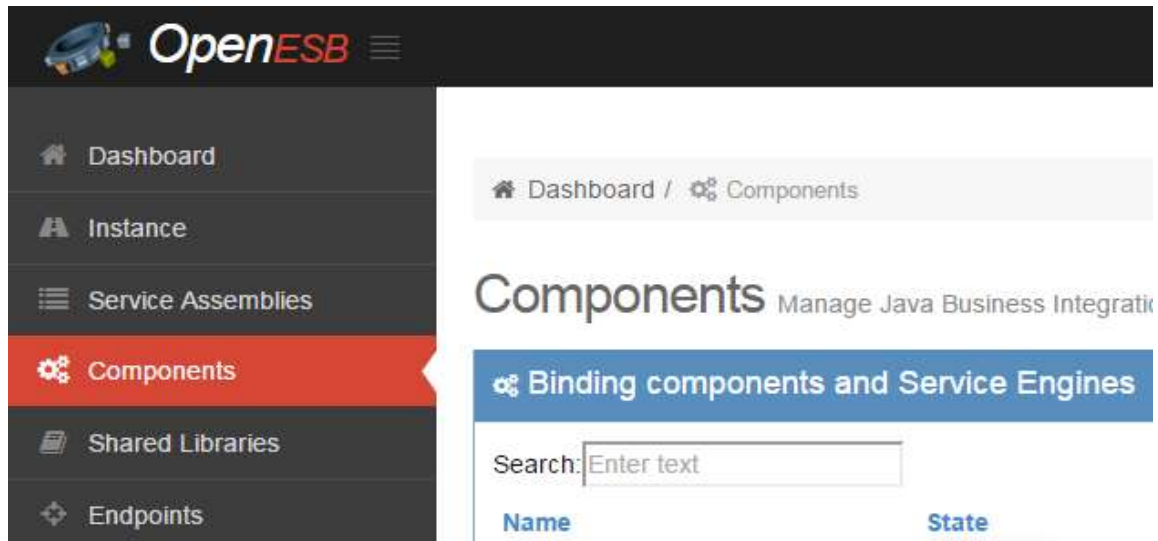
| Shared Libraries 2 | | | | + Install |
|---|---|----------------|---|---------------------------|
| Search: <input type="text" value="Enter text"/> | | | | |
| Name | Description | Version | | |
| sun-encoder-library | JBI shared library for encoders. Encoders supported: customencoder-1.0 hl7encoder-1.0 cocoencoder-1.0 | 2.3.2-SNAPSHOT |  | |
| sun-wsdl-ext-library | A collection of wsdl utilities for JBI components. | 2.3.2-SNAPSHOT |  | |

The shared libraries are installed then, we can install the components.

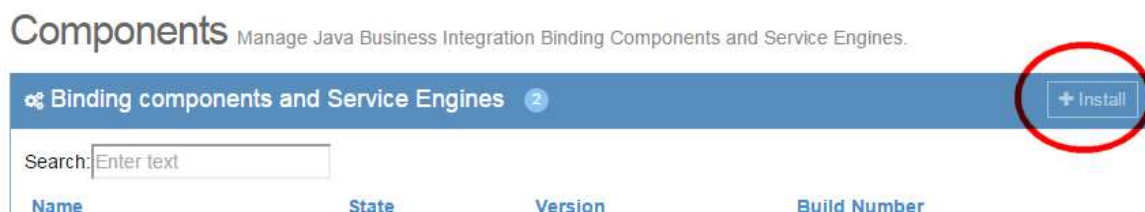
4 Install the components

Shared libraries and components installations are similar. Nevertheless, component installation requires an additional step. By default, a component is installed in the state shutdown and to run it must be set up in the state start after the installation. For more information on component life cycles, please read JBI specifications. In this tutorial, we install the HTTP and BPEL components:

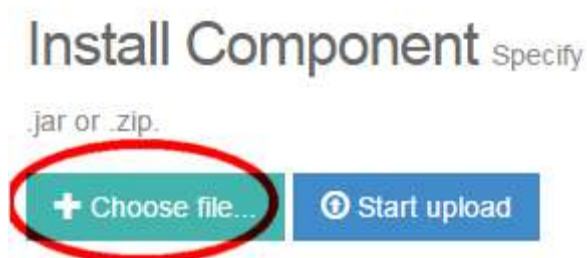
On the main menu, click on Components.



Click on the Install button on the left of the console.

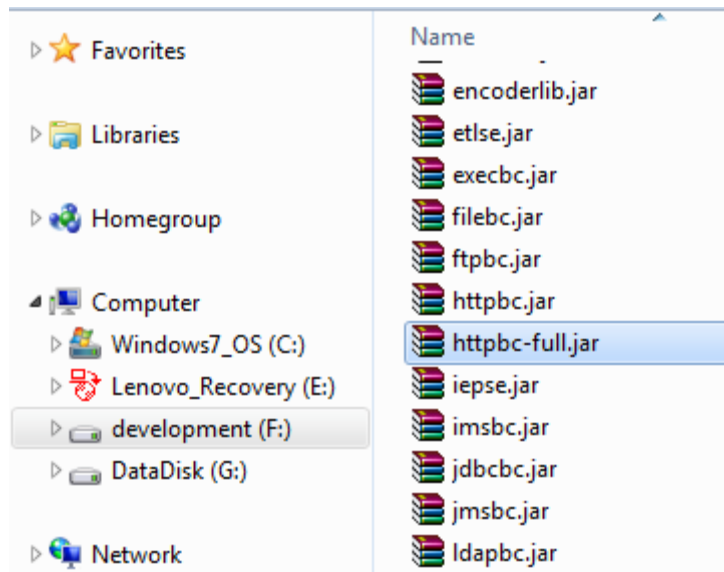


Click on the button “Choose file”



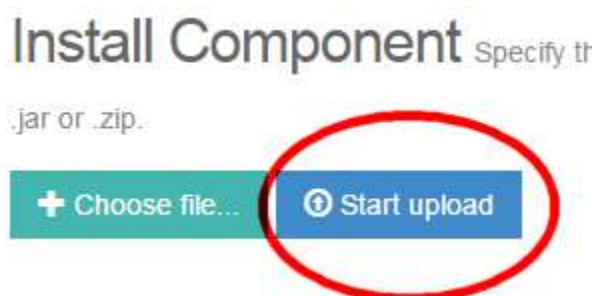
Select the component HTTP-Full.jar.

Doc: 770-002: OpenESB Standalone Enterprise Edition Hello World

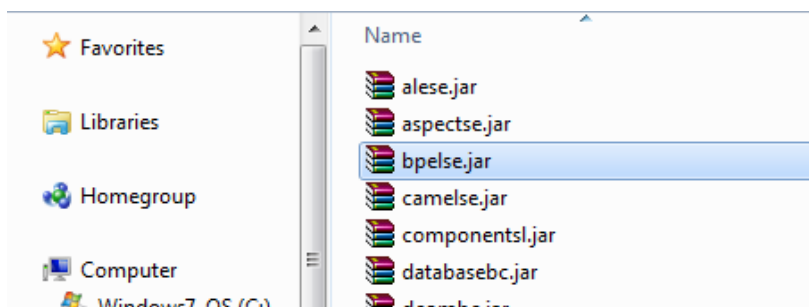


If httpbc.jar is present in your distribution, you should not use it with the OpenESB Enterprise Edition. Httpbc.jar was used with OpenESB version 2.x.

Install the component.



Use the same process to install the component bpelse.jar.



When a component is uploaded, its state is set up to **“Shutdown”**. A component must be in the start state to process messages.

Dashboard / Components

Components Manage Java Business Integration Binding Components and Service Engines

Binding components and Service Engines 2

Search:

| Name | State |
|--|----------|
|  sun-bpel-engine | SHUTDOWN |
|  sun-http-binding | SHUTDOWN |

Click on the component name, select the general tab and select “Start”

Dashboard / Components / sun-bpel-engine

sun-bpel-engine SHUTDOWN

General Configuration Application Configurations Application Variables Descriptor Loggers

Shared Libraries Monitoring

Binding Component General Properties

View the details for this Java Business Integration Binding Component

Description

Name: sun-bpel-engine
Type: SERVICE_ENGINE
Description: WS-BPEL 2.0 compliant JBI Service Engine
Version: 2.3.2-SNAPSHOT
Build Number: 803

Operation

State: SHUTDOWN

Actions: **Start** Stop Shutdown



After a few seconds, the BPEL component starts. Start the HTTP component to continue the tutorial.

Dashboard / Components

Components Manage Java Business Integration Binding Components and Service Engines

Binding components and Service Engines 2 + Install

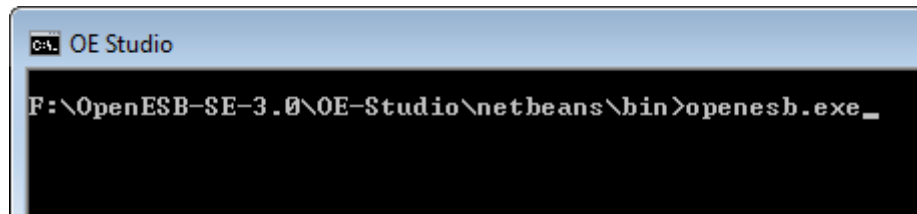
Search:

| Name | State | Version | Build Number |
|--|---------|----------------|--------------|
|  sun-bpel-engine | STARTED | 2.3.2-SNAPSHOT | 803 |
|  sun-http-binding | STARTED | 2.3.2-SNAPSHOT | 803 |

Shared libraries and components are installed. Now, you can develop and deploy your application.

