Designing, deploying and monitoring BPEL processes with Spagic3

Author Daniela Butano

1 Purpose

The purpose of this document is to provide an introduction on using the Spagic3 platform: it does so with walkthrough of the design, deployment and the monitoring stages of a sample BPEL process.
For this purpose, Spagic Studio integrates on the Eclipse platform, the BPEL Designer to model the processes and Apache ODE runtime to execute them.
Installation and configuration of the BPEL Designer in Spagic Studio and ODE are outside the scope of this document.

2 Versions History

Version/Release n° :	1.0	**Date**	23/09/2009
Description	First release (English version)		
Version/Release n° :	2.0	**Date**	10/05/2010
Description	Updates for Spagic 3.0.0		

3 Introduction

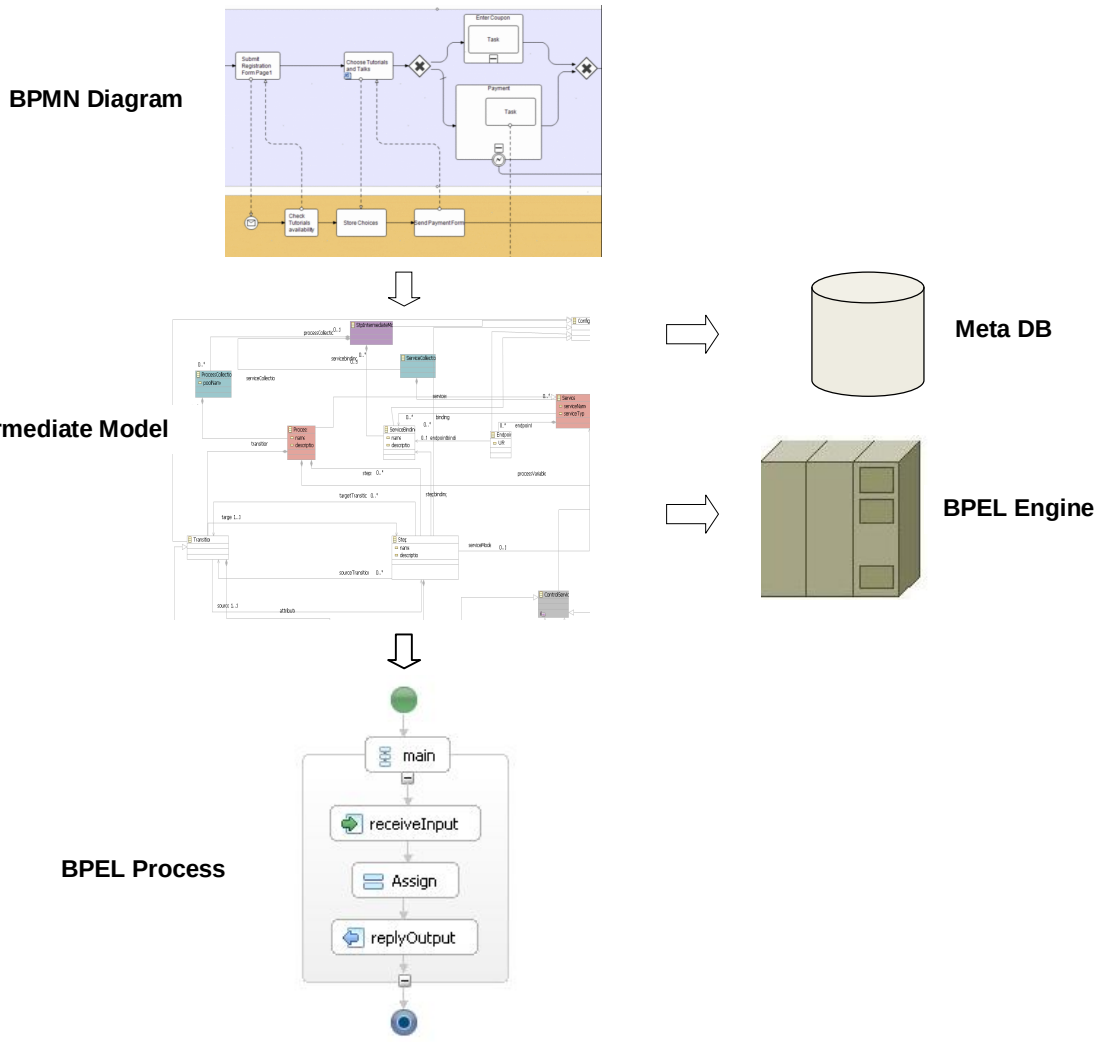
Spagic3 platform allows designing, executing and monitoring BPEL processes.

