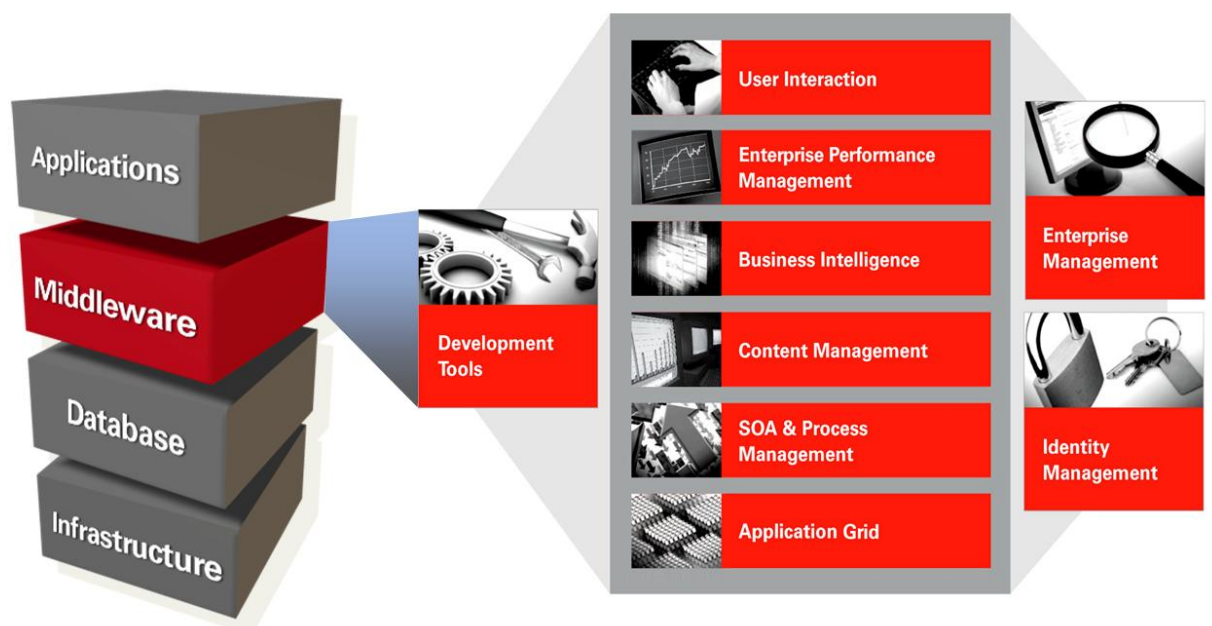


## Part I - Introducing SOA, St Matthews and the Oracle SOA Suite

### Chapter Complement 3 - Oracle Fusion Middleware and SOA Suite 11g

Note: the steps that describe the installation and configuration of the SOA Suite 11g, OSB 11g and BPM 11g have been included in a separate document that will be updated as later releases become available. This document can be found through the book's wiki at <http://wiki.oracle.com/page/Oracle+11g+SOA+Suite+Handbook>, on the page for Chapter 3.



#### (1)Getting started with SOA Suite 11g

This book is by no means intended to be only theoretical. Yes, we will tell a story and hopefully explain a great deal about the SOA Suite by showing examples and describing the concepts behind and workings of the service engines, underlying standards and technologies and supporting tools. However, you - and your fingers - will only start to learn for real once you start practicing what we preach. So now is the time to get into gear. Time to get yourself a fully operational SOA Suite 11g that will allow you to start develop, deploy and run composite service applications and start to appreciate what it is really like to create a service oriented applications.

## ***Installation***

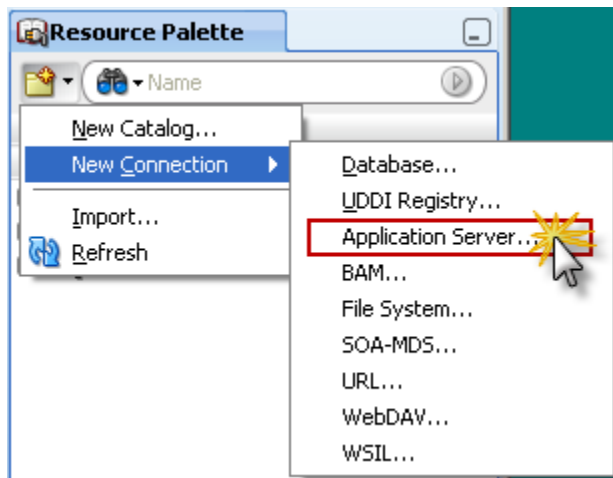
The installation is explained and demonstrated in a lot of detailed instructions and screenshots in the on line document on the book's wiki, on the page for Chapter 3:

<http://wiki.oracle.com/page/Chapter+3+-+Introduction+of+Oracle+SOA+Suite+11g>.

## ***Post installation Configuration steps***

To deploy SOA applications directly onto the SOA Suite we need to configure an Application Server Connection in JDeveloper, to the WebLogic Server soa domain.

To create this connection start JDeveloper. Go to the Resource Palette. Click on the *New* icon, select *New Connection* and pick the Application Server connection type from the list.



Enter *FMW11g\_SOASuite11g\_local* as the name for the new Application Server connection and select WebLogic 10.3 as the connection type.

**Create Application Server Connection - Step 1 of 5**

### Name and Type

Specify a unique name and type for the connection. The name must be a valid Java identifier.

Create connection in: Resource Palette

Connection Name:  
FMW11g\_SOASuite11g\_local

Connection Type:  
WebLogic 10.3

Help < Back Next > Finish Cancel

Press the Next button. Provide the authentication details: again the weblogic/weblogic1 username/password combination.

**Create Application Server Connection - Step 2 of 5**

### Authentication

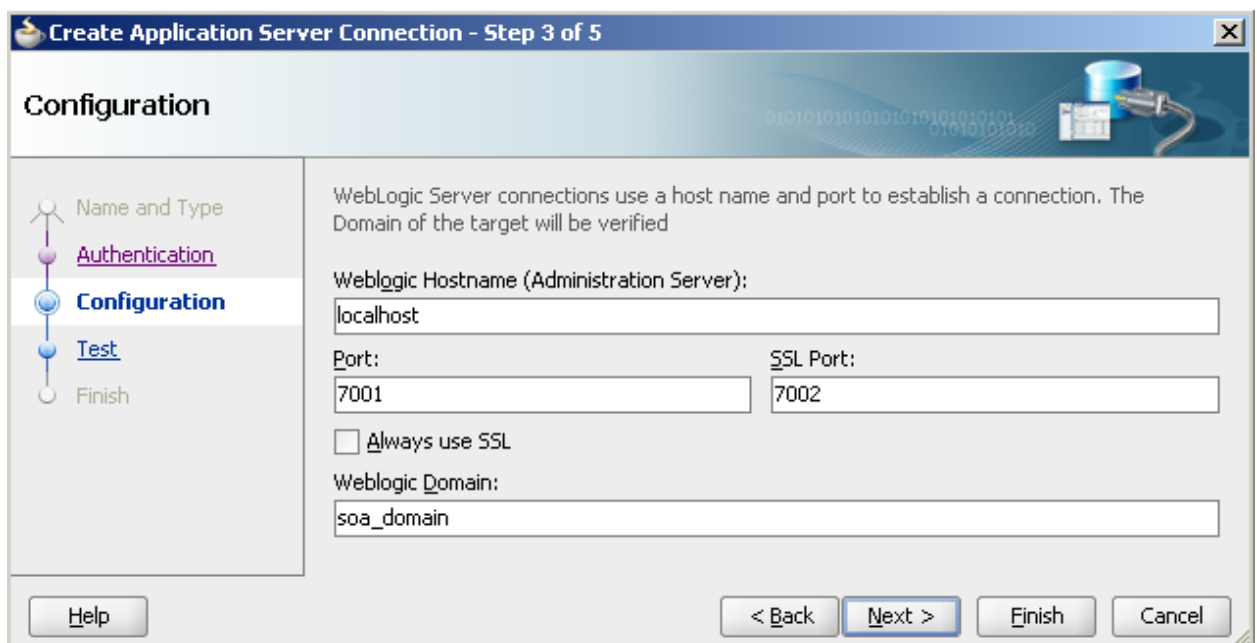
Specify a username and password to authenticate the connection.

Username:  
weblogic

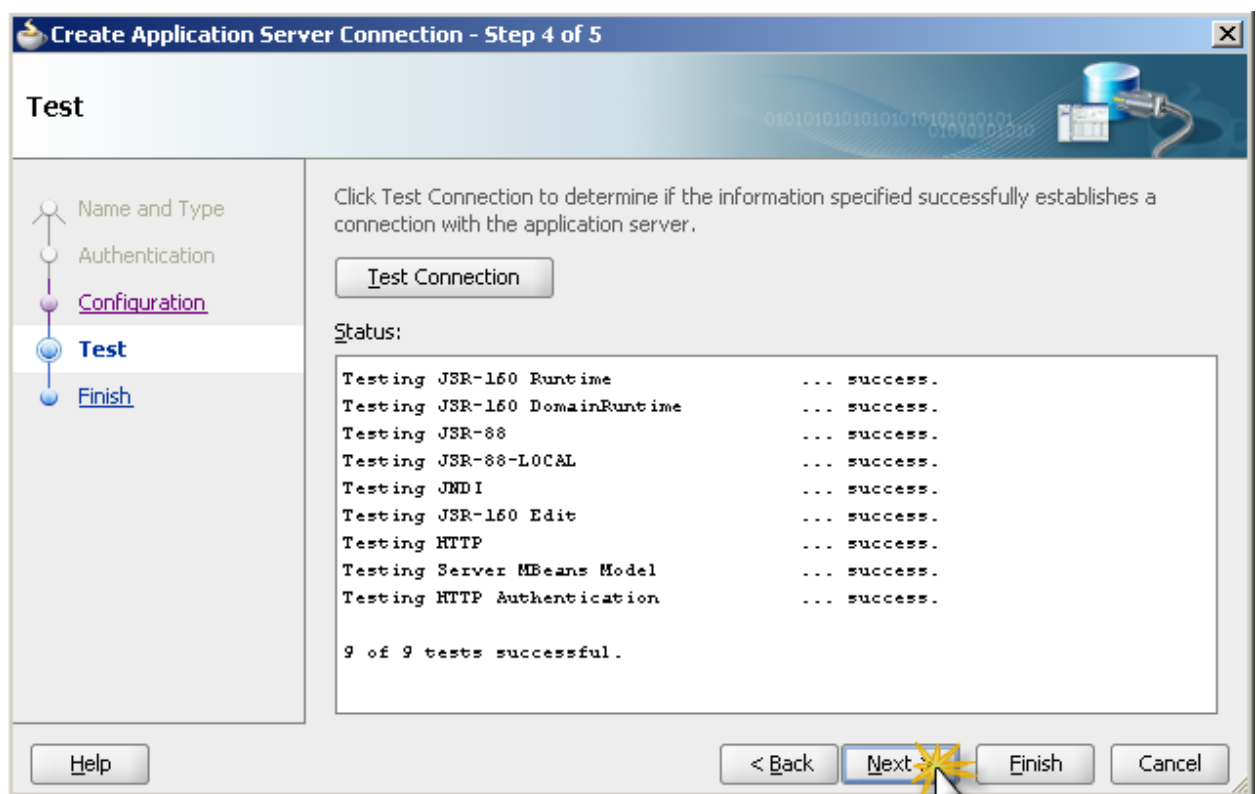
Password:  
weblogic1

Help < Back Next > Finish Cancel

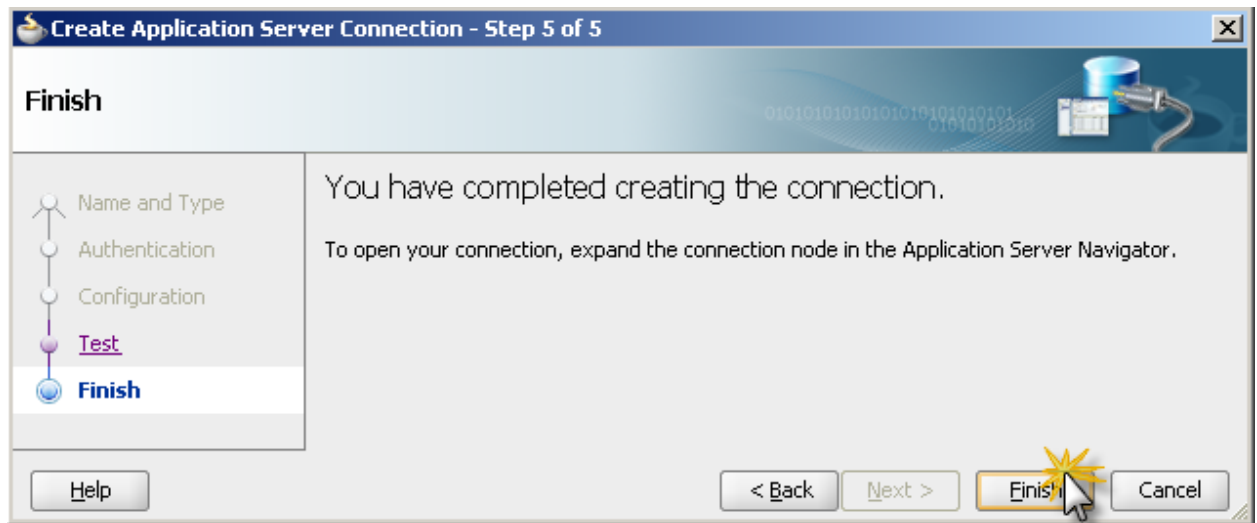
Next you need to indicate the WebLogic Hostname (localhost for local installations) and the port for the AdminServer - 7001 by default. Also enter the name for the domain you connect to; the domain name that was suggested in the installation instructions is soa\_domain.



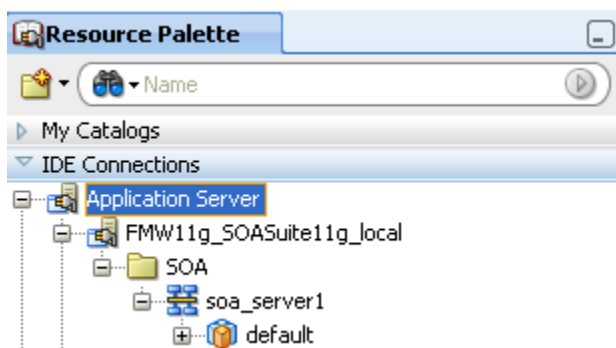
Press the *Next* button again. When you test the connection, you should receive a number of success messages, one for each of the different ways of connecting to the domain.



The summary page is next:



Press the Finish button to close the Create Connection dialog. The new connection is now available on the Resource Palette.



The connection provides insight in the SOA contents of the application server's `soa_domain`, which is the `soa_server1` managed server and inside it the *default* partition.

## Create and Run the “HelloWorld” of Service Composite Applications

This section will walk you through the creation, deployment and test run with SOA Suite 11g of the world’s most basic SOA composite application. At the end of this section - after maybe ten minutes worth of work - you will have your first application running in the SOA Suite. The steps are:

1. Fire up the engines: first, start the database that hosts the Meta Data Repository and then the WebLogic Servers in the SOA domain (AdminServer and the managed soa\_server1) using the command line scripts:

locate the startWebLogic.cmd script (Linux: startWebLogic.sh) in the MIDDLEWARE\_HOME \user\_projects\domains\soa\_domain directory. Run this script from the command line or terminal.