5 Connect OE-Studio to an OE-Instance

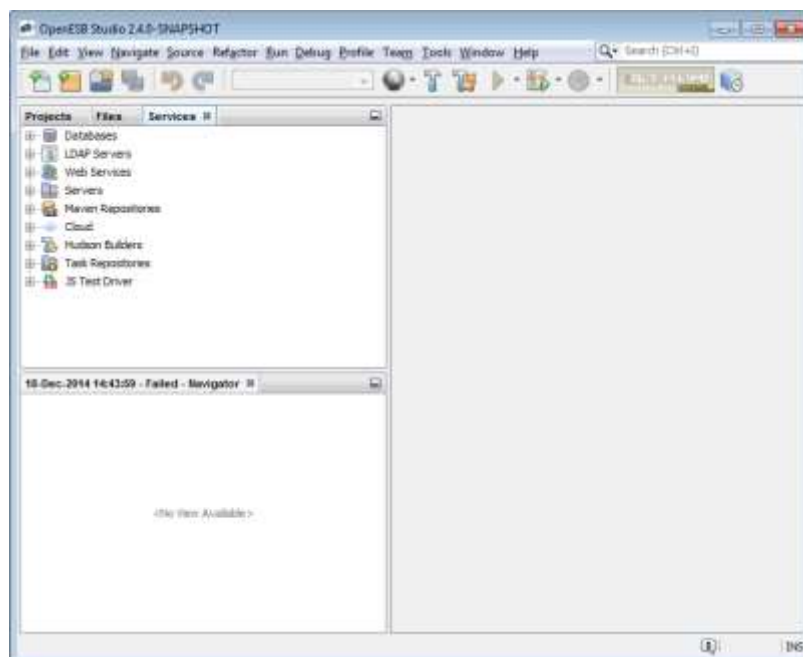
In this chapter, we connect OpenESB studio and the OpenESB instance we already started in the previous chapter. Open a new console and go to the repository .../OpenESB-SE-3.0/OE-Studio/NetBeans/bin. Run `openesb.exe` for Windows or `openesb` for Linux.



The OpenESB studio logo appears.



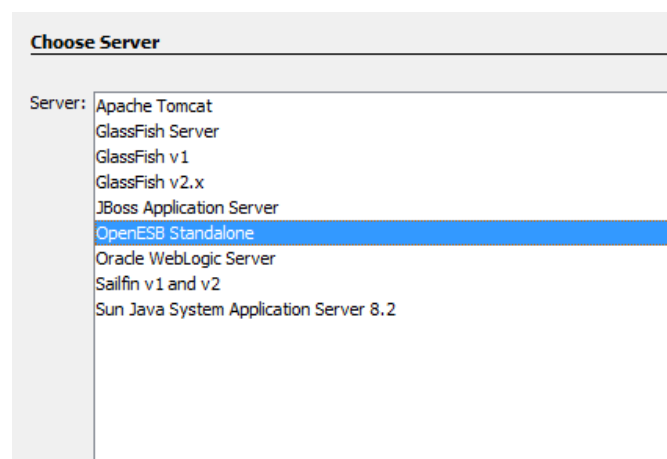
After a few moments, the OpenESB studio opens.



Select the node Server, right click and select Add Server.



Select the server OpenESB Enterprise.



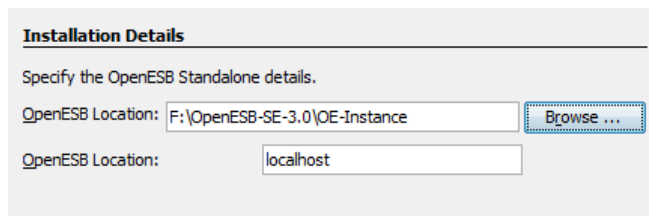
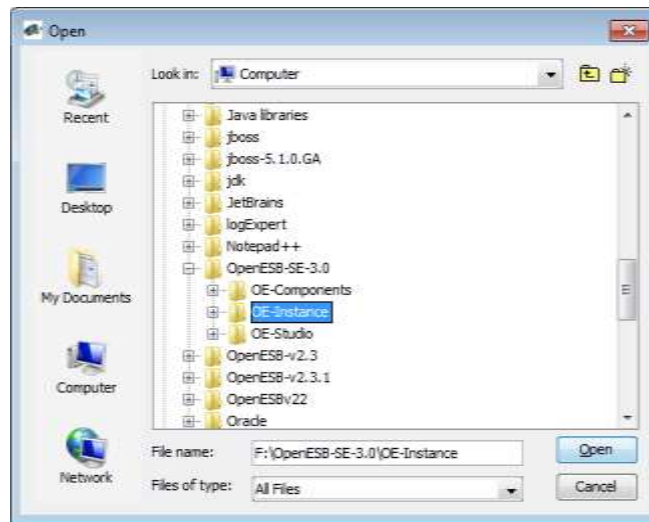
Then click on the NEXT button.



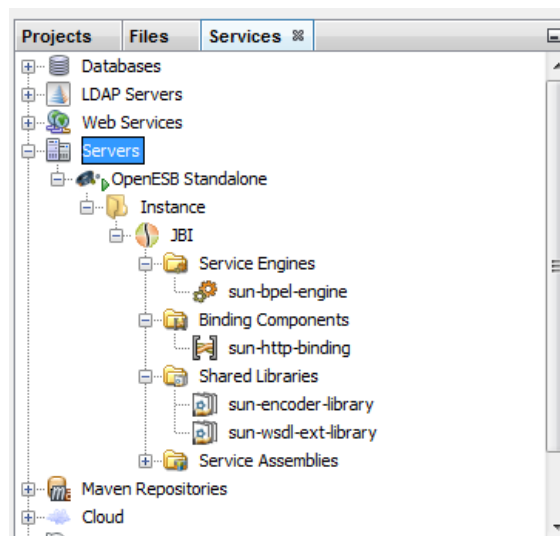
In the OpenESB Studio V3.x, the Installation screen has two fields both entitled, “OpenESB Location”. The first one is used for local installations. The second one is dedicated to remote installations planned in the next OE-Studio versions.

OpenESB location is **.../OpenESB-SE-3.0/OE-Instance**. Leave the second field as it is. It should be **localhost**.

Doc: 770-002: OpenESB Standalone Enterprise Edition Hello World



Click on the Finish button.

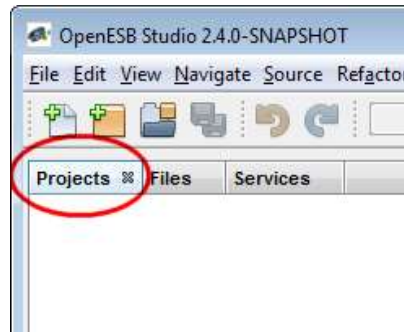


Expand the node “Servers”. You could notice that the OpenESB Studio is connected to our OE Instance.

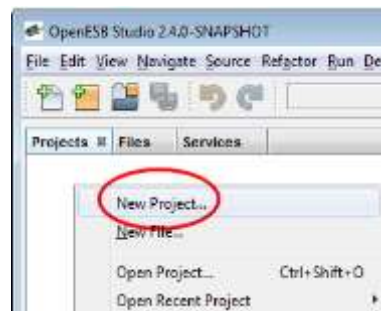
6 Create a BPEL project

In this paragraph, we create the “Hello World” Services. To do it we have to open the BPEL editor.

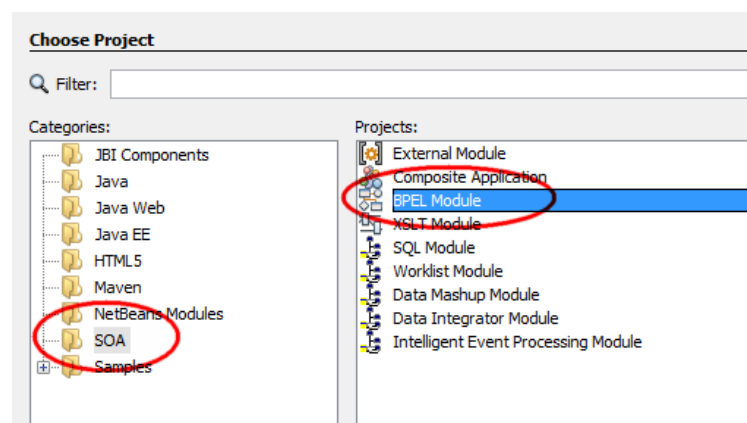
Select the tab project.



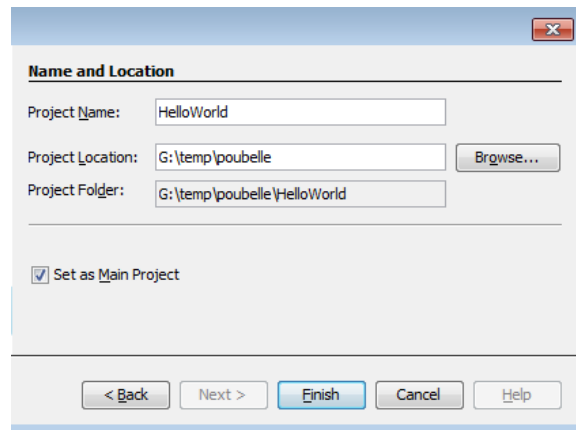
Right-click on the project windows and select New Project.



Chose the category SOA and then choose the BPEL Module.