Spagic 3 allows creating BPEL processes either from BPMN diagrams or, new to this version, by using the integrated BPEL designer.

In fact, in **previous versions**, to design a BPEL process in order you had to:

- model a BPMN diagram representing the process, using the BPMN editor integrated in the platform,
- generate the intermediate model by the BPMN diagram,
- generate the BPEL process by the intermediate model,
- complete the BPEL process with technical configurations and data, using th BPEL designer editor integrated in the platform,
- deploy the intermediate model into meta database,
- generate the service assembly and publish it in servicemix, where it is executed and monitored.

The process executions are displayed through the console. The following figure shows the flow:



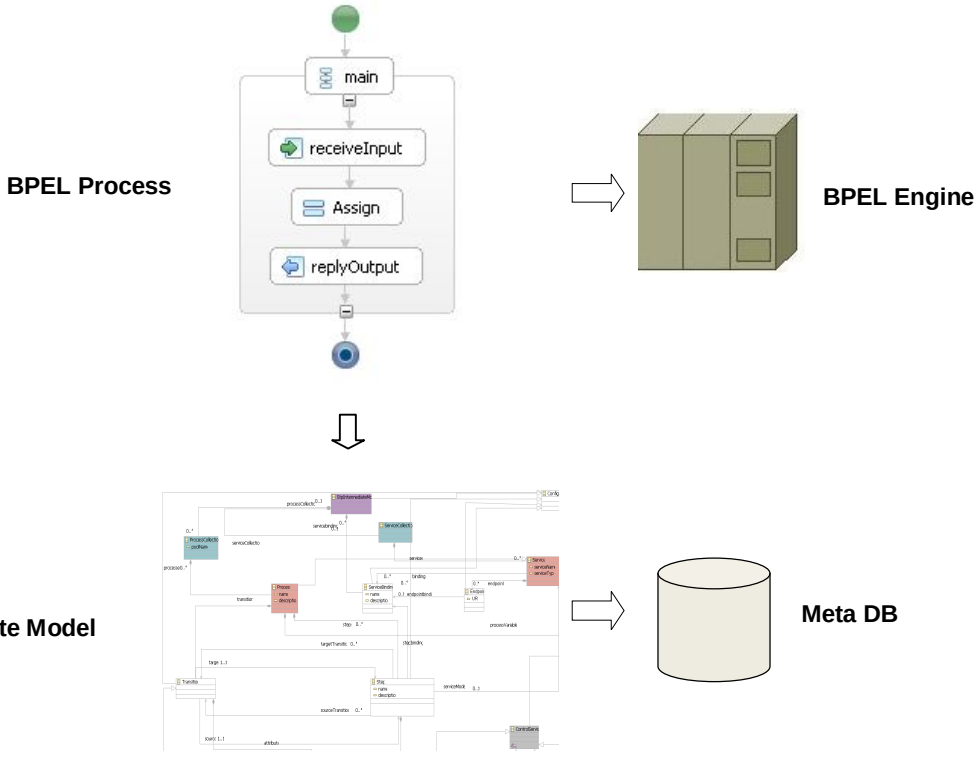
In **Spagic3**, to design a BPEL process you can:

- model the BPEL process using the BPEL designer editor,
- deploy the process into the Apache ODE, the BPEL engine, where it is executed and monitored.

The intermediate model generation is transparent to the user.

The process executions are displayed through the console.

The following figure shows the flow:



This document is organized as follows:

- A section that shows how to design a process and also a how to configure the port, the binding and the deployment descriptor associated;
- A section that explains how to execute it;
- A section that introduces the console to monitor the process execution;
- A section with some examples associated with the main BPEL constructs.

For more details about Spagic Console, see the document *Spagic Studio User Guide*.doc

For more details about Spagic Studio environment see the document *Spagic Studio User Guide.doc.*

4 Requirements

Required Tools	URL for download/Notes	Required for Spagic Studio	Required for Spagic Console
Database	MySQL Oracle H2	✓	✓
Apache ODE 2.0 Beta	http://ode.apache.org/		✓
GraphViz	http://www.graphviz.org/	✓	
JDK 1.6.X or later	http://java.sun.com/	✓	✓
Apache Tomcat 6.0.x	http://tomcat.apache.org		✓
Mozilla Firefox 3.x	http://www.mozilla.com		✓

5 Developing a Process