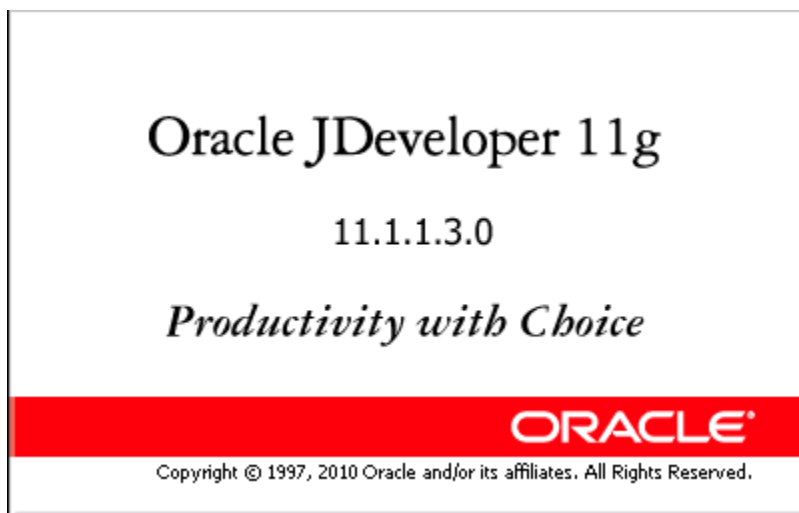
```
E:\Oracle\Middleware\user_projects\domains\soa_domain\startWebLogic.cmd
```

When the AdminServer is running, we should start the SOA Server. Go to the directory MIDDLEWARE\_HOME \user\_projects\domains\soa\_domain\bin that was created by the SOA Suite configuration wizard. Open a command window and enter the following command to execute:

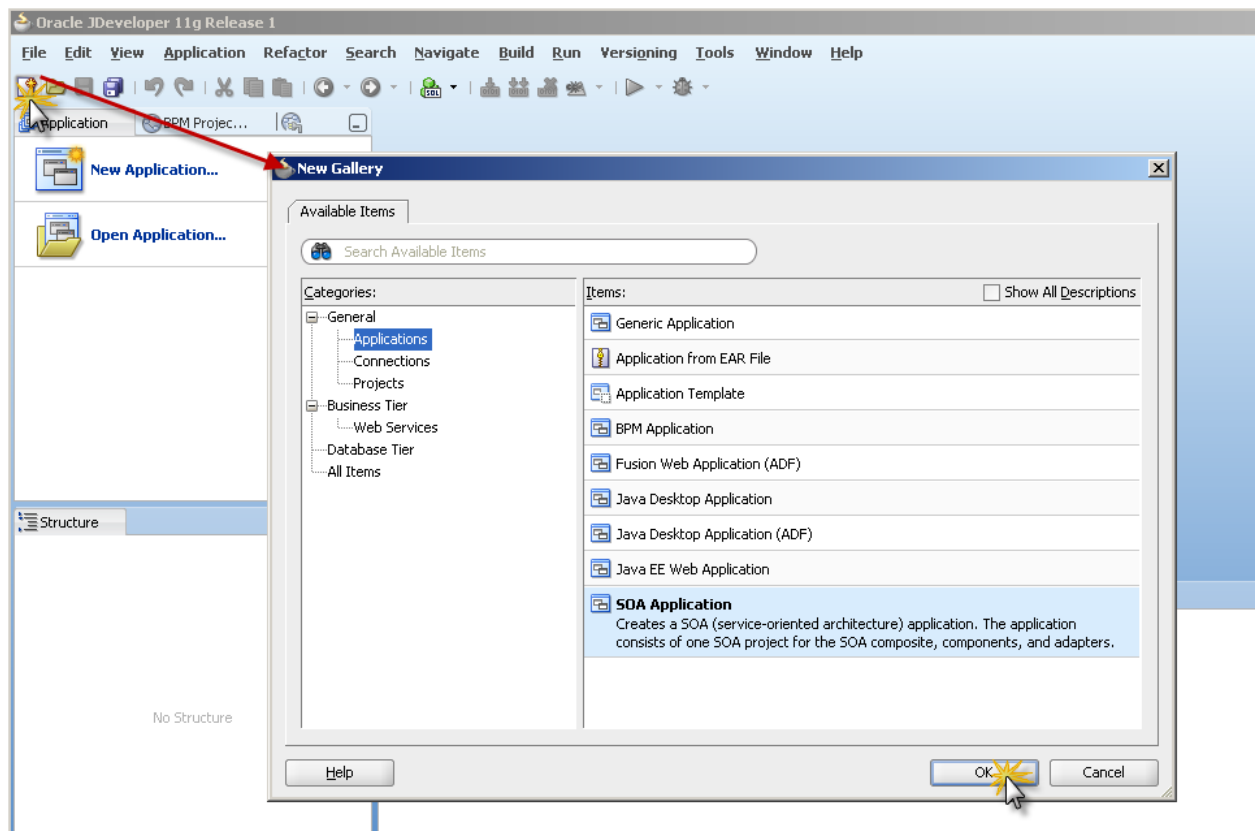
```
startManagedWebLogic.cmd soa_server1
```

(on Linux use the script startManagedWebLogic.sh) from the command line or terminal.

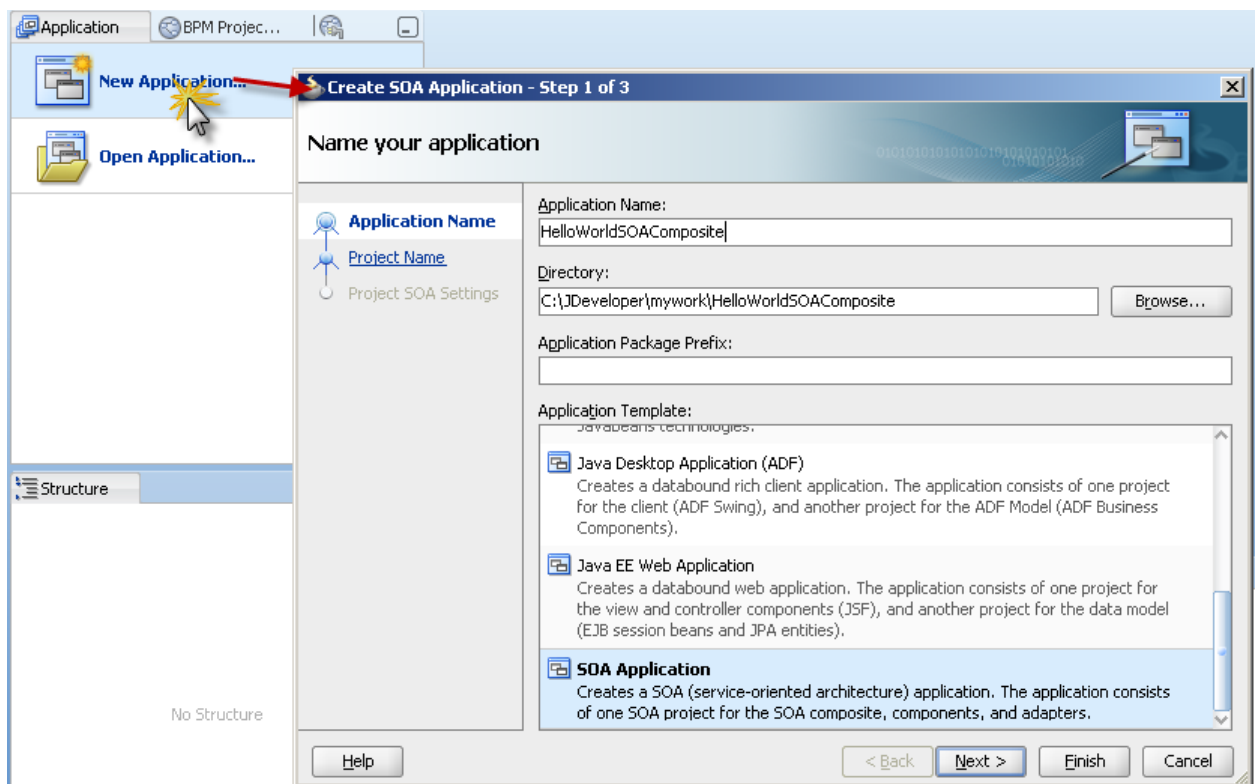
2. Start JDeveloper (choose Default Role if you are prompted to select a role)



3. Select New from File menu.



From the New Gallery that is presented next, select the SOA Application item in the Applications Category (under the General node).



Press the OK button to continue.

You will be prompted to provide a name for the application - enter HelloWorldSOAComposite. Leave the Application Package Prefix field empty and press the Next button.



**Create SOA Application - Step 1 of 3**

### Name your application

**Application Name**  
**Project Name**  
Project SOA Settings

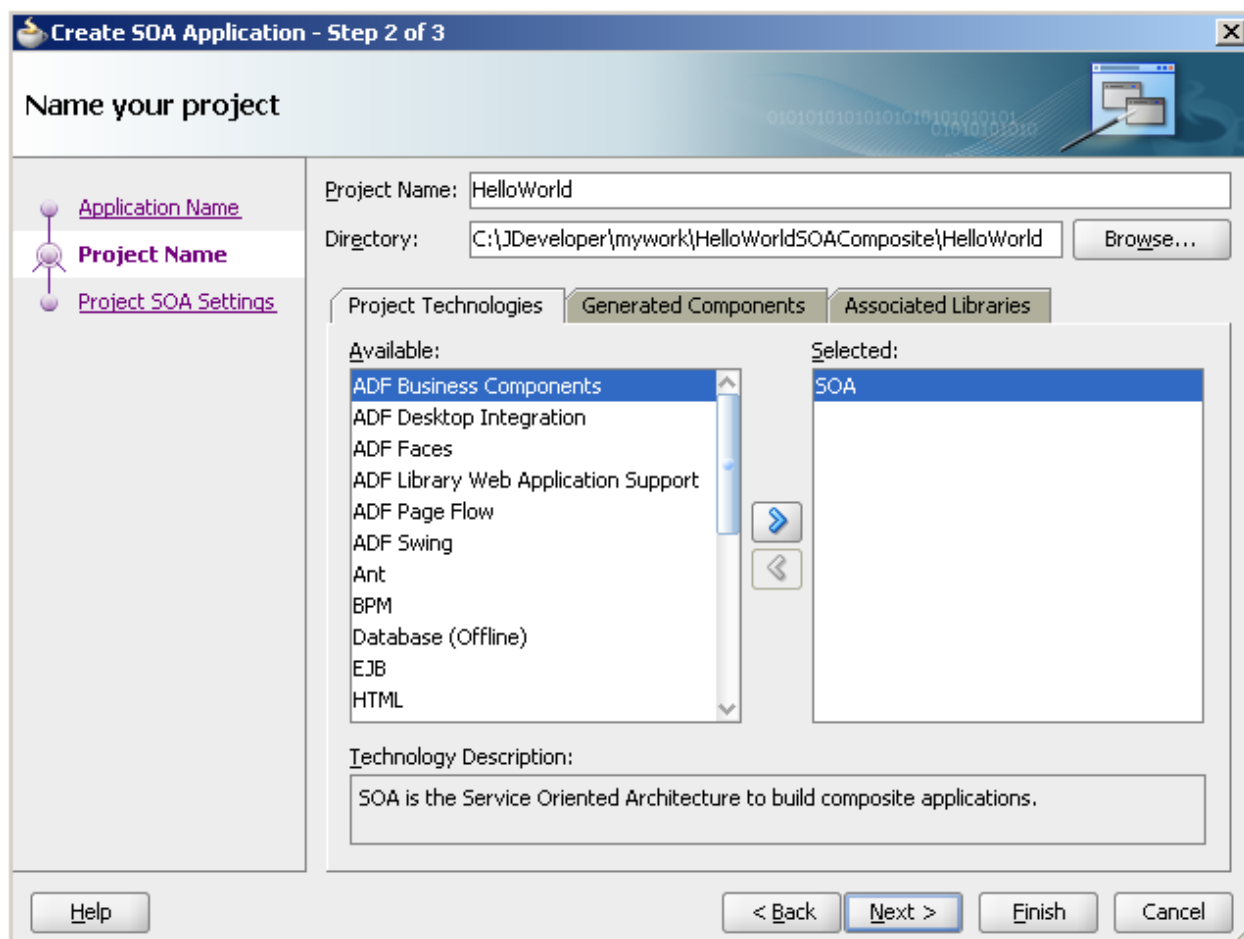
Application Name:  
HelloWorldSOAComposite

Directory:  
C:\JDeveloper\mywork\HelloWorldSOAComposite Browse...

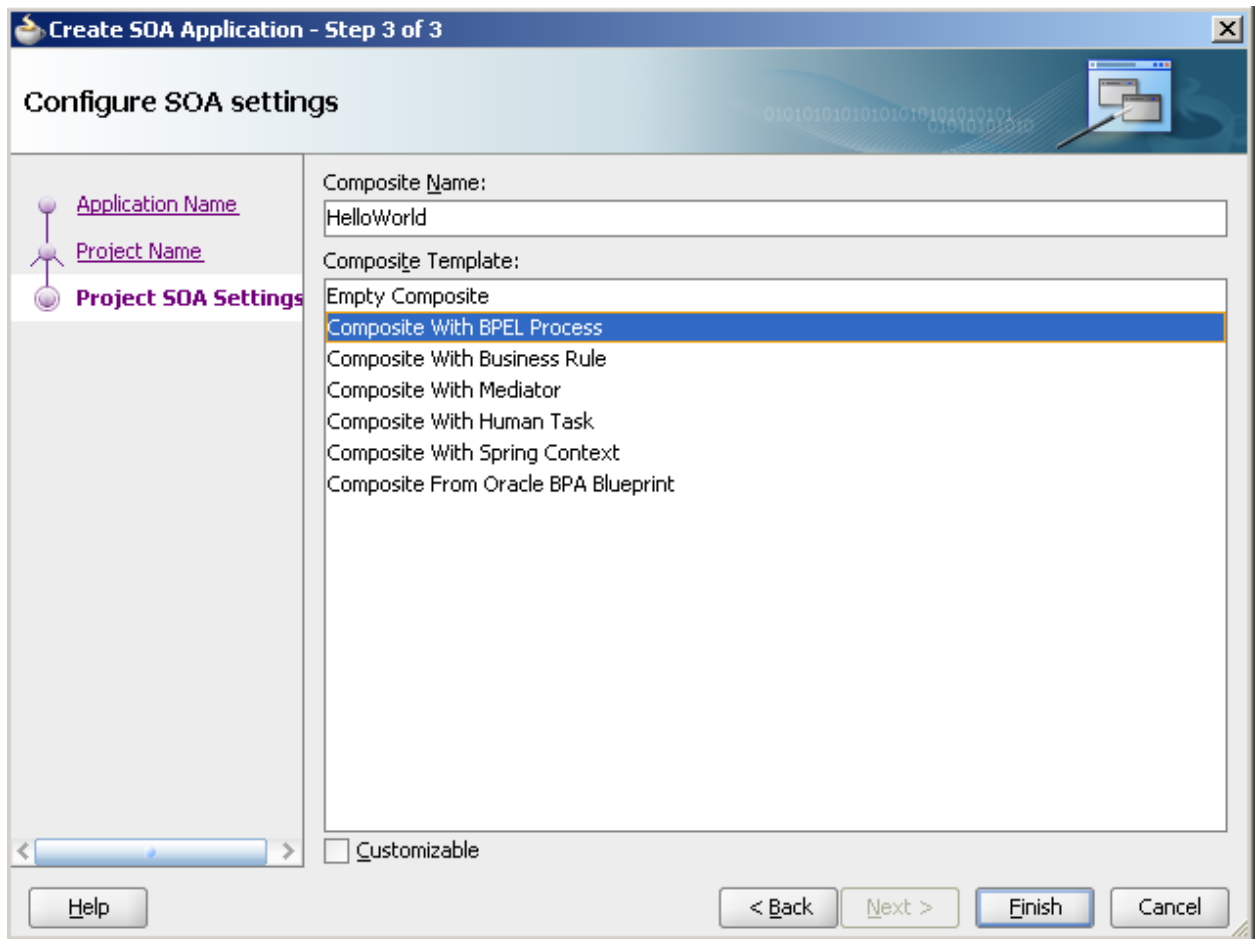
Application Package Prefix:

Help < Back Next > Finish Cancel

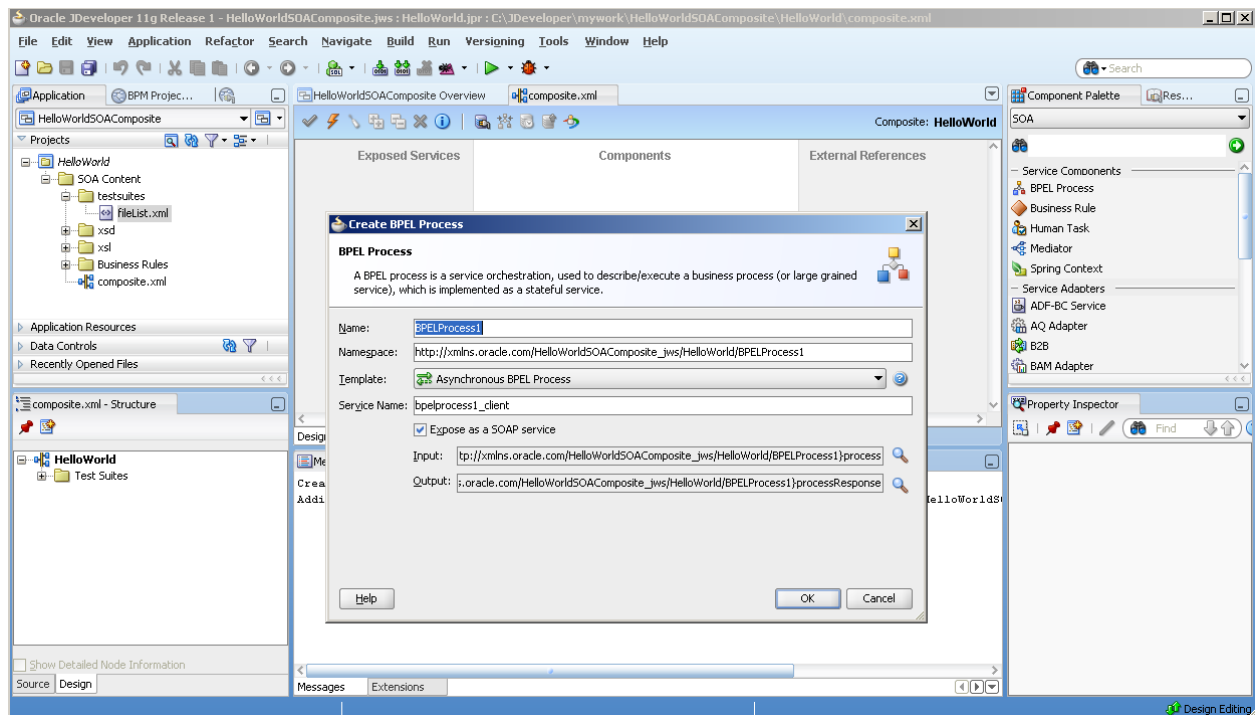
On the next page enter HelloWorld as the name of the project, and press Next again.



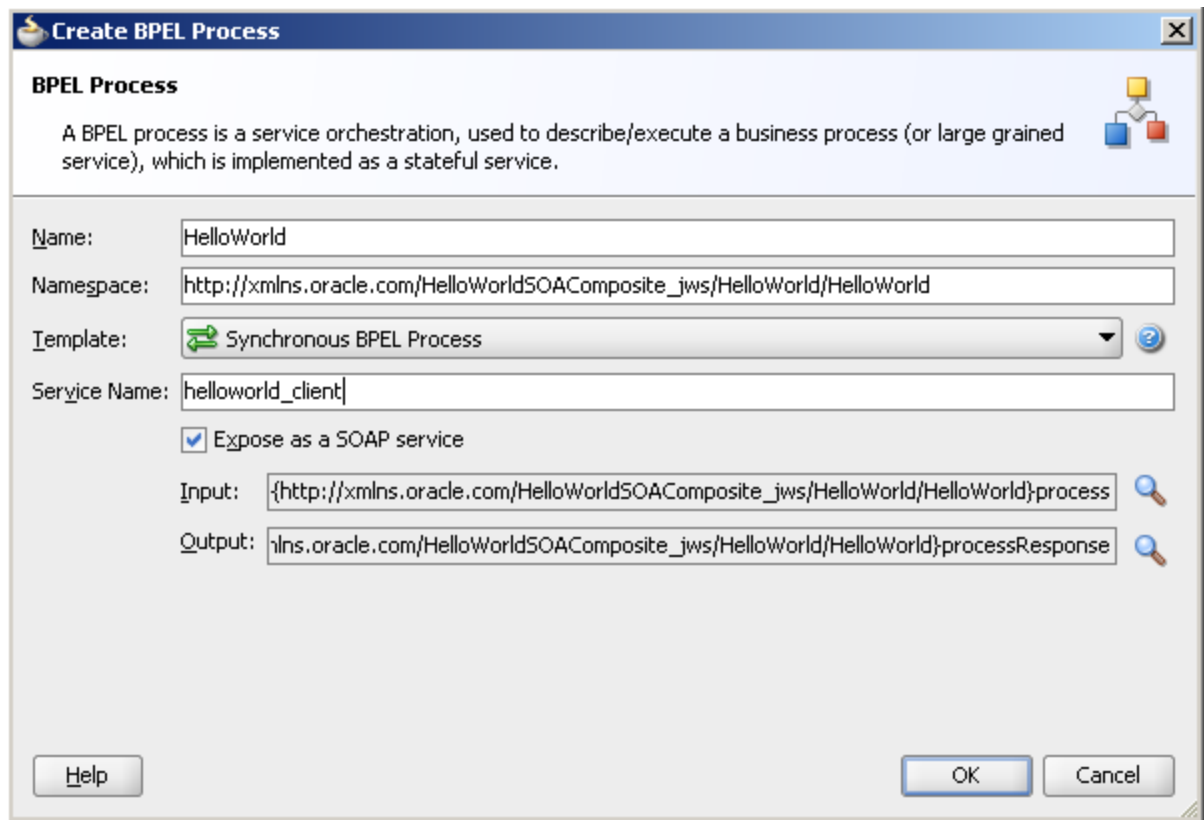
JDeveloper then asks you what type of composite application this will be; pick 'Composite with BPEL' on the Configure SOA settings step.



Press Finish to have the application, project and service composite created.



4. The Create BPEL Process dialog comes up next. Specify the name for the new BPEL process - HelloWorld - and the Template: Synchronous BPEL process. Make sure the checkbox *Expose as a SOAP Service* is checked and accept the other defaults for Namespace, Service Name and the input and output [variables].



**Create BPEL Process**

A BPEL process is a service orchestration, used to describe/execute a business process (or large grained service), which is implemented as a stateful service.

Name: HelloWorld

Namespace: http://xmlns.oracle.com/HelloWorldSOAComposite\_jws/HelloWorld/HelloWorld

Template: Synchronous BPEL Process

Service Name: helloworld\_client

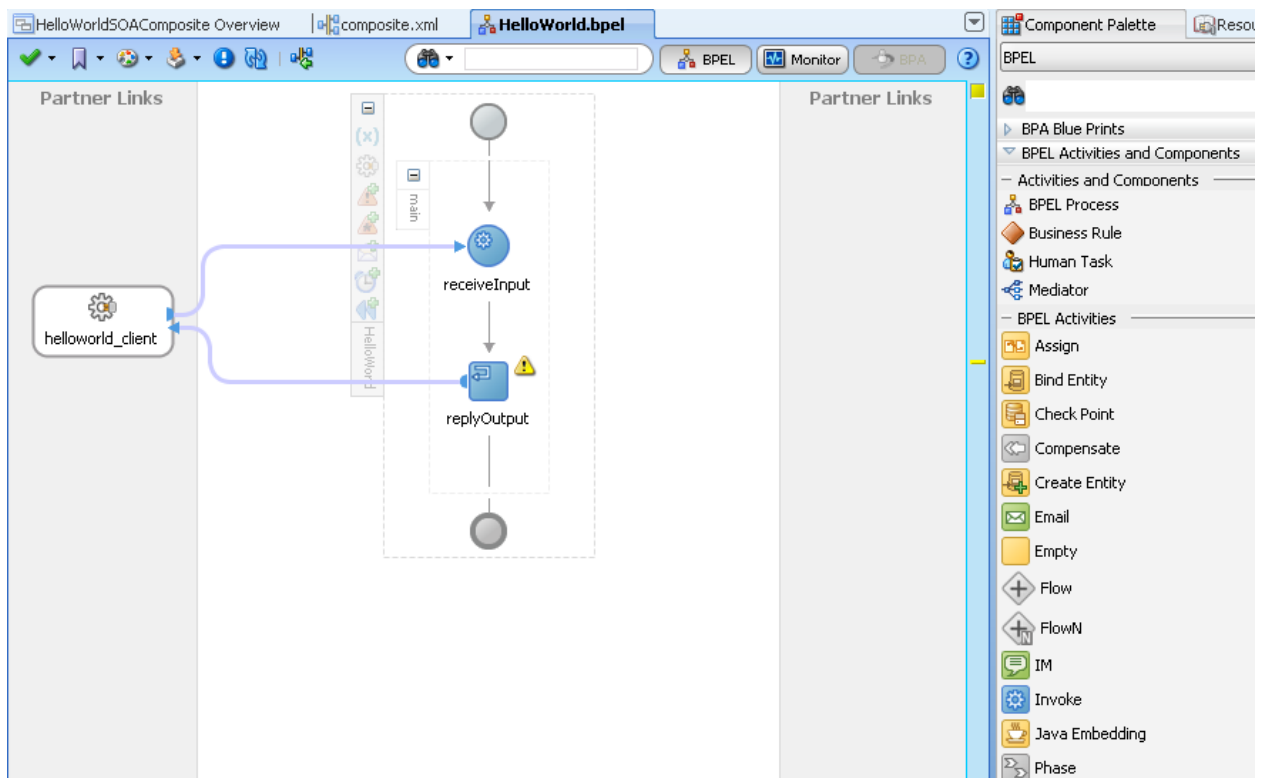
☒ Expose as a SOAP service

Input: {http://xmlns.oracle.com/HelloWorldSOAComposite\_jws/HelloWorld/HelloWorld}process

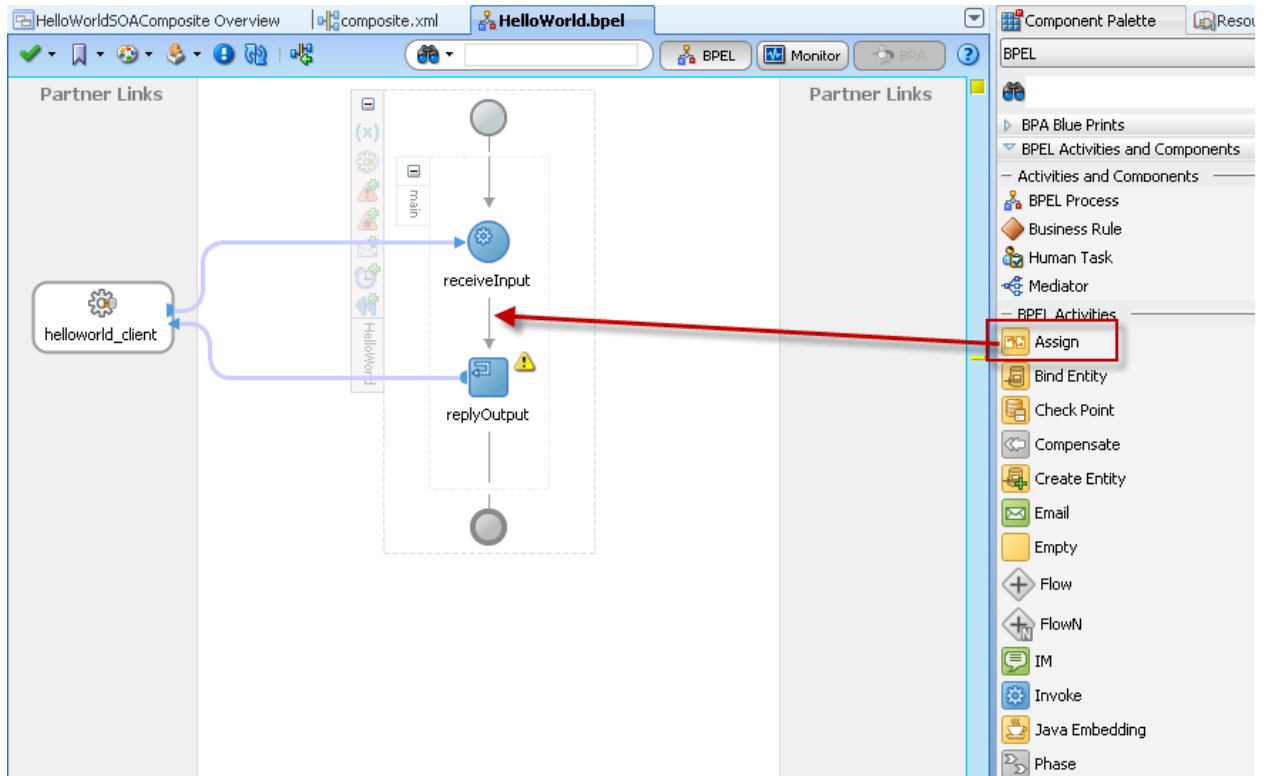
Output: {http://xmlns.oracle.com/HelloWorldSOAComposite\_jws/HelloWorld/HelloWorld}processResponse