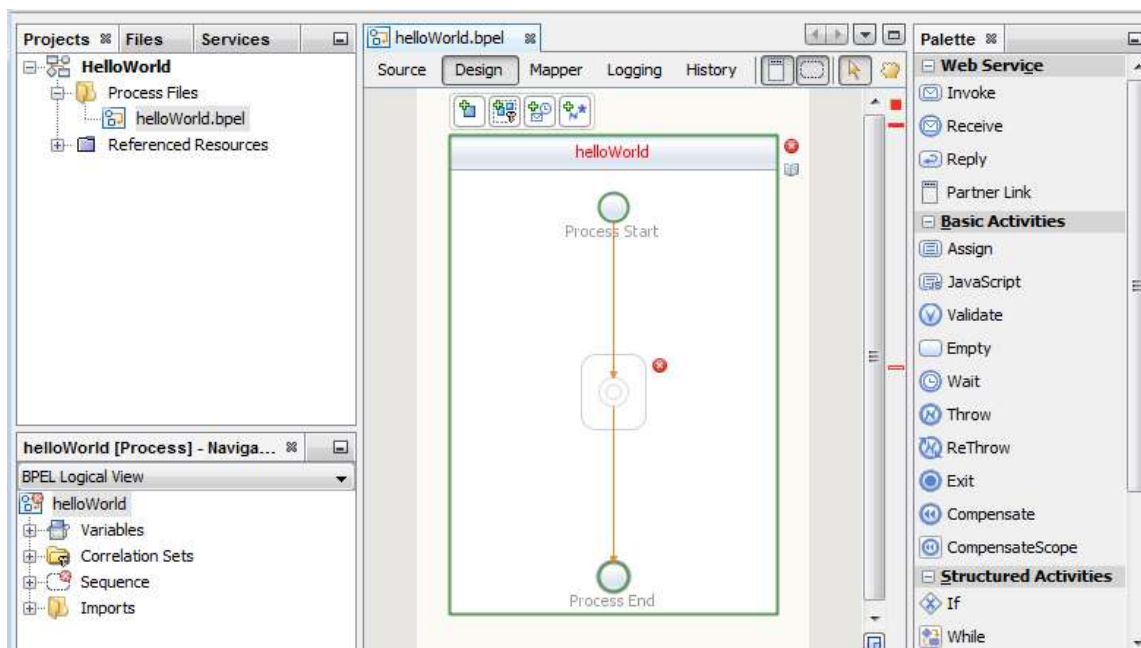


Click on Next.



Enter the Project Name and then click on Finish.

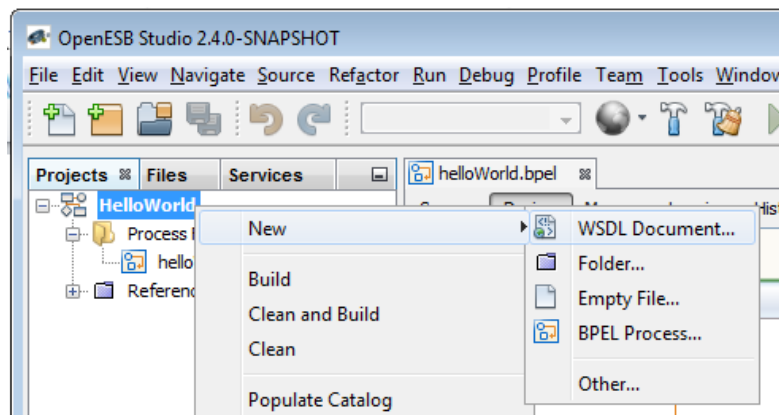
After a while, the OE-Studio screen appears as follows.



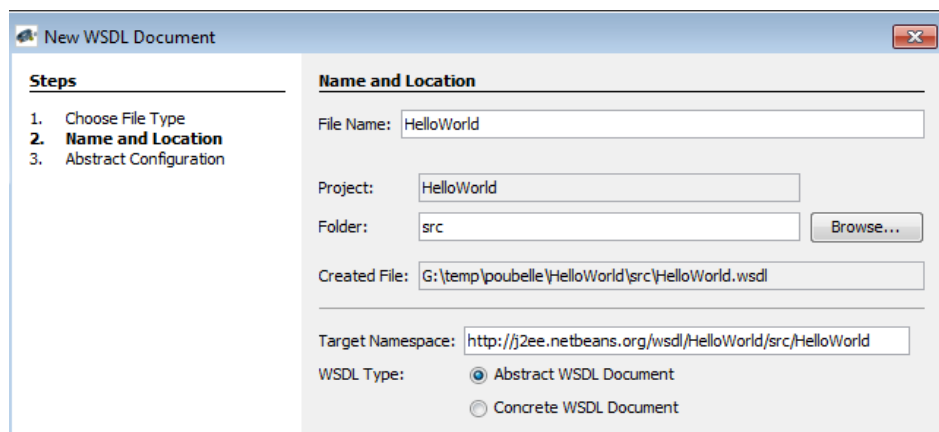
As part of the OpenESB process, before working on the orchestration, we have to define the contract of our hello world service (a WSDL). The contract defines the operation(s) and the parameter of our services. For Hello world service, the input is a String (ex: Pymma) and the output a String as well (ex: Hello Pymma).

6.1 Create a WSDL

Right-click on the project node HelloWorld and select New → WSDL.

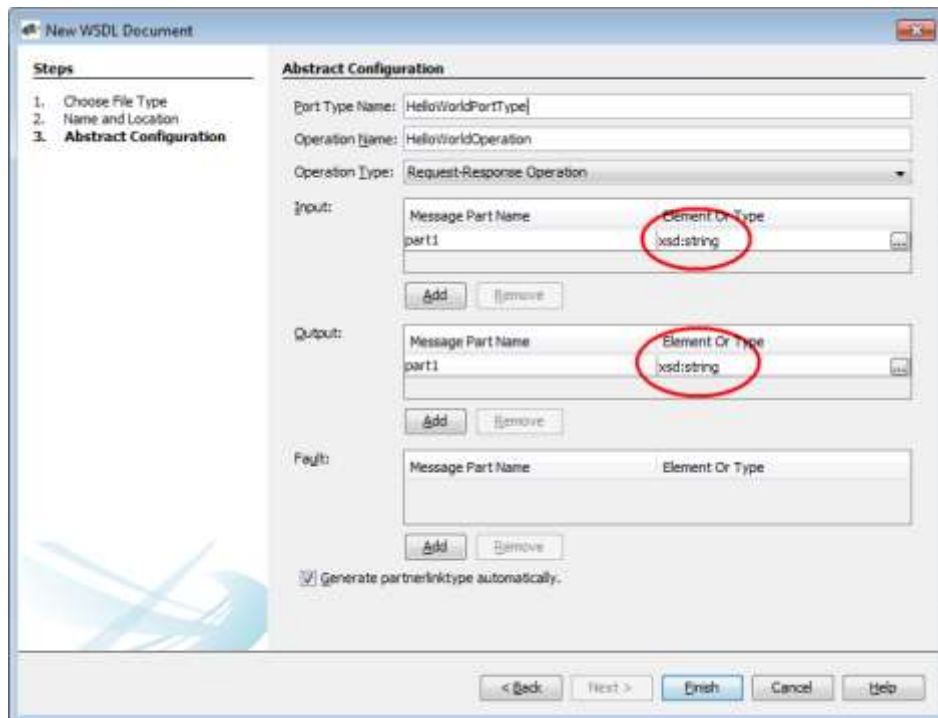


The WSDL wizard opens.

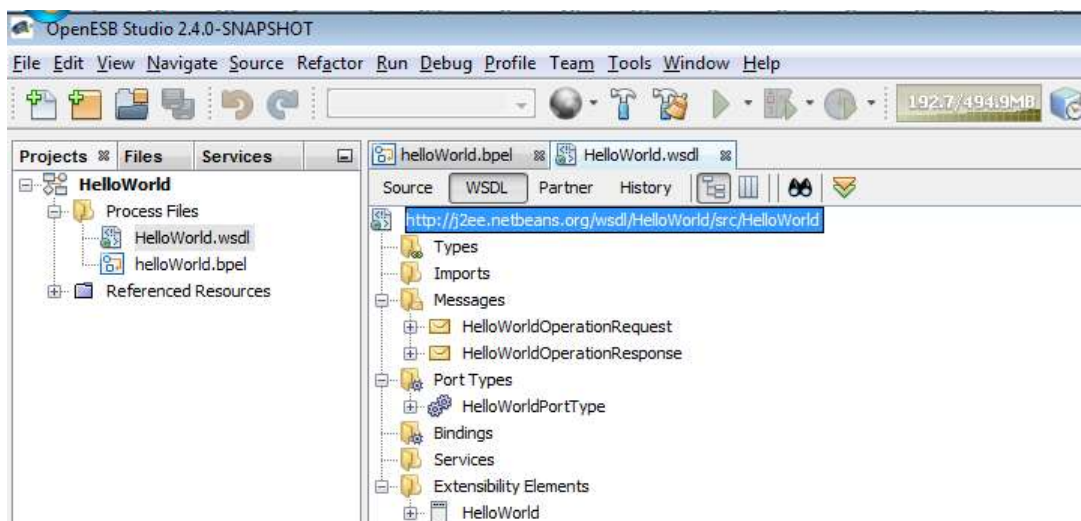


Type HelloWorld as File Name and let the WSDL Type as Abstract. Then click on **NEXT**.

Note that Input and output element are Strings by default. Keep the default field values and click on the **Finish button**.



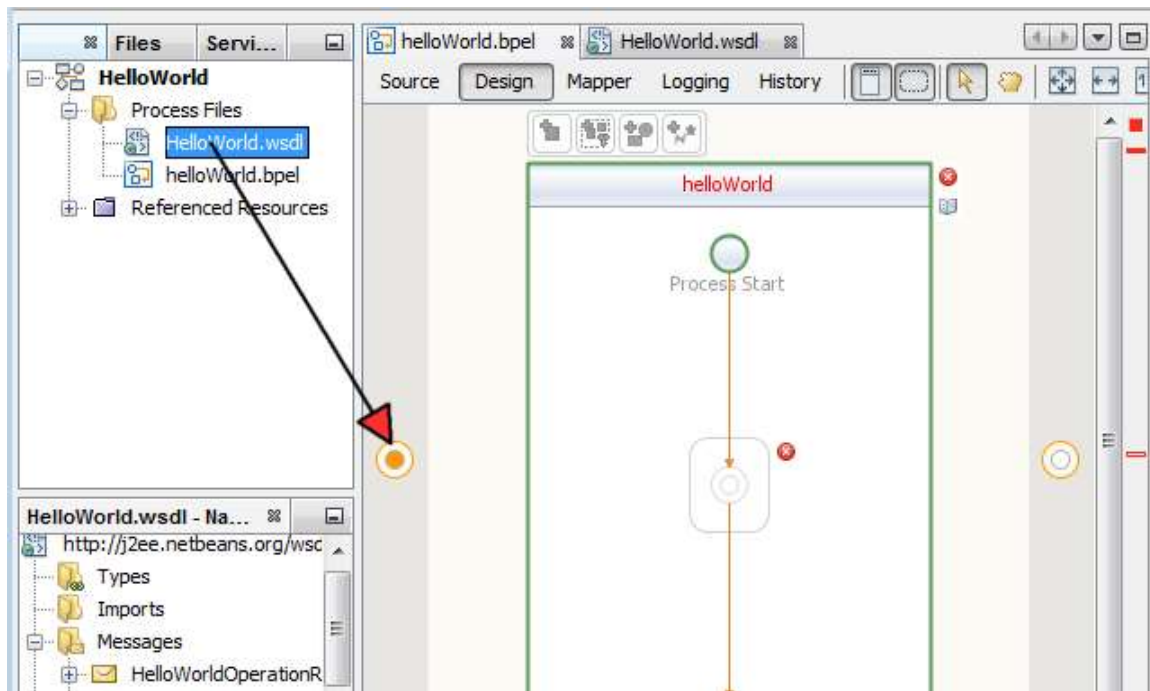
The HelloWorld WSDL file is added to your project.



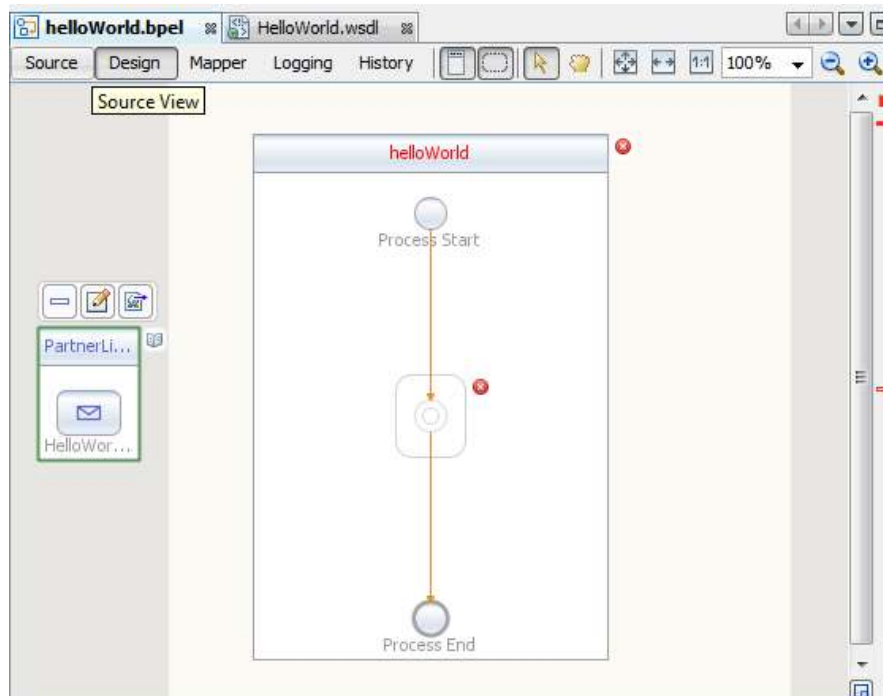
6.2 Create a BPEL

Click back on the tab HelloWorld.bpel. Select HelloWorld.wsdl and drag it onto the left grey strip of the BPEL editor.

When you position the mouse cursor in the left grey strip an orange circle appears. Drop the WSDL there.

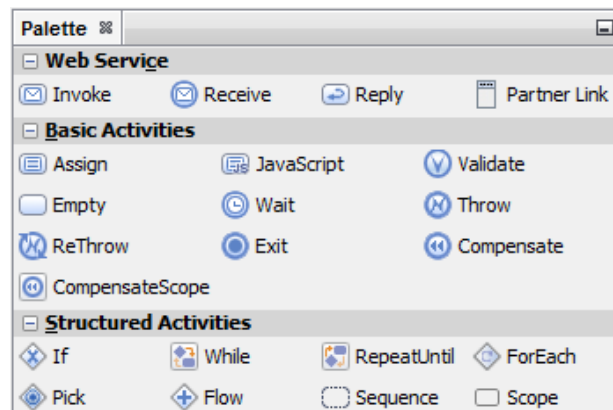


After dropping the WSDL on the orange circle, a PartnerLink appears in the BPEL Editor.



6.2.1 Add BPEL activity

On the right side of OE-Studio, you can see the BPEL palette.

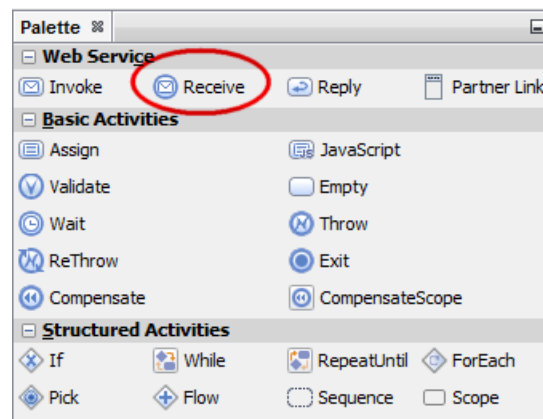


For this document, we use 3 BPEL activities:

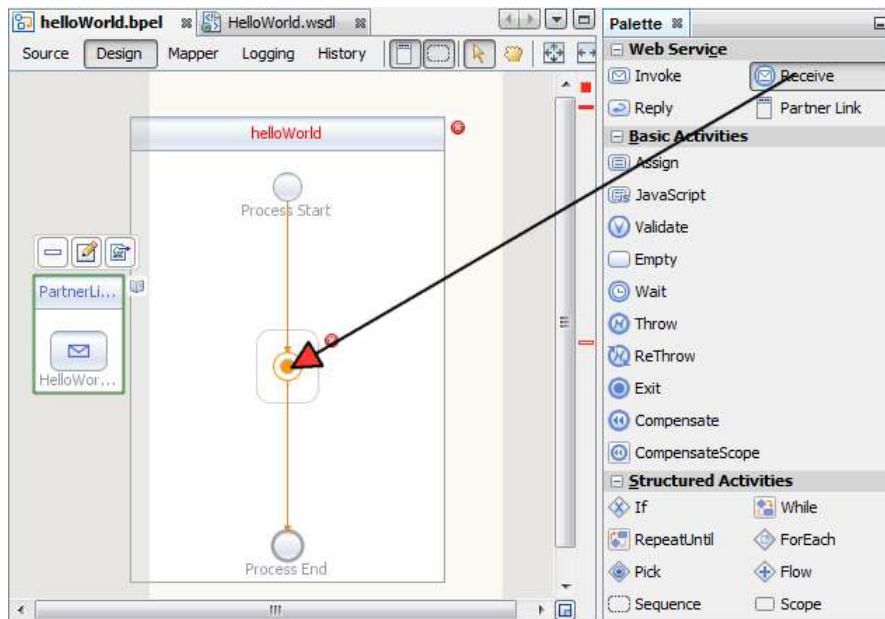
- Receive: used to receive the message
- Assign: map the input variable to the output variable
- Reply: return the message.

Then drag and drop the activities to design the process.

Select the Receive activity in the BPEL palette.

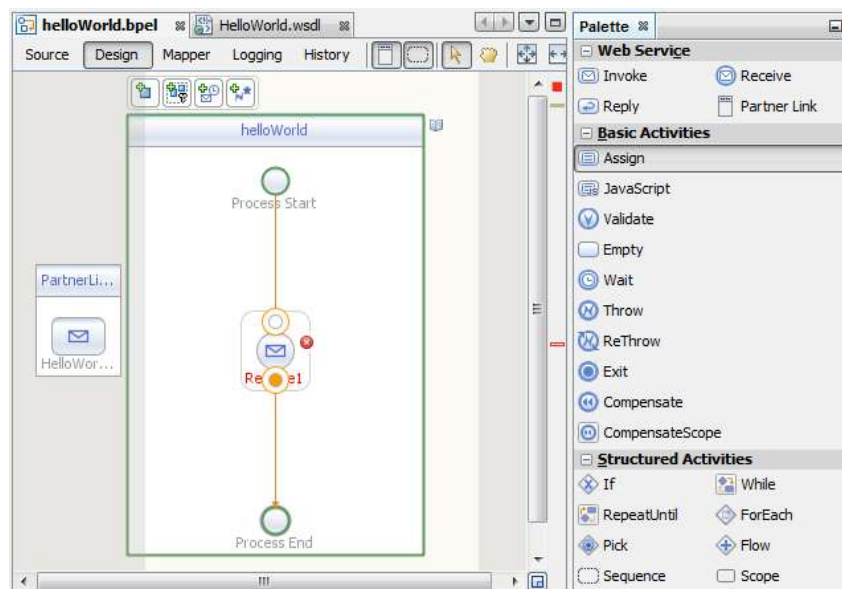


Drag the activity to the centre of the BPEL Editor.