Buttons: Help, OK, Cancel

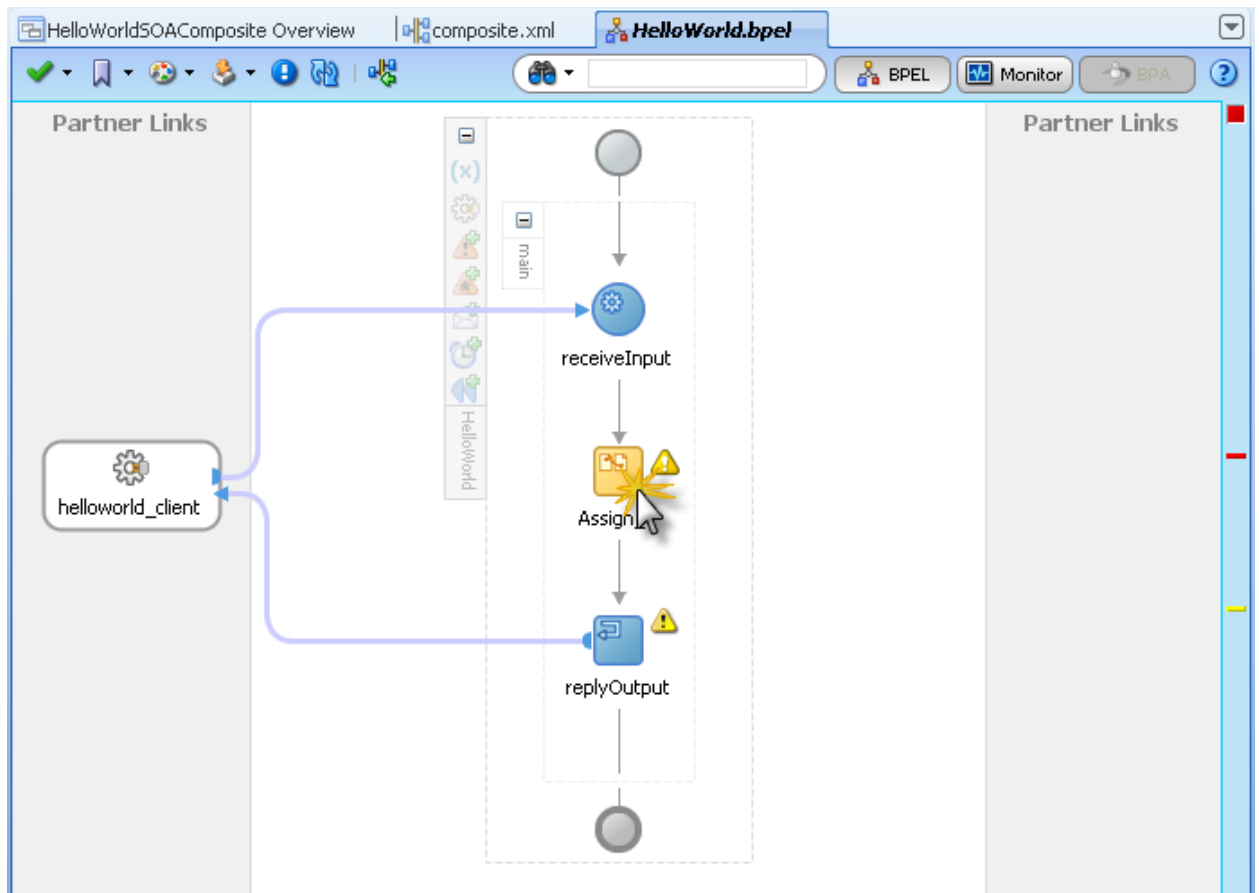
Press OK.



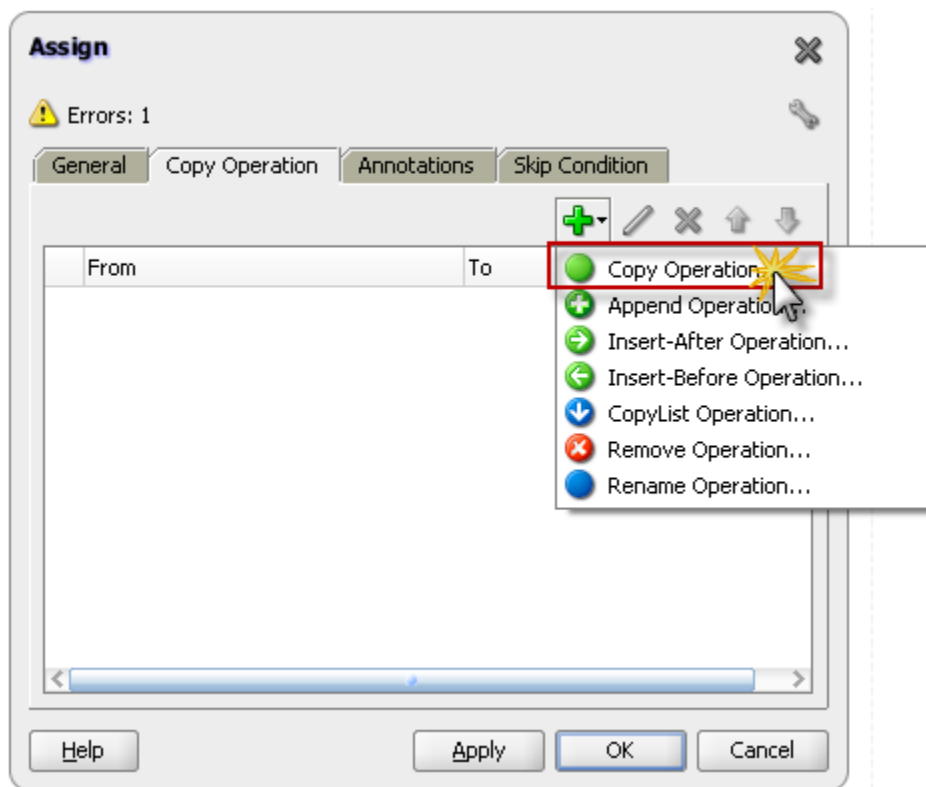
5. The BPEL editor opens up. You will see the basic structure of the BPEL process with a receive and a reply activity, by default configured to receive a single string and return a single string. You need to add one activity to set the value of that string result: drag an Assign activity from the Component palette



and drop it between the receive and reply activities that are already in the process.

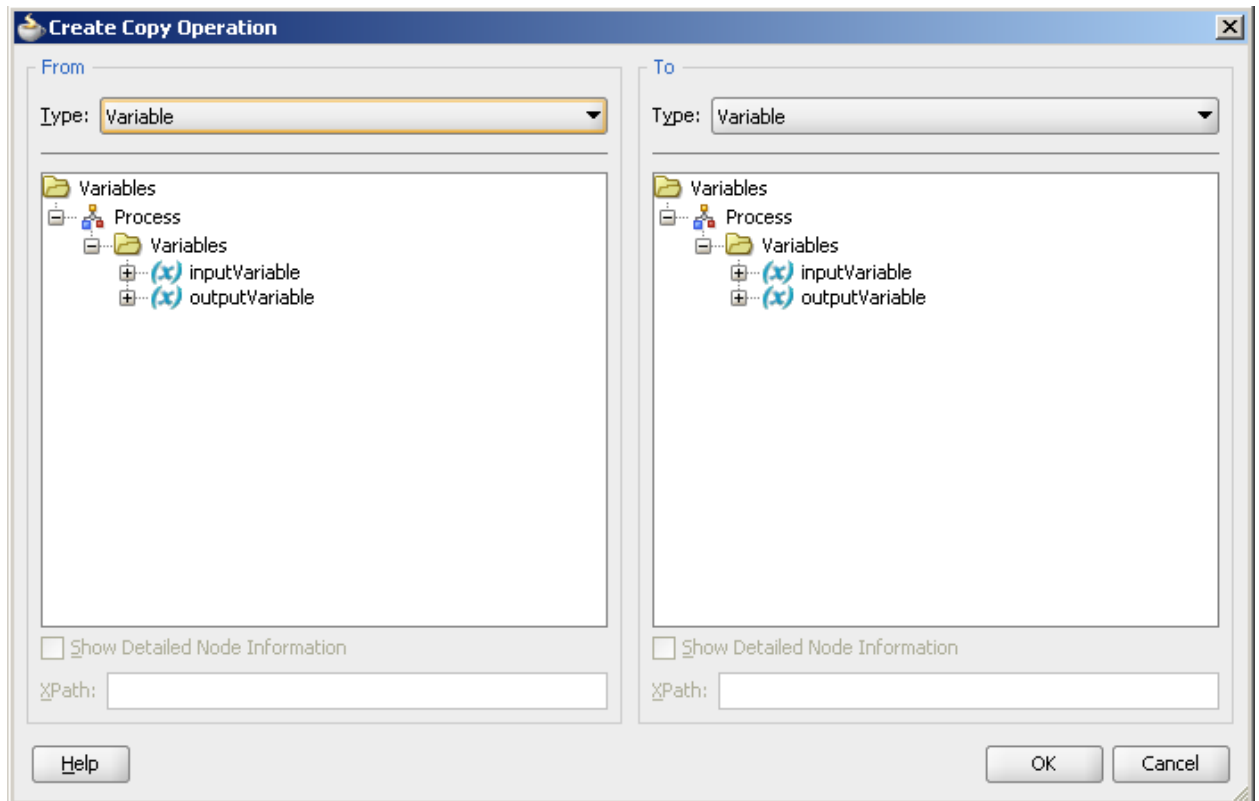


Double click the assign activity, to open the editor.

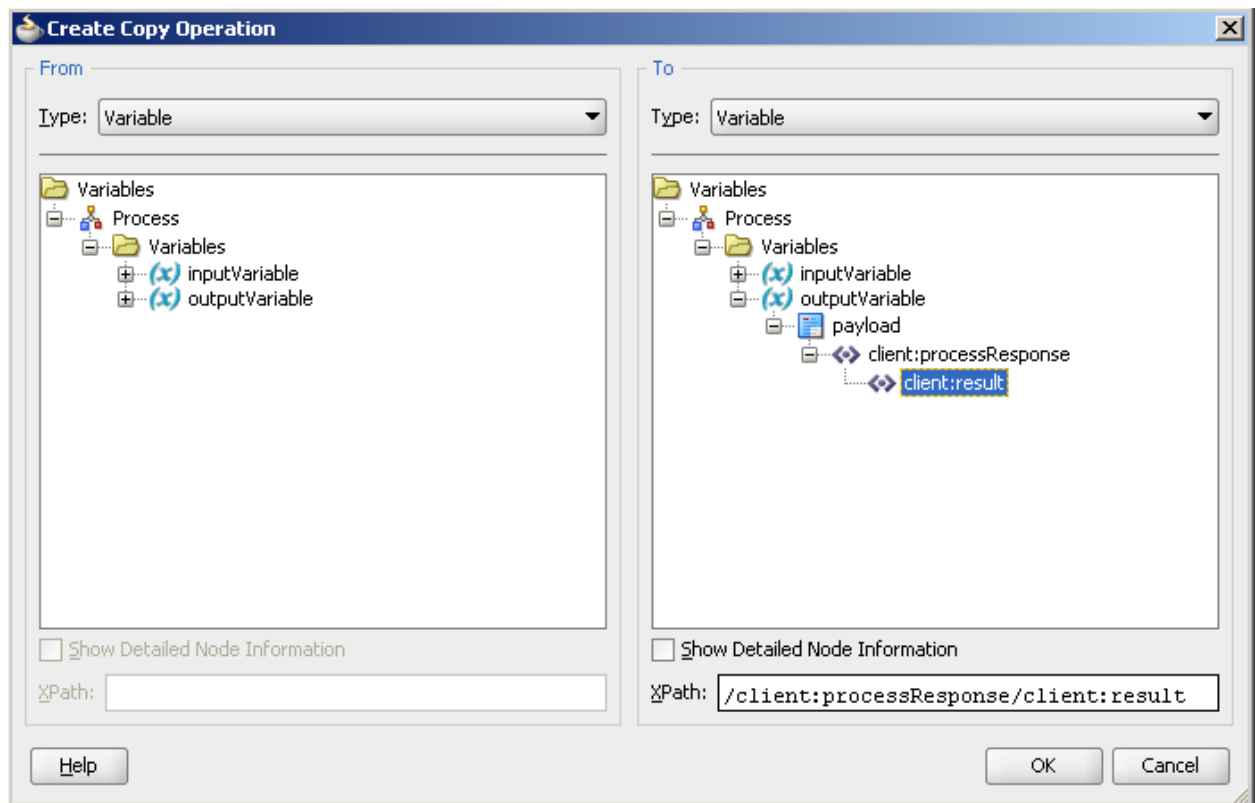


Select the second tab, labeled *Copy Operation* if it is not already selected. Click the green plus icon and select the *Copy Operation* from the drop down list.