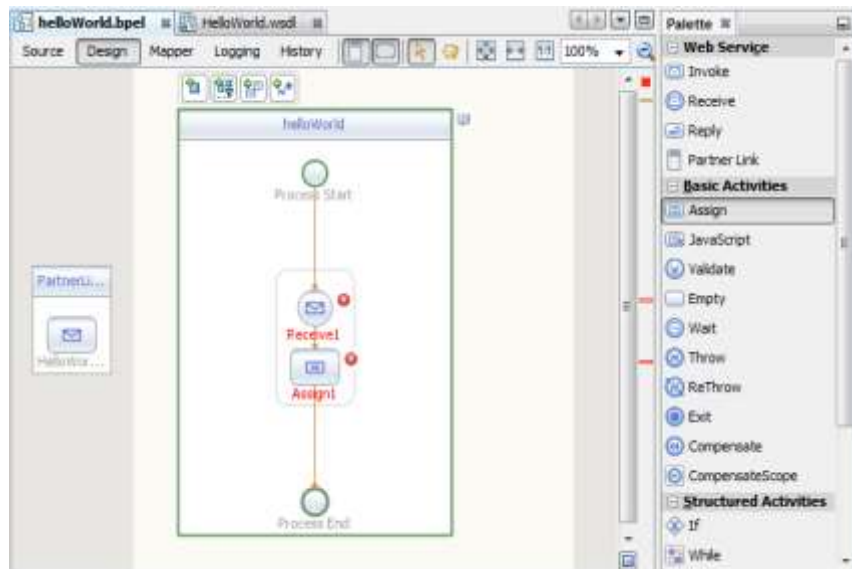
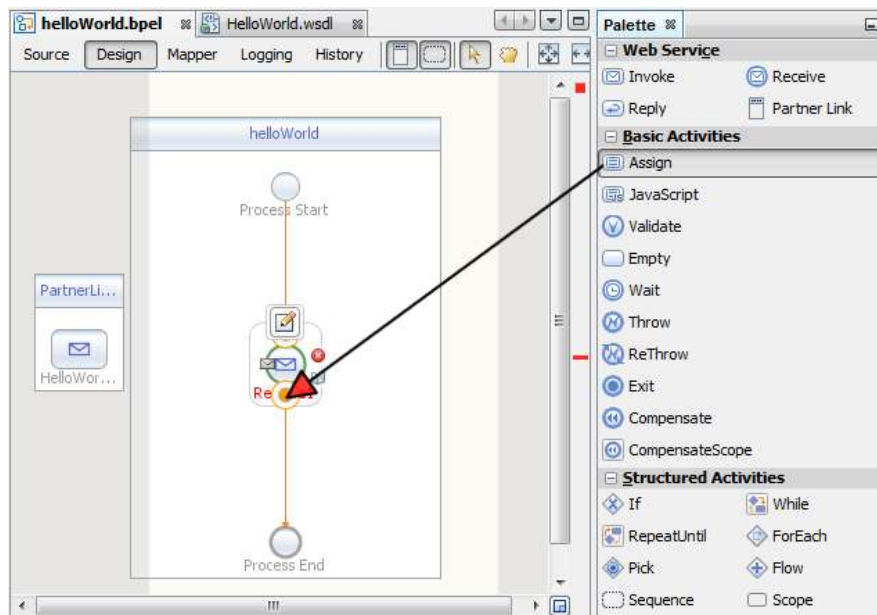


Drop the activity on the Orange circle, and the Received activity appears in the BPEL Editor.

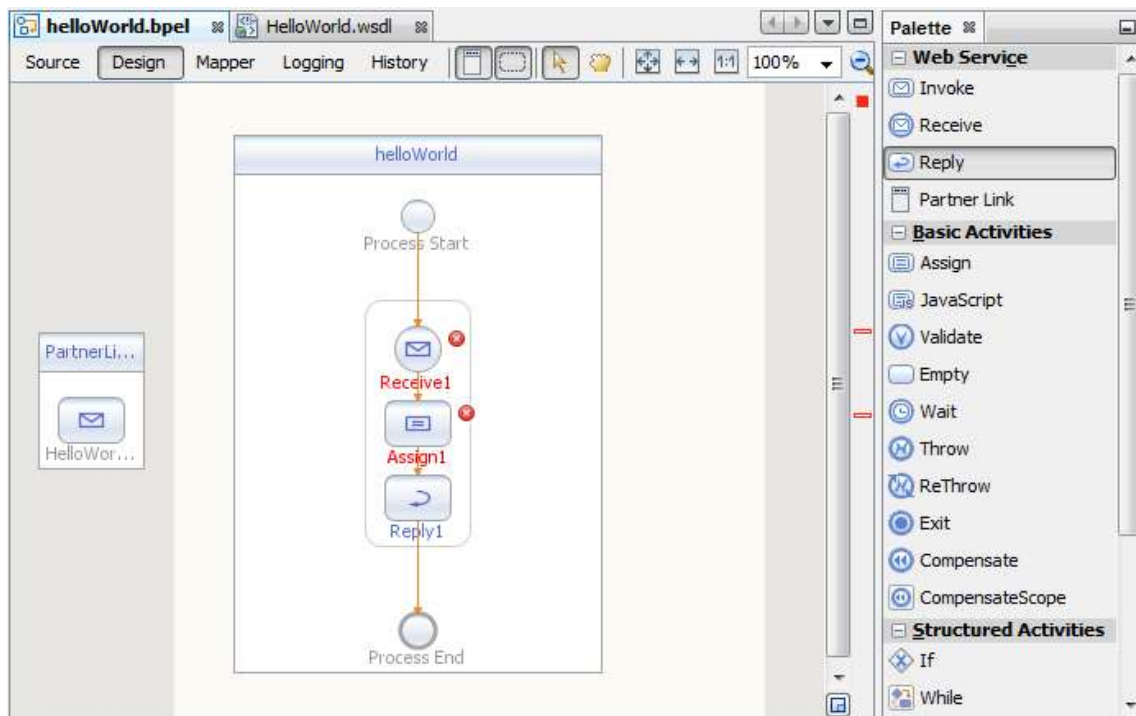
Follow the same steps to the Assign activity.



Drag and Drop the Assign icon on the Orange beneath the Receive activity.



Do the same with the Reply icon.



6.2.2 Link the activities with the PartnerLink(s)

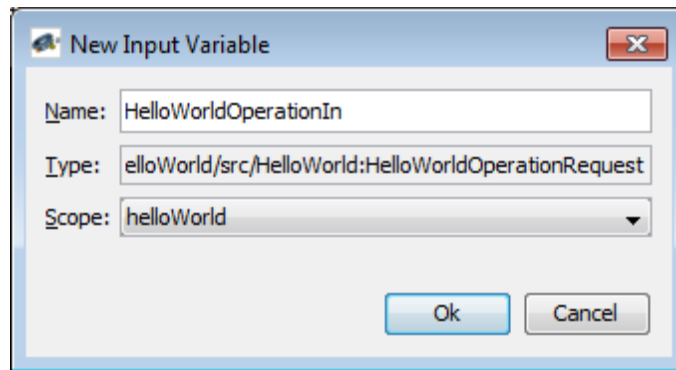
Click on the Receive icon, a small box appears. Click on the small box to edit the Activity properties.



Partner Link: Select PartnerLink1

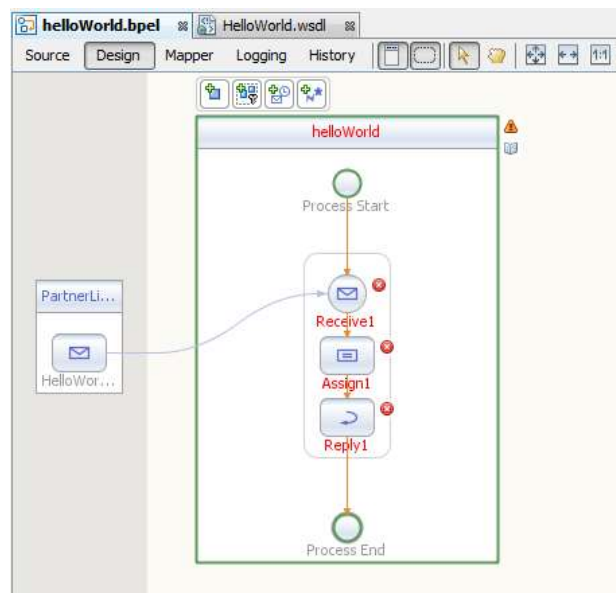
Operation: Select HelloWorldOperation

Input Variable: Click on Create button.



Keep the values by default. Then click on OK.

Click OK to validate the properties.

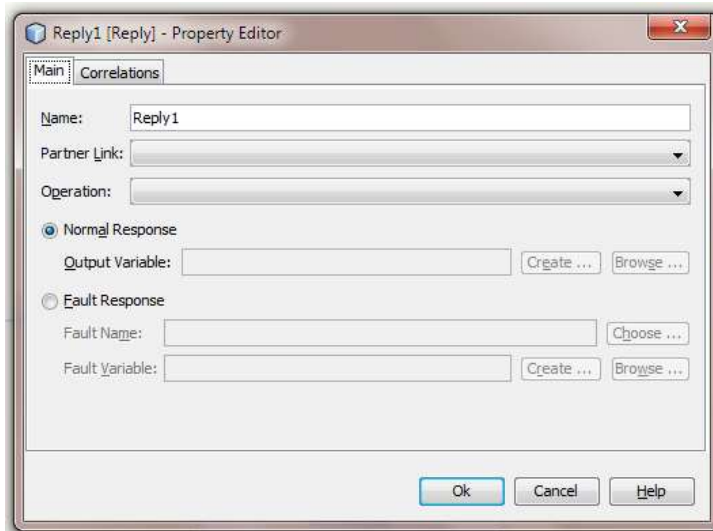


A link has been created between PartnerLink1 and the Receive activity.

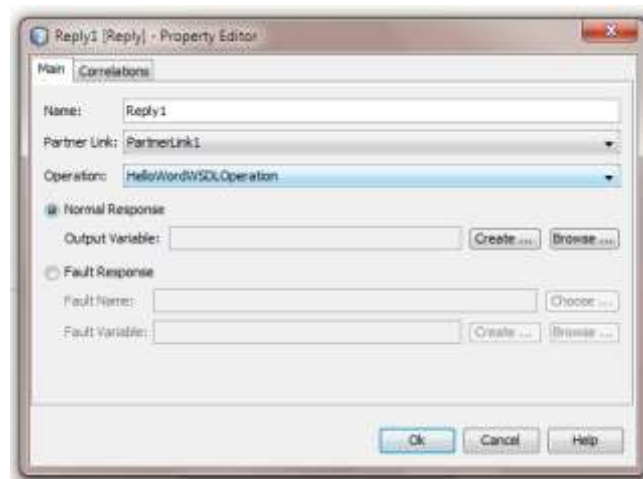
Click on the Reply1 activity then click on the properties box.

Do the same process with Reply. Double click on Reply1.





The pop-up Reply-Property Editor opens. Select the default values for “Partner Link” and “Operation” and set this Reply activity as follows.