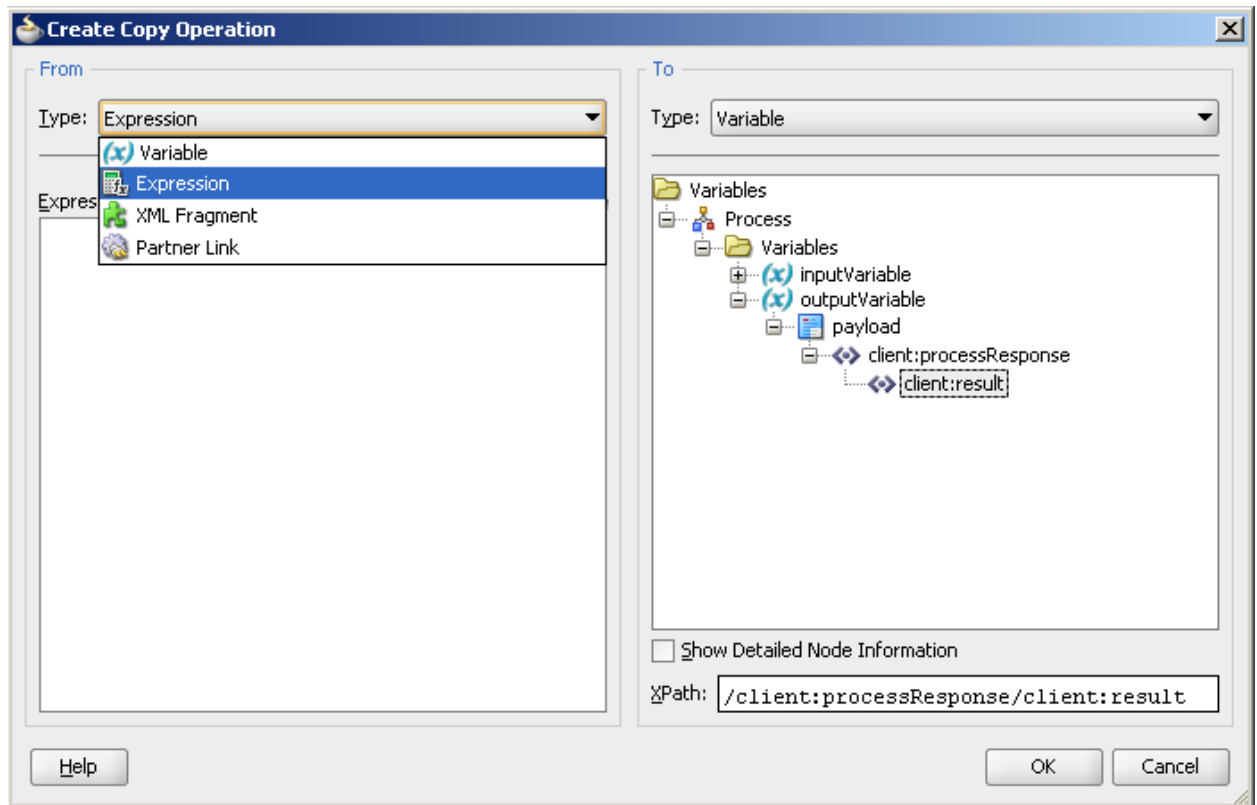




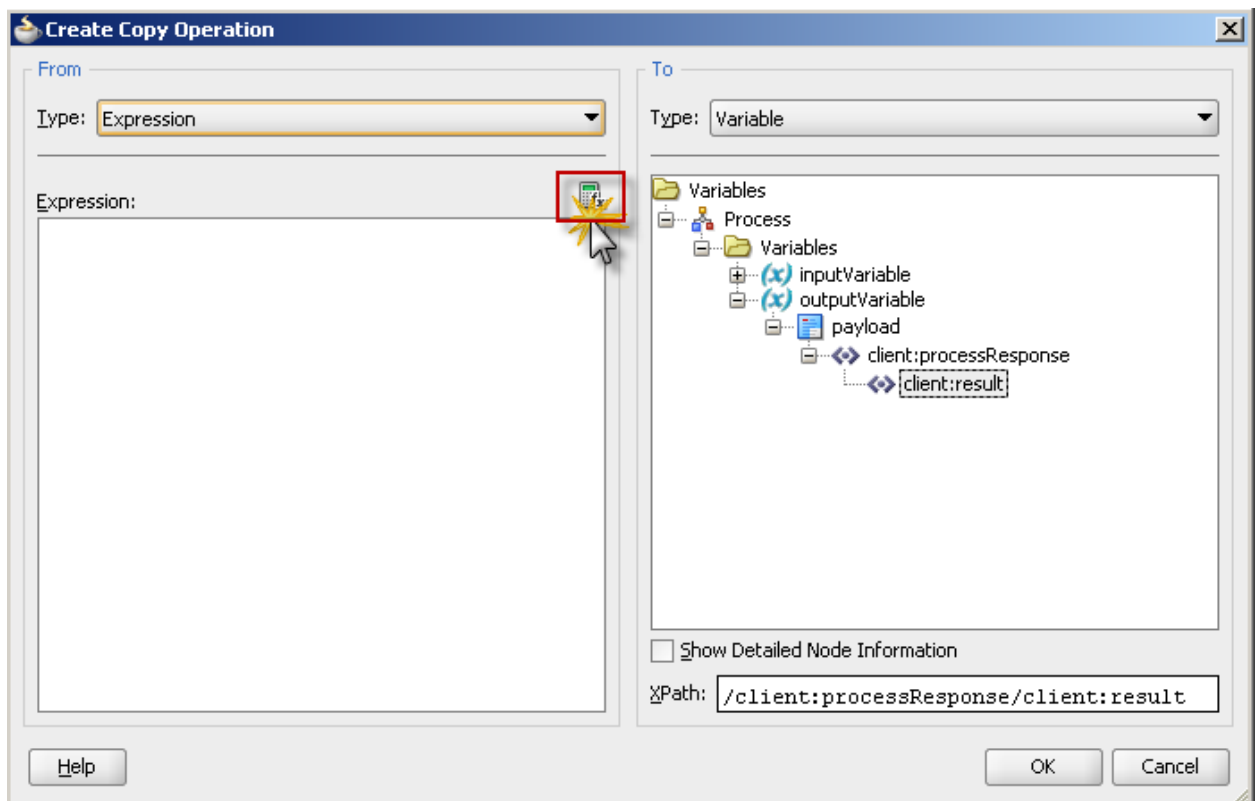
On the right (or To) side of the window, expand the outputVariable node, the payload child node and its client:processResponse node and select the client:result node. That is the target of the copy operation.



On the left (or From) side of the window, choose Expression in the dropdown list.

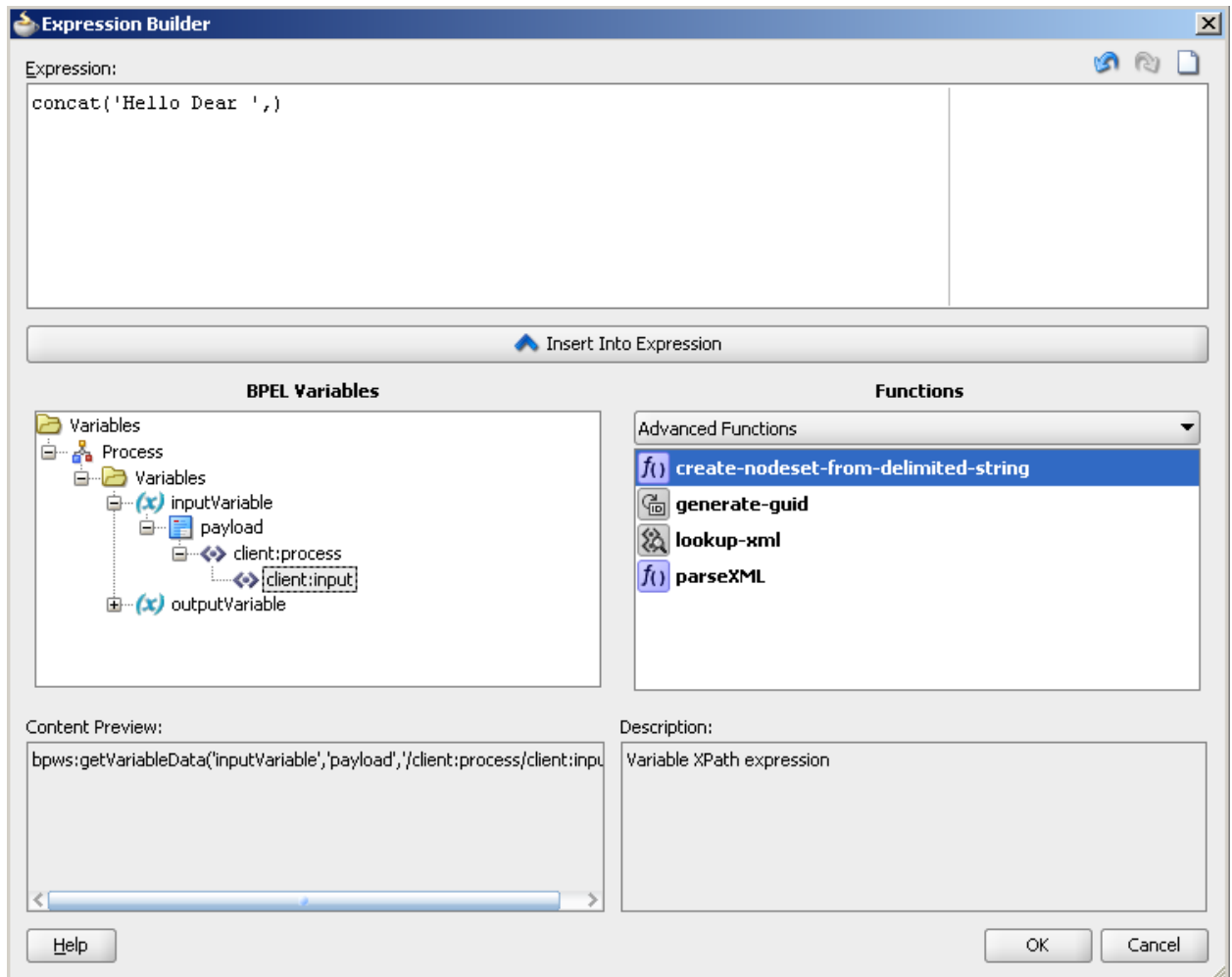


Click on the calculator icon, to open the XPath expression editor.



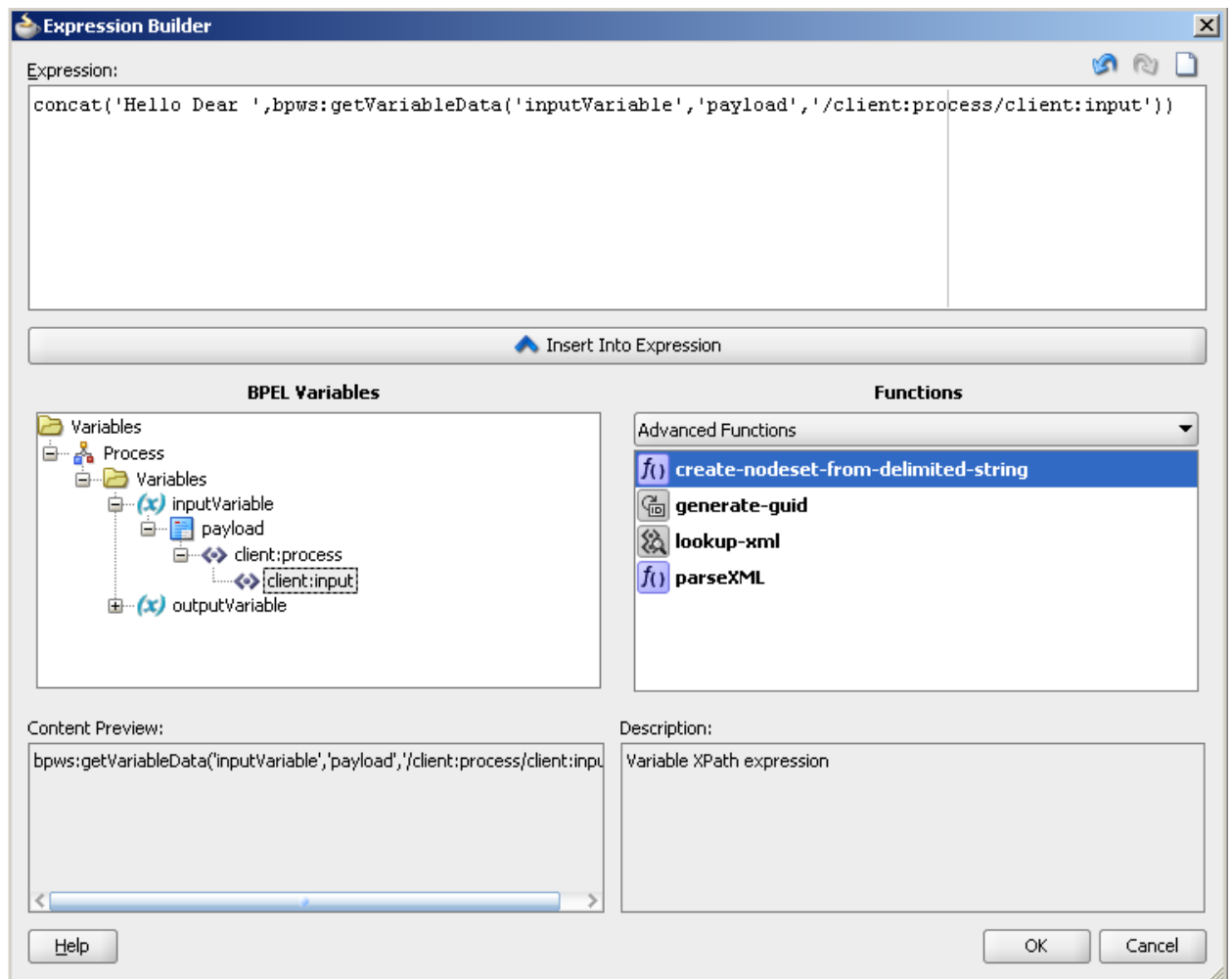
Type the following text in the expression box:

Concat('Hello dear',)

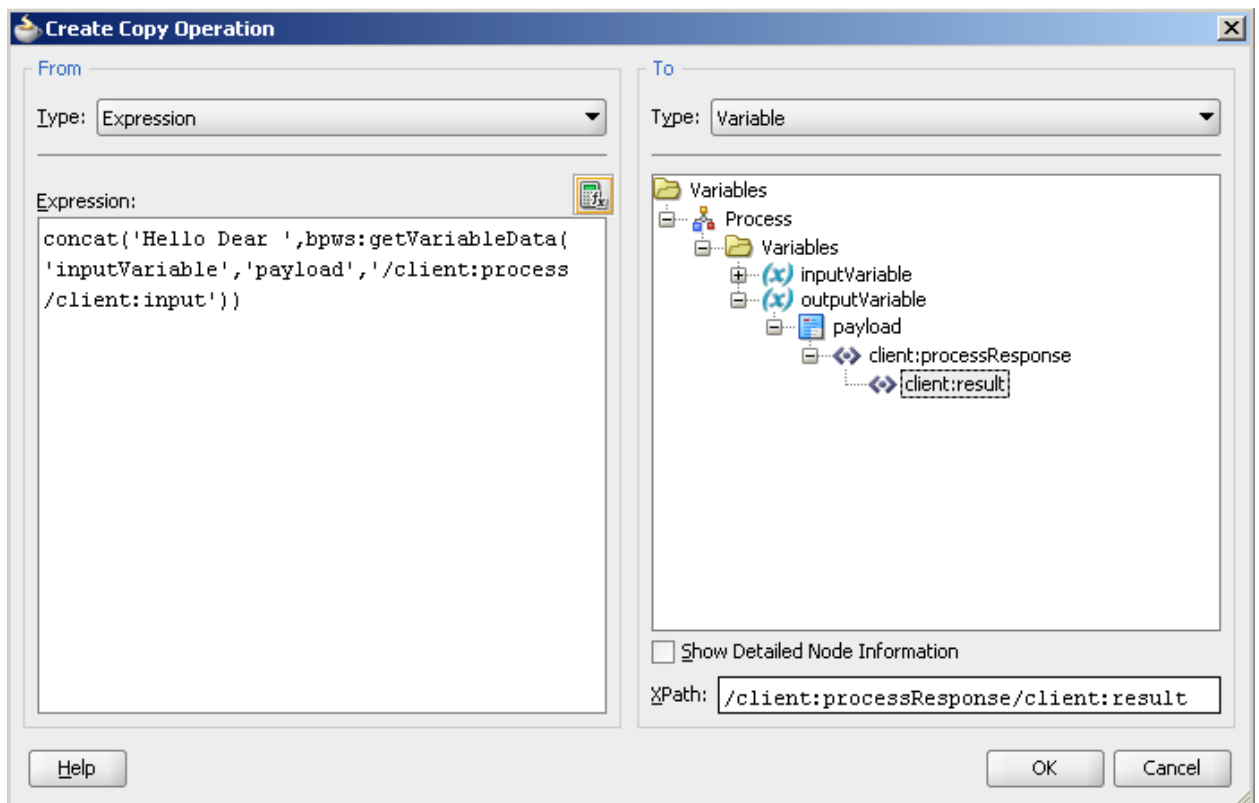


Position the cursor between the comma and the closing parenthesis. Expand the inputVariable in the BPEL Variables tree, all the way down until you can select the node client:input. Select that node.

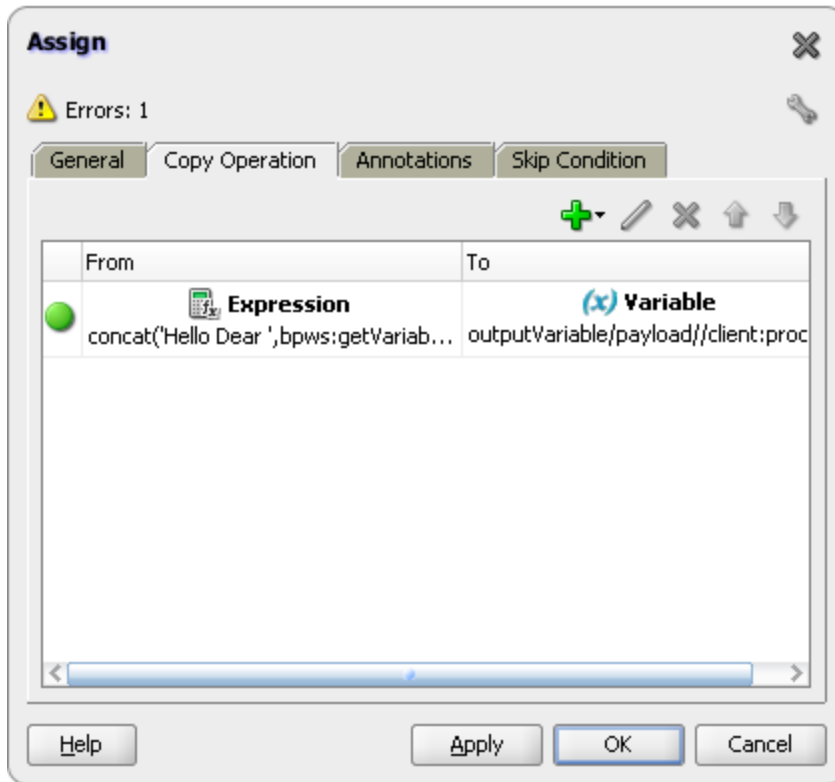
Click on the “Insert Into Expression” button to add [an expression to extract] the value of the input variable to the expression.



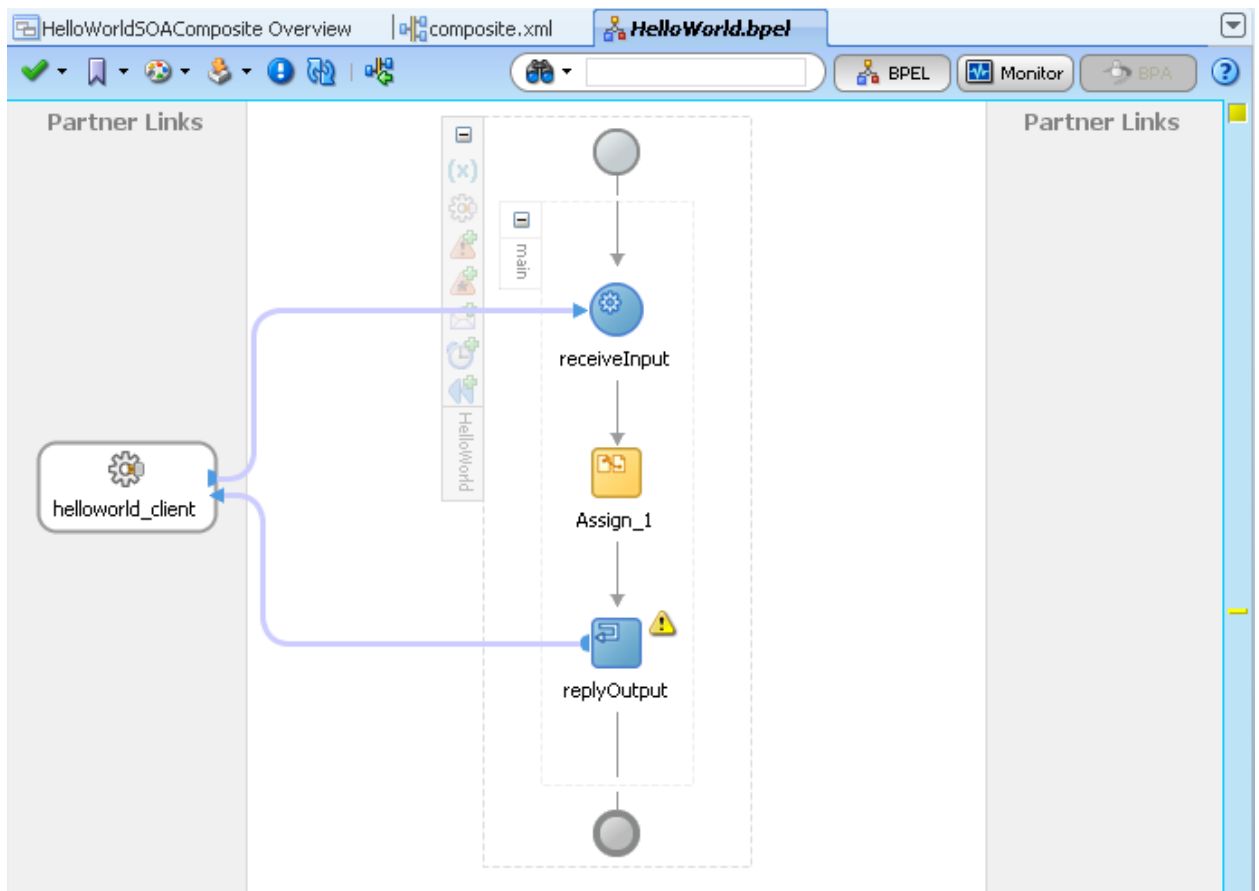
Click on the OK button to close the expression editor.



Click again on OK to close the Create Copy operation dialog



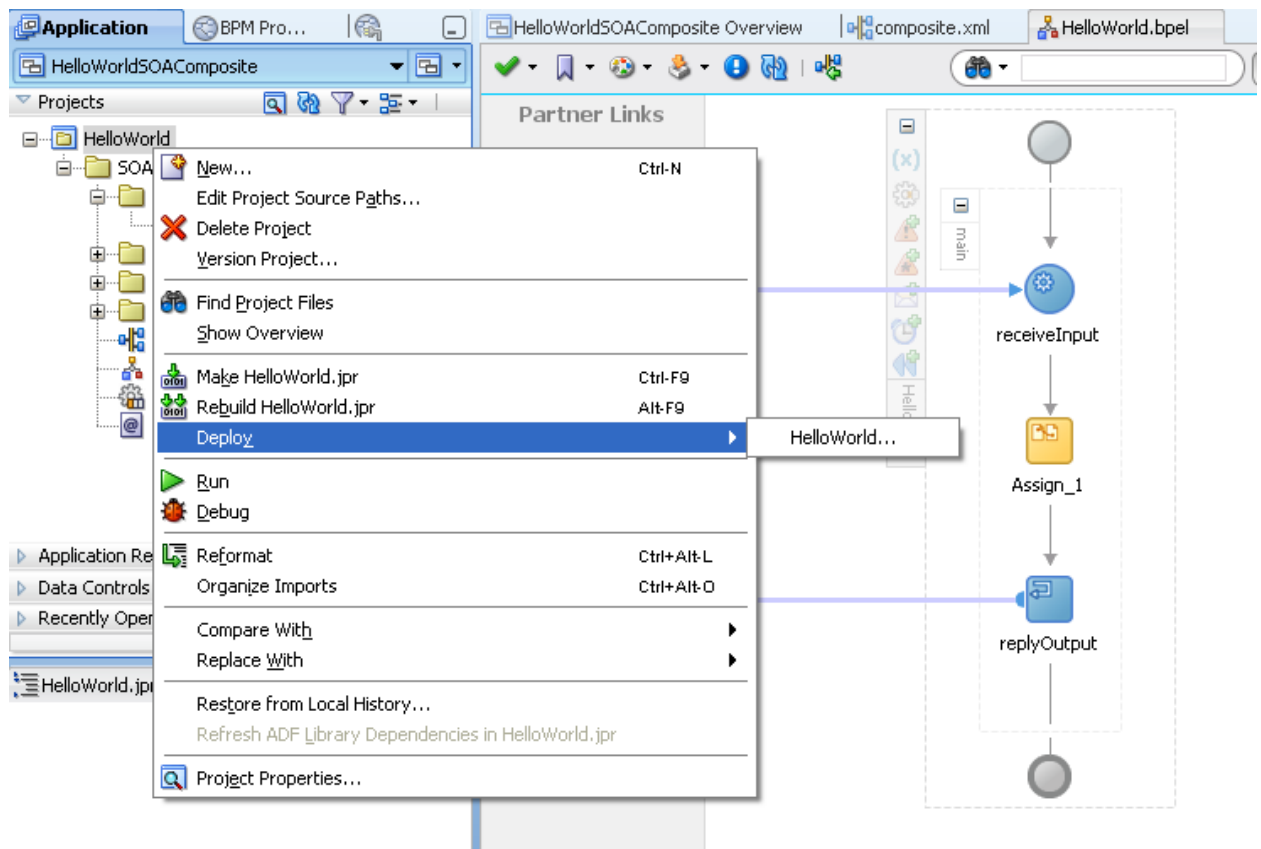
and then OK again to close the Assign editor.



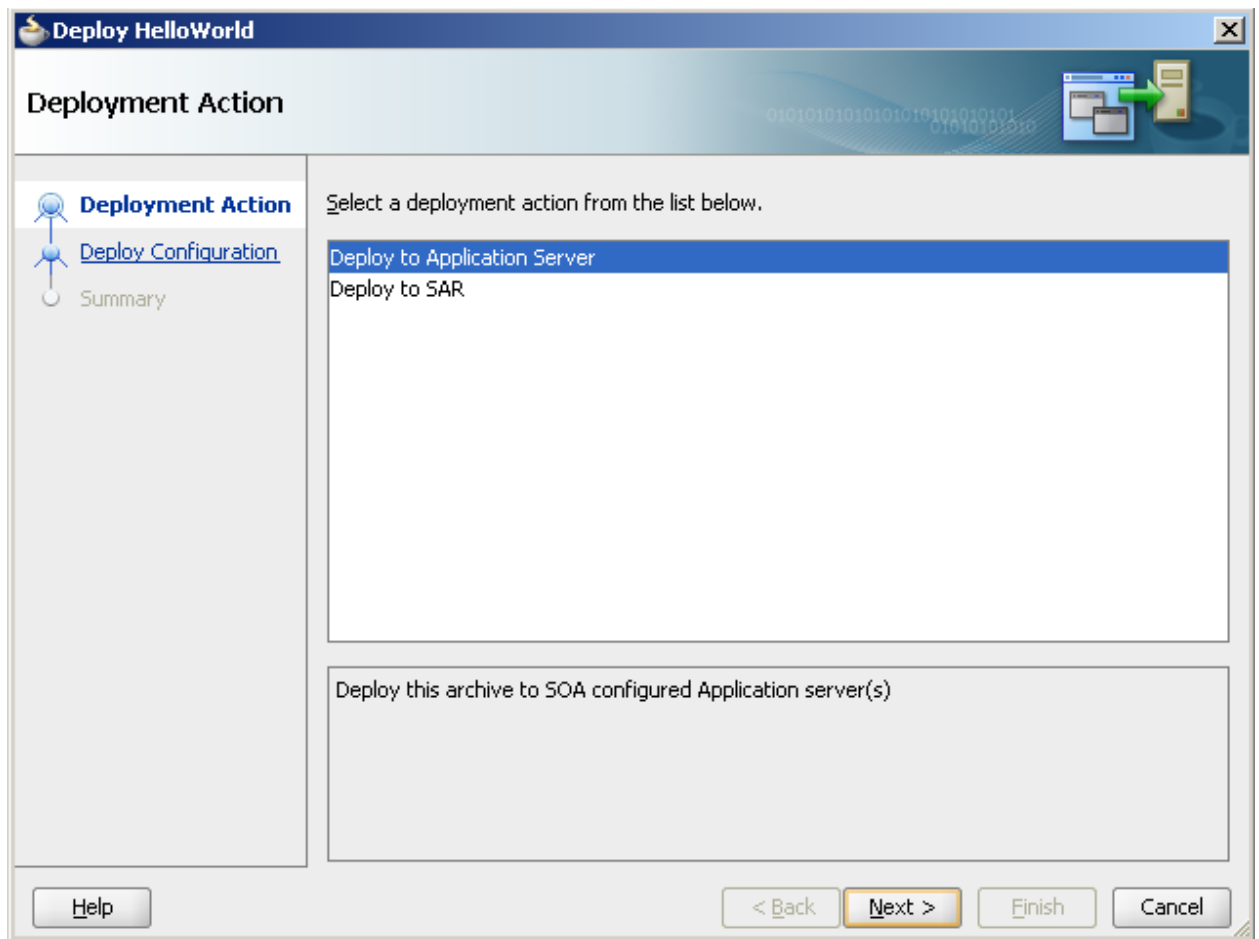
You have now created a valid BPEL process. One that receives a request message that contains a single string and returns a message that will contain the concatenation of “Hello dear” with that same input string. It’s not much, but it constitutes a real BPEL process inside the HelloWorldSOAComposite application.

6. To test run this application, we need to deploy it first. Right-click on the HelloWorld project. From the context menu, select “Deploy” and its nested option “HelloWorld”.

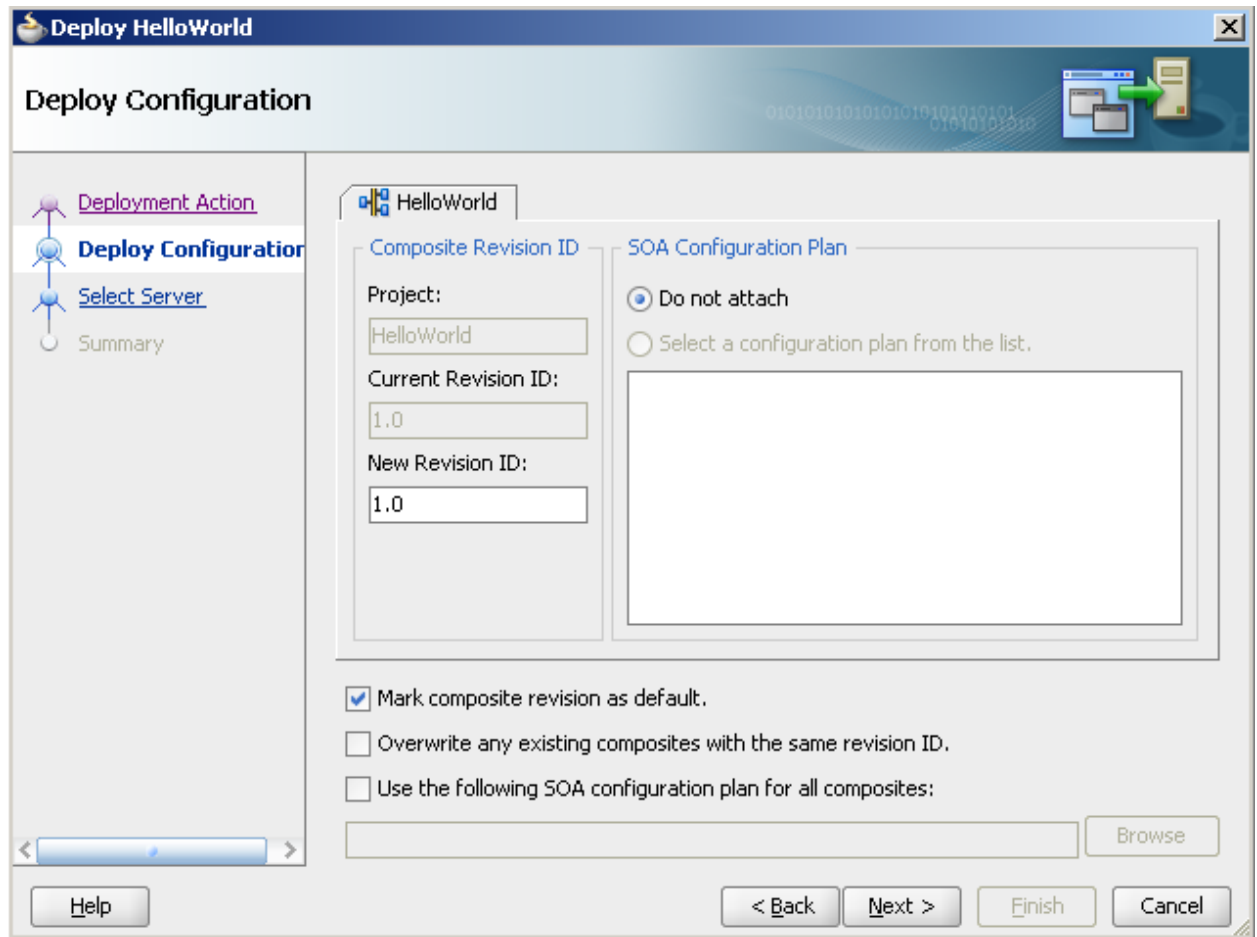




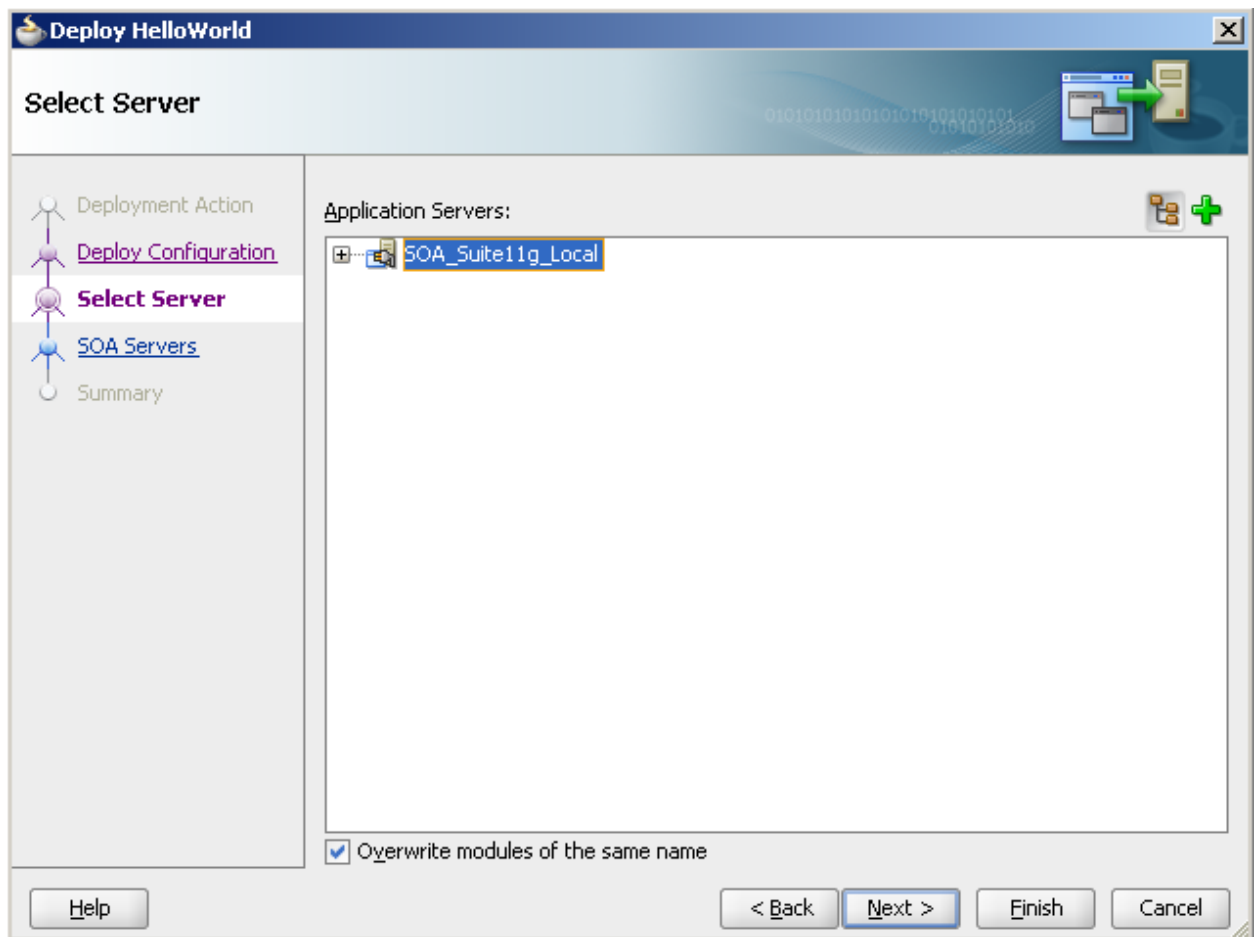
The Deployment wizard appears – a multi step dialog. In the first step, elect to *Deploy to Application Server* (instead of SAR file).



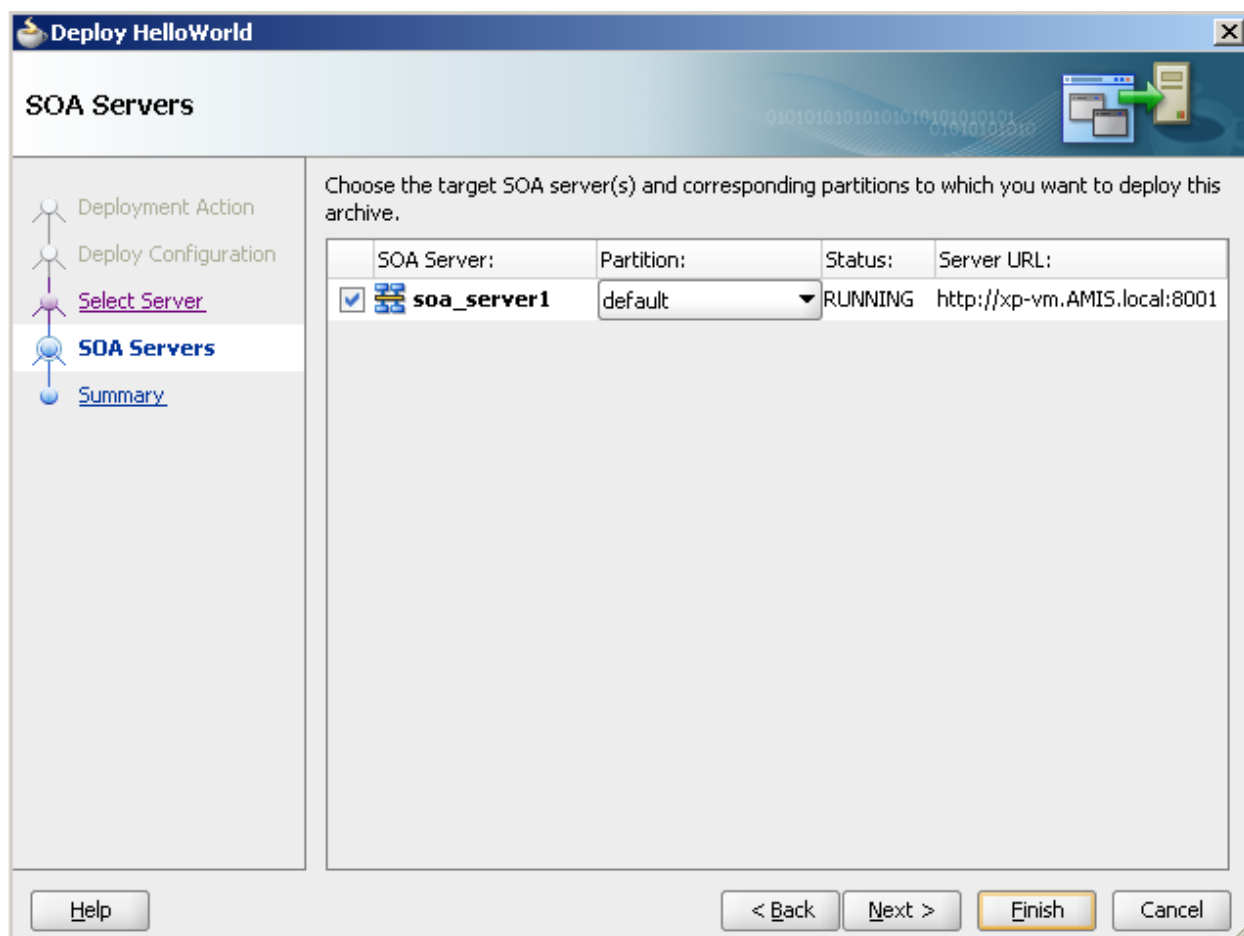
Click Next. Accept all default settings in the second step



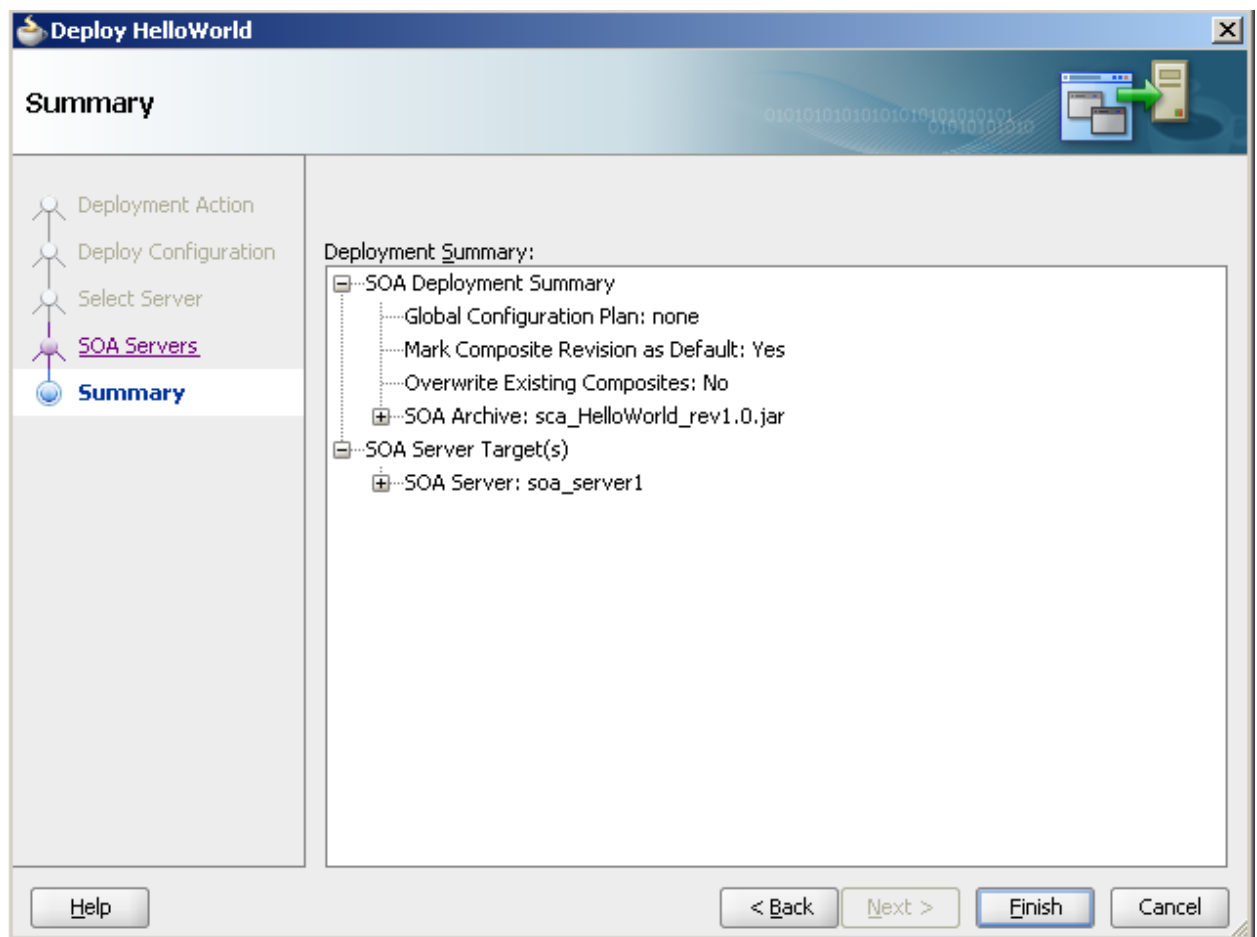
and press Next. On the third page: select the *SOASuite11g\_local* connection to the WebLogic Server with the *soa\_domain*.



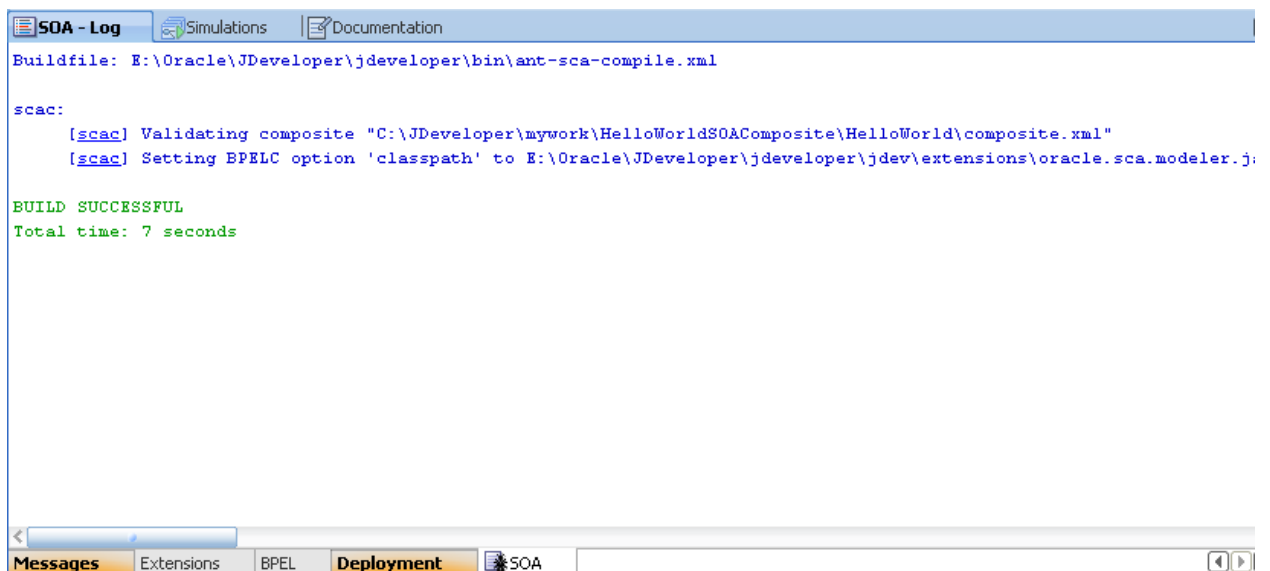
Press Next. On the next page, select *soa\_server1* as the target server for deployment.



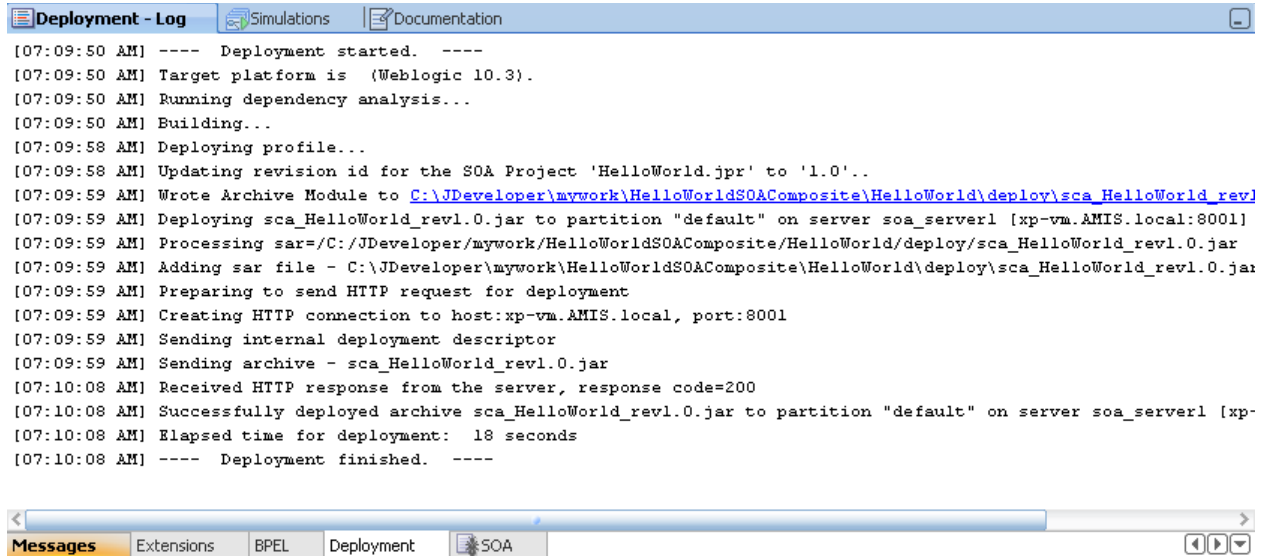
Press Next and the Summary page appears. Now you can press *Finish*.



The SOA Composite Application is built, resulting in a SAR (Service Archive) file.



Then it gets deployed to the application server: the SAR file is now handed to the soa\_server1 in the SOA Domain on the WebLogic Server. The message “Deployment Finished” should appear in the Deployment sub-tab of the console window after several seconds up to one minute.

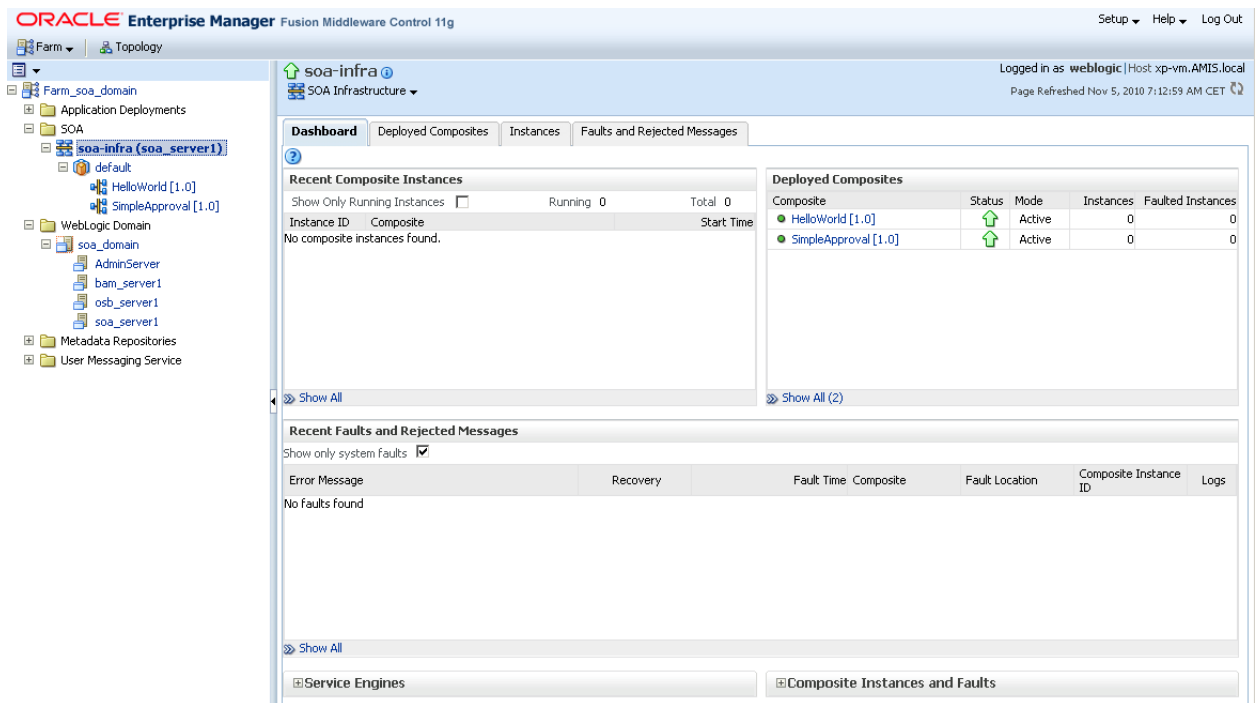


```

Deployment - Log
[07:09:50 AM] ---- Deployment started. ----
[07:09:50 AM] Target platform is {Weblogic 10.3}.
[07:09:50 AM] Running dependency analysis...
[07:09:50 AM] Building...
[07:09:58 AM] Deploying profile...
[07:09:58 AM] Updating revision id for the SOA Project 'HelloWorld.jpr' to '1.0'..
[07:09:59 AM] Wrote Archive Module to C:\JDeveloper\mywork\HelloWorldSOAComposite\HelloWorld\deploy\sca_HelloWorld_rev1
[07:09:59 AM] Deploying sca_HelloWorld_rev1.0.jar to partition "default" on server soa_server1 [xp-vm.AMIS.local:8001]
[07:09:59 AM] Processing sar=/C:/JDeveloper/mywork/HelloWorldSOAComposite/HelloWorld/deploy/sca_HelloWorld_rev1.0.jar
[07:09:59 AM] Adding sar file - C:\JDeveloper\mywork\HelloWorldSOAComposite\HelloWorld\deploy\sca_HelloWorld_rev1.0.jar
[07:09:59 AM] Preparing to send HTTP request for deployment
[07:09:59 AM] Creating HTTP connection to host:xp-vm.AMIS.local, port:8001
[07:09:59 AM] Sending internal deployment descriptor
[07:09:59 AM] Sending archive - sca_HelloWorld_rev1.0.jar
[07:10:08 AM] Received HTTP response from the server, response code=200
[07:10:08 AM] Successfully deployed archive sca_HelloWorld_rev1.0.jar to partition "default" on server soa_server1 [xp-
[07:10:08 AM] Elapsed time for deployment: 18 seconds
[07:10:08 AM] ---- Deployment finished. ----
  
```

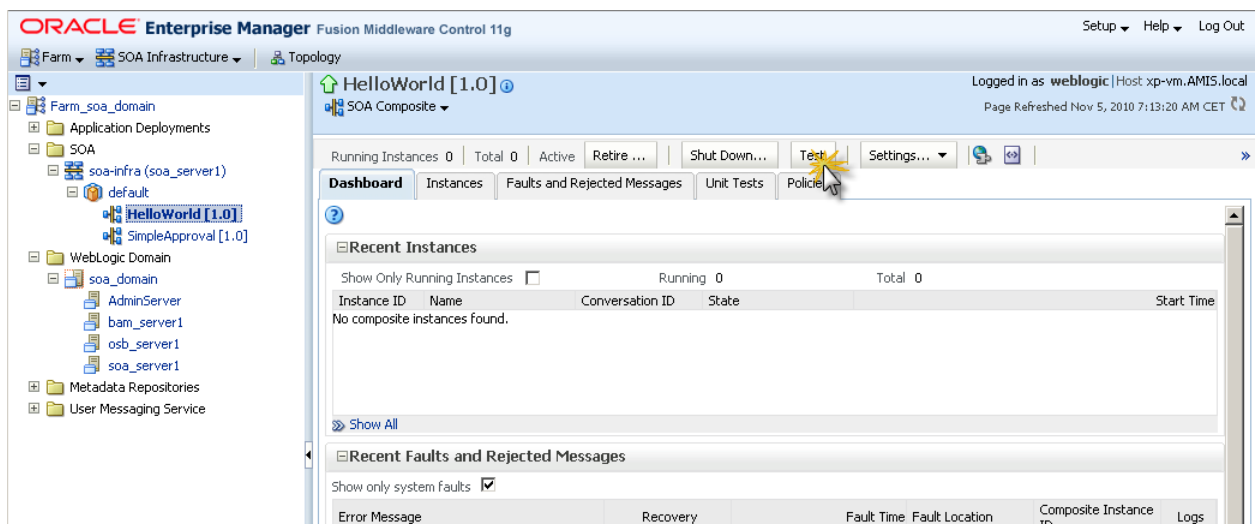
7. With the deployment done, we could access the composite application’s web service interface from tools such as the HttpAnalyzer inside JDeveloper or soapUI (an open source tool frequently used for testing web services). We can also open the Enterprise Manager to first inspect the deployed composite application and then test it.

Open the Enterprise Manager (<http://localhost:7001/em>).






Expand the node SOA node and its child, the soa-infra node, under the root node Farm\_soa\_domain. The node for the HelloWorld composite application should be listed under the *default* partition node. Select that composite application node.


The right side of the page is refreshed to present the details for this composite application.



Click on the Test button to call the service exposed by this composite application.




 HelloWorld [1.0]  Logged in as **weblogic** | Host: xp-vm.AMIS.local  
Page Refreshed Nov 5, 2010 7:14:59 AM CET 

 SOA Composite ▼

---

**Test Web Service** Test Web Service

Use this page to test any WSDL, including WSDLs that are not in the farm. To test a Web service, enter the WSDL and click Parse WSDL. When the page refreshes with the WSDL details, first select the Service, then select the Port, and then select the Operation that you want to test. Specify any input parameters, and click Test Web Service.

WSDL   Parse WSDL

[HTTP Basic Auth Option for WSDL Access](#)

Service: helloworld\_client\_ep  
Port: HelloWorld\_pt  
Operation: process ▼

Endpoint URL:  Edit Endpoint URL ☐

Request

Response

Security

☐ WSS Username Token
 ☐ HTTP Basic Auth
 ☐ Custom Policy
 ☒ None

Input Arguments


Tree View ▼

Name	Type	Value
* payload	payload	
* input	string	<input type="text" value="Lucas"/>

Request

Response

Test Web Service



Enter a value for the *input* field –for example your own first name - and press the button labeled Test WebService. The Web Service exposed by the HelloWorld application is invoked. This will create a new instance of the composite application. After a few seconds, the result from the service should be displayed, something to the effect of ‘Hello dear Lucas’.

Request

Response

Test Status: Passed

Response Time (ms): 6282

Tree View ▼

[Launch Message Flow Trace](#)

Name	Type	Value
* payload	payload	
result	string	<input type="text" value="Hello Dear Lucas"/>

When you click on the node for the HelloWorld application, you will see the new instance listed.

**HelloWorld [1.0]** SOA Composite Logged in as weblogic | Host xp-vm.AMIS.local Page Refreshed Nov 5, 2010 7:19:12 AM CET

Running Instances 0 | Total 1 | Active | Retire ... | Shut Down... | Test | Settings... |

**Dashboard** | Instances | Faults and Rejected Messages | Unit Tests | Policies

**Recent Instances**

Show Only Running Instances ☐ Running 0 Total 1

Instance ID	Name	Conversation ID	State	Start Time
?	---			Nov 5, 2010 7:18:29 AM

[Show All](#)

**Recent Faults and Rejected Messages**

**Component Metrics**

Name	Component Type	Total Instances	Running Instances	Faulted Instances	
				Recoverable	Non Recoverable
HelloWorld	BPEL	1	0	0	0

**Services and References**

Name	Type	Usage	Faults	Total Messages	Average Processing Time (sec)
helloworld_client_ep	Web Service	Service	0	1	3.031

You can drill down on this instance to find out more about the components in the instance that have executed, how long they took to complete for example, and all trace details for activity inside those components - such as the activities in the BPEL process. You need to click on the instance ID itself to see more information (not just on the row where the instance information is shown).

**Flow Trace** Data Refreshed Nov 5, 2010 7:19:41 AM CET

This page shows the flow of the message through various composite and component instances. ECID: 0000IkQfttM7U8YVLquHOA1BsFGH00001c Started Nov 5, 2010 7:18:29 AM

**Faults (0)**

**Faults**

Select a fault to locate it in the trace view.

Error Message	Recovery	Fault Time	Fault Location	Composite Instance
No faults found				

**Sensors (0)**

**Trace**

Click a component instance to see its detailed audit trail.

Show Instance IDs ☐

Instance	Type	Usage	State	Time	Composite Instance
helloworld_client_ep	Web Service	Service	✓ Completed	Nov 5, 2010 7:18:29 AM	HelloWorld of 1
HelloWorld	BPEL Component		✓ Completed	Nov 5, 2010 7:18:32 AM	HelloWorld of 1

The detailed audit trail for the BPEL process instance appears:

**Instance of HelloWorld** ⓘ  
This page shows BPEL process instance details. ⓘ

**Audit Trail** | Flow | Sensor Values | Faults

Expand a payload node to view the details.

- <process>
  - <scope name=main>
    - receiveInput
      - Nov 5, 2010 7:18:32 AM Received "inputVariable" call from partner "helloworld\_client"
        - <payload>
 

```
<inputVariable>
  <part name="payload">
    <ns1:process>
      <ns1:input>Lucas</ns1:input>
    </ns1:process>
  </part>
</inputVariable>
```
      - Assign\_1
        - Nov 5, 2010 7:18:32 AM Updated variable "outputVariable"
        - Nov 5, 2010 7:18:32 AM Completed assign
      - replyOutput
        - Nov 5, 2010 7:18:32 AM Reply to partner "helloworld\_client".
          - <payload>
 

```
<outputVariable>
  <part name="payload">
    <processResponse>
      <result>Hello Dear Lucas</result>
    </processResponse>
  </part>
</outputVariable>
```

Nov 5, 2010 7:18:32 AM BPEL process instance "1" completed