Click on Create.

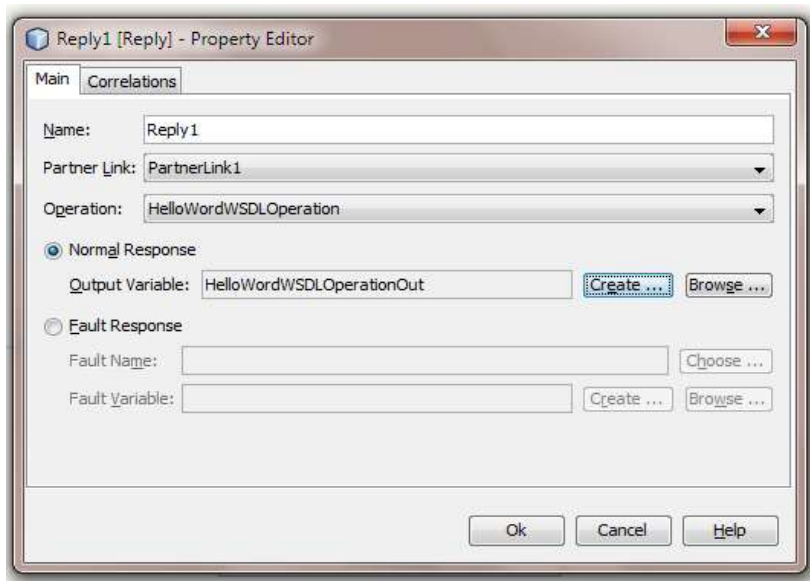


Keep the default values and click on OK.

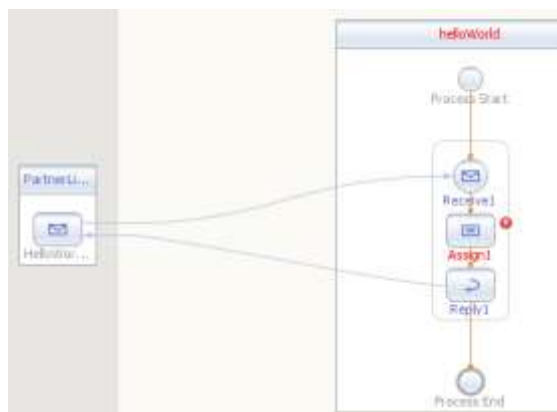
Doc: 770-002: OpenESB Standalone Enterprise Edition Hello World

Copyright © Pymma Services 2014-2018. All Rights Reserved.

Page 29 of 42



The Receive activity is now set up. Click on OK.

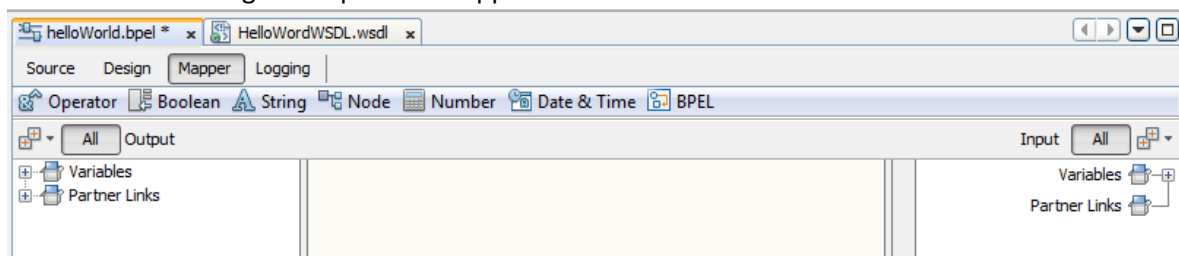


The set-up creates a link between the partner link and the activity Reply. the

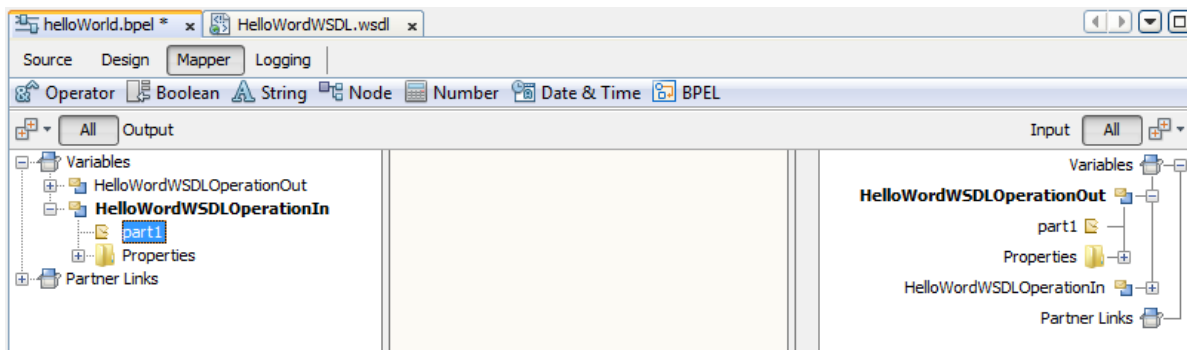
6.2.3 Setup Assign activity

Assign is the activity that maps the input messages to output messages. It is the place where we concatenate "Hello"+ the input name.

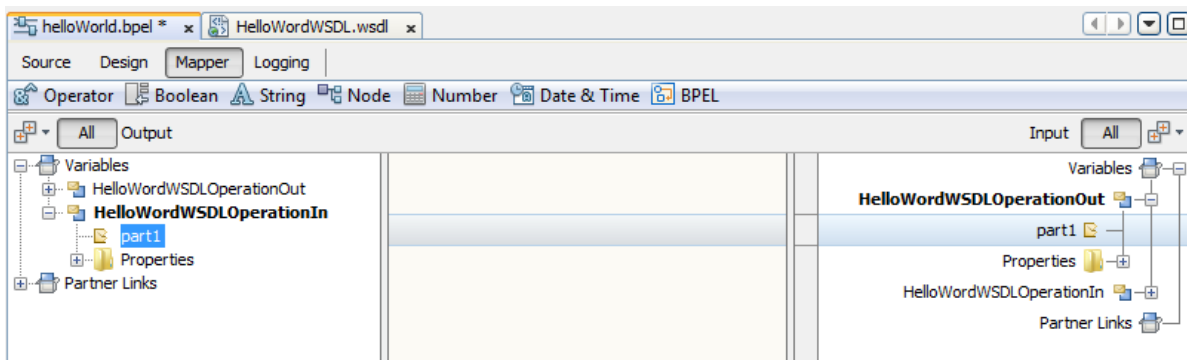
Double click on Assign1 to open the mapper editor.



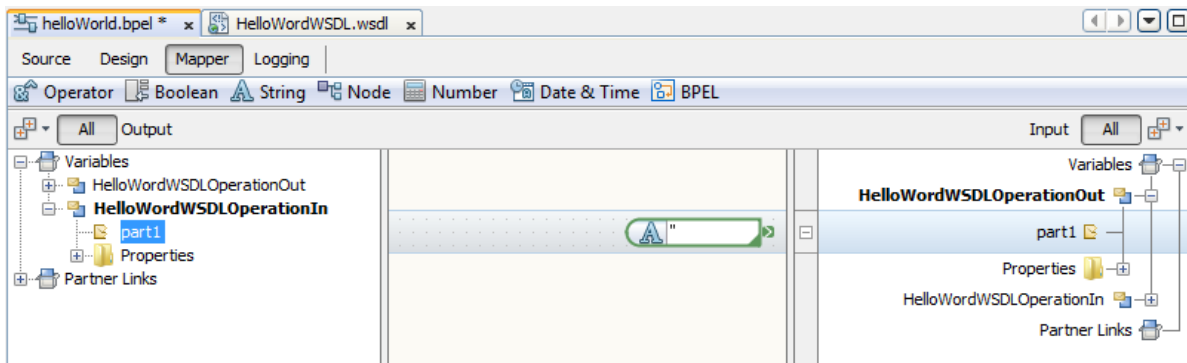
Expand the variable nodes on both sides.



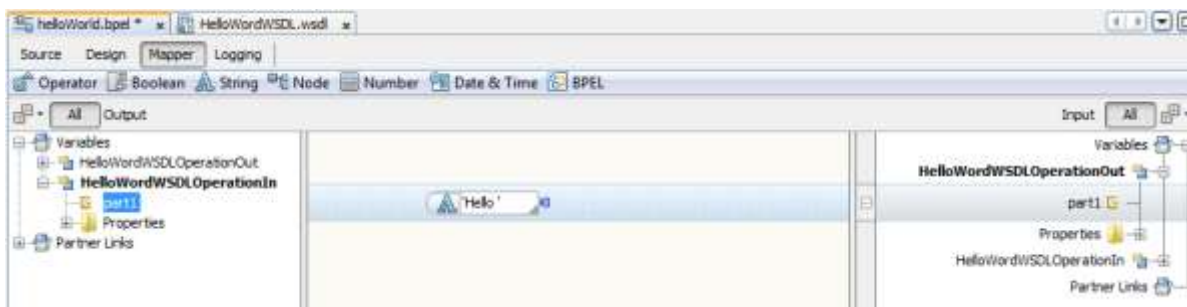
Click on HelloWorldWSDLOperationOut→part1. A blue lane appears at the part1 level.



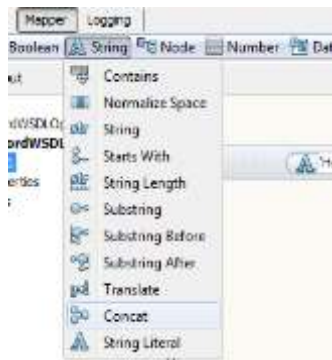
Select the Icon “String’ on the mapper editor top menu and select the item “String Literal”.



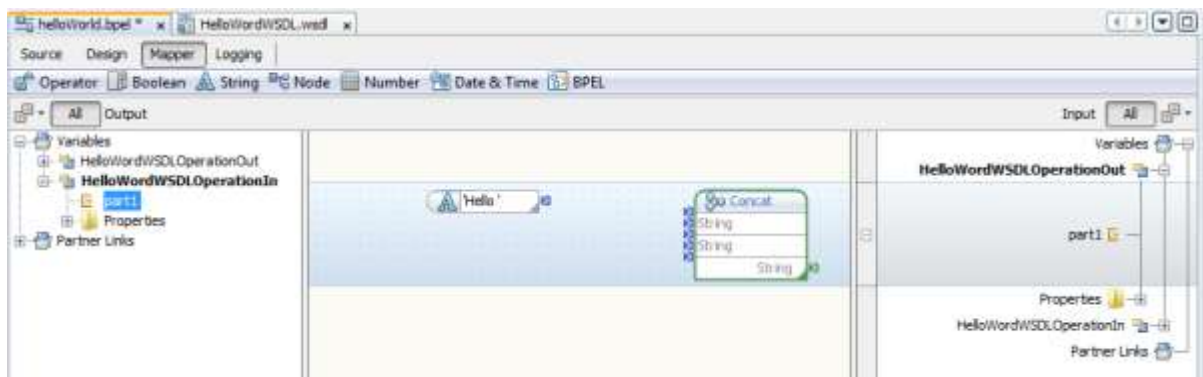
A new icon appears on the blue lane. It is the place to store “Hello ”. Double click on the icon and type “Hello” (without “ but with a space after Hello).



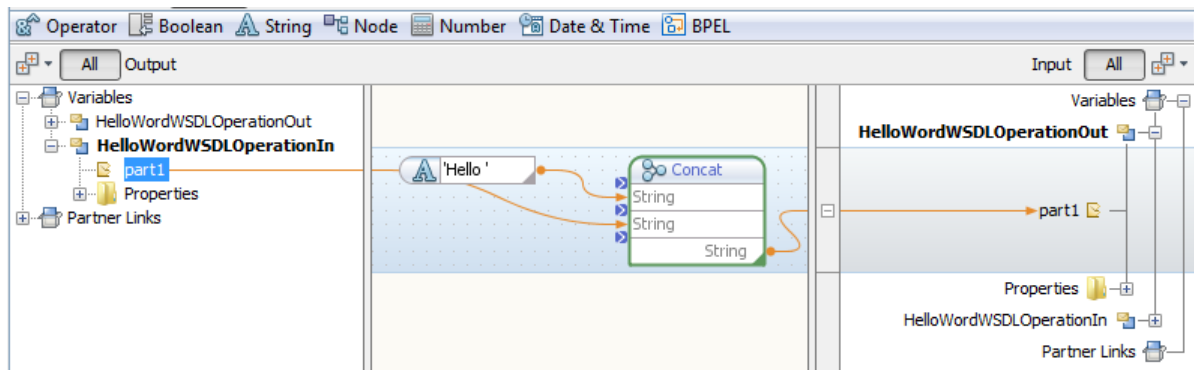
Select the Icon “String’ on the mapper editor top menu and select the item “Concat”.



A Concat icon is added in the blue lane next to “Hello”.



Drag and Drop then link the element as shown below.

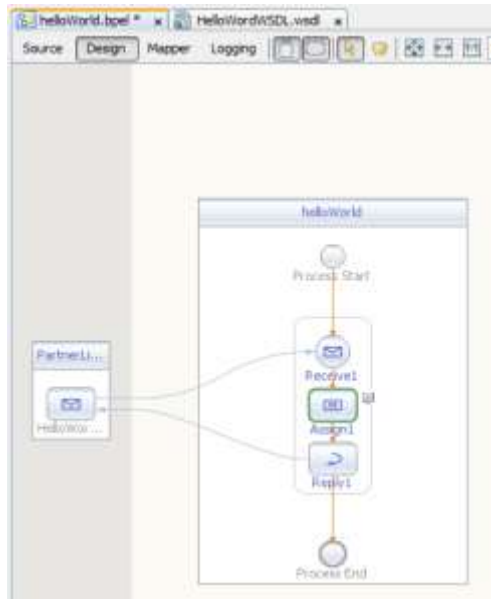


1-Link Hello to Concat input.


2-Link HelloWorldWSDLOperationIn→part1 to another Concat input.

3-Link Concat output to HelloWorldWSDLOperationOut→part1.

Concat = concatenates Hello + the name found in the input message and put the result into the output message.



Click on the Design tab and come back to the BPEL editor. Note that the red signs are not present anymore; it indicates your BPEL is valid without error.

Save your BPEL. Click on  icon or type ctrl-shift-S, then close the BPEL editor.

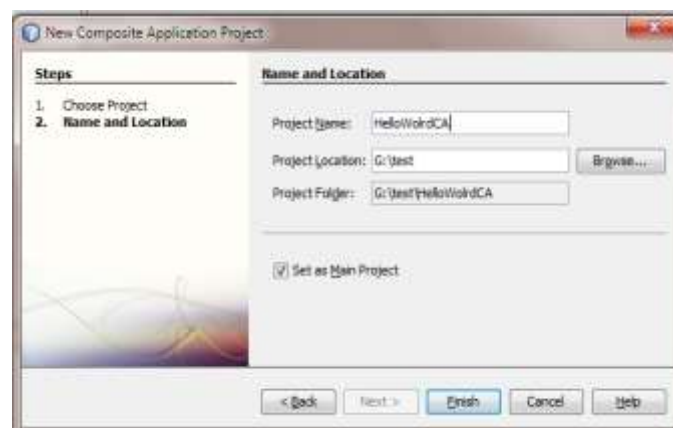
7 Create a Composite Application

To deploy an application with OpenESB, you create a composite application. A composite application is a deployable entity for OpenESB.

Right-click on the project tab and select “New project “. Select the SOA Category and the Composite Application project.

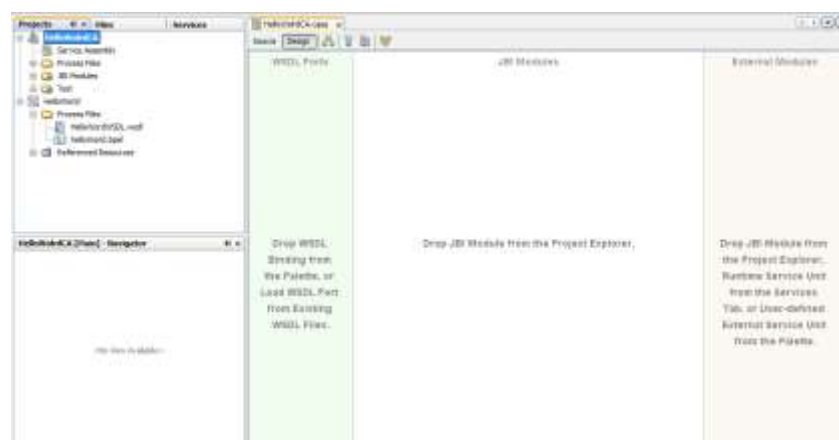


Click on Next and set the project name.



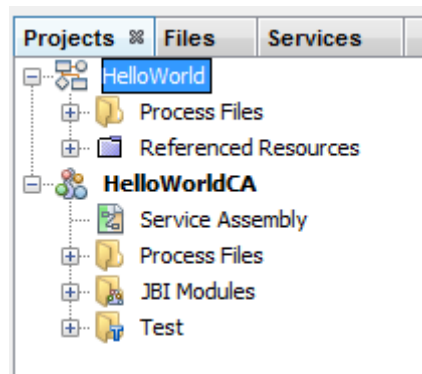
Click on Finish

After a while, HelloWorldCA project is added into the Project tab, and the Composite Application Editor opens.



7.1 Add the BPEL Module in a composite application

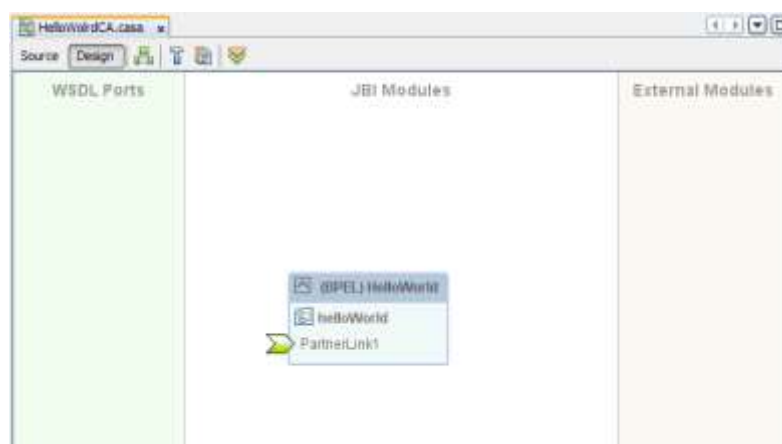
Select HelloWorldCA project.



Drag it and drop it in the JBI Module Lane, in the Composite Application editor.



Click on the build button  represented by a hammer.

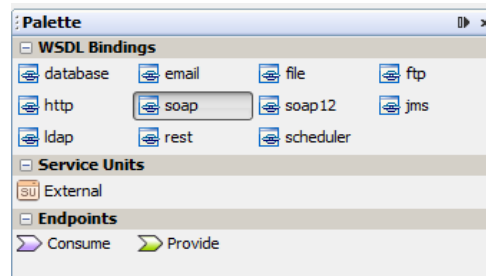


Composite Editor introspects Hello Word project and finds out the partner links associated with the project.

7.2 Add a Binding Component

Our project is ready to be deployed, but we have to precise how the project receives the messages from the outside world. To send messages to the BPEL module, we use a SOAP protocol. You have to define a SOAP port where the message are sent.

On the right part of the screen, in the palette, select the icon SOAP (not SOAP12).

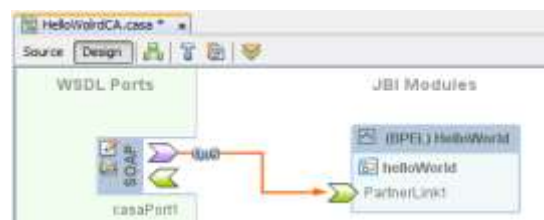


Drag the icon and drop it on the WSDL Ports lane in the Composite application editor.



A SOAP icon appears on the WSDL Ports Lane.

With a simple drag and drop Link SOAP and BPEL.

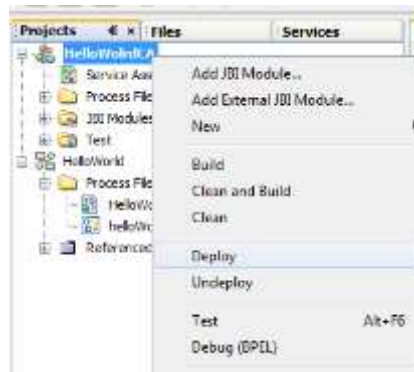


Save the project: Type Ctrl-shift-S.

8 Deployment

8.1 Deploy the composite application

Go back to the Projects tab. Select HelloWorldCA. Click right and select “Deploy”.



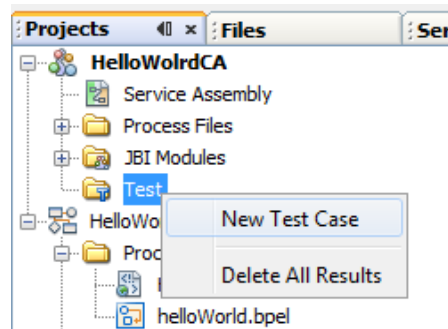
After a few seconds, at the bottom of your screen, in the build.xml tab in the output window, check if the build is successful.

```
Output - build.xml (run)
Output - build.xml (run) :
[deploy-service-assembly]
    Deploying a service assembly...
        host=localhost
        port=8699
        file=G:\temp\poubelle\HelloWorldCA\dist\HelloWorldCA.zip
[start-service-assembly]
    Starting a service assembly...
        host=localhost
        port=8699
        name=HelloWorldCA
run:
BUILD SUCCESSFUL (total time: 3 seconds)
```

The application is deployed.

9 Test the application.

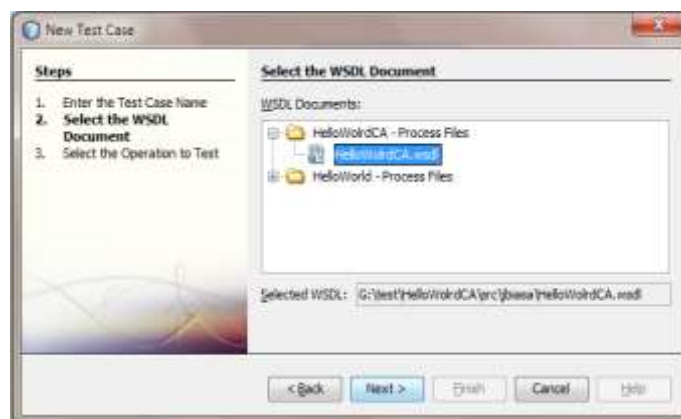
Now let's test the application. The OpenESB has a simple test generator. It allows you to test any SOAP port in your applications. Select the node "Test" in the HelloWorldCA project. Right-click and select "New Test case".



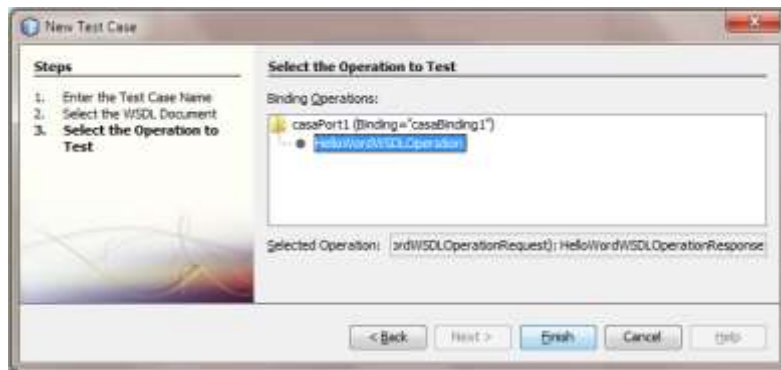
Give a name to the test case and click Next.



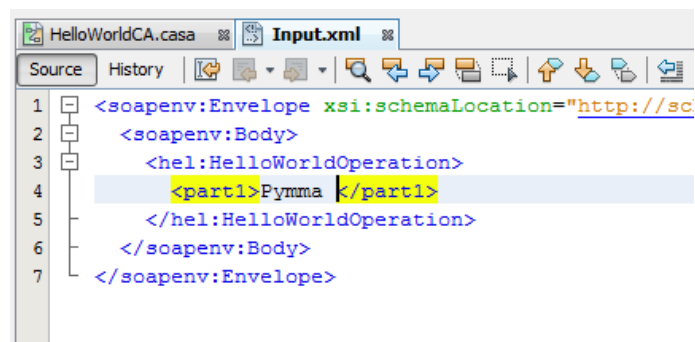
Select the HelloWorldCA and click Next.



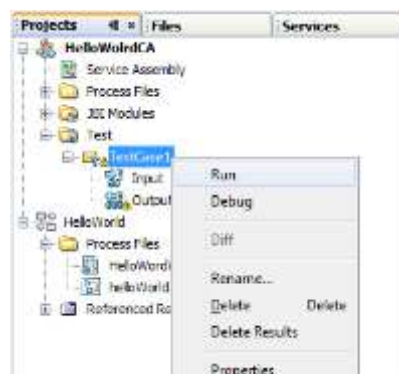
Select the HelloWorldWSDLOperation and click Finish.



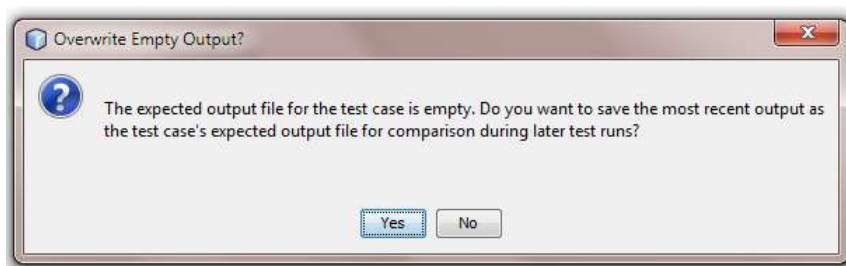
The XML Editor opens. Type your name in the Part1 (Ex: Pymma).



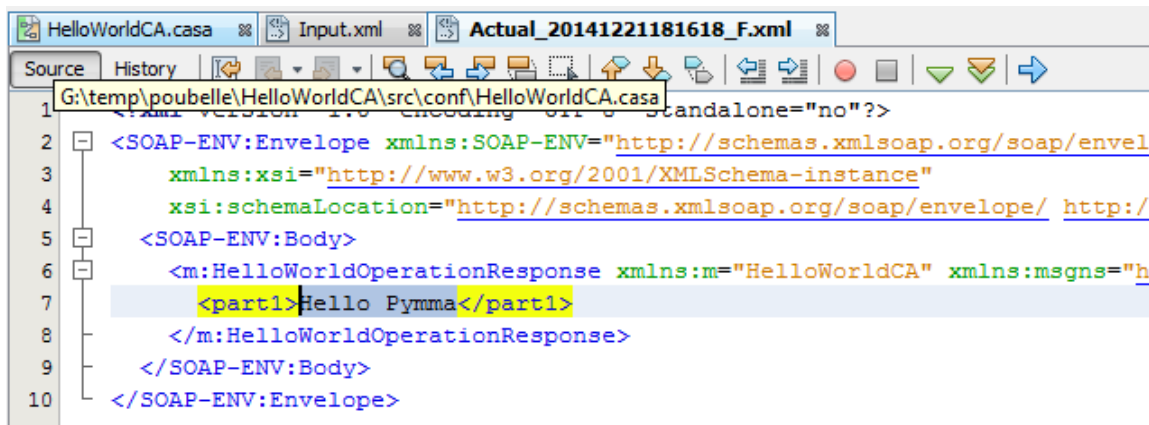
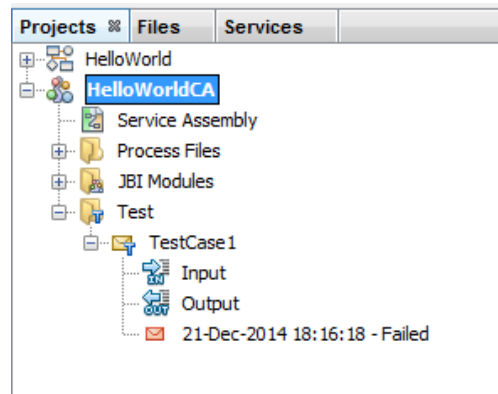
Save the XML document: Ctrl-Shift-S. The new test case "TestCase1" appears in the hierarchy. Run TestCase1.



After a while, the following pop-up appears.



Double click on the output and ignore the message saying Failed that opens afterwards.



The unit test failed because we did not set up an output for the test. Do the test again, and you won't get an error.

Congratulation!!! It works. You're the best ;-).

10 Next steps

The next step will be to install the OpenESB components and Libraries on your instance then create and deploy your project with the OpenESB Studio.

We advise you to read our documents: **770-003 OE Web admin console** and the administrative guide: **770-004 OE Admin Guide**.

11 Help and support

Pymma is deeply involved in the community and offers services and consulting on OpenESB. Pymma has professional services that can assist you with the development of your service design, implementation and ongoing management. All our skills and background are based on our extensive first-hand experience and industry-leading methods.

Pymma releases an OpenESB Enterprise Edition with many additional enterprise features and professional support.

In addition to OpenESB development, Pymma designed a new service-based development process named Rebecca to help business, architect and development team during the design and the implementation of their service-oriented projects with OpenESB or any other service-based development tool.

Feel free to contact us by email at contact@pymma.com for any further information on our OpenESB Services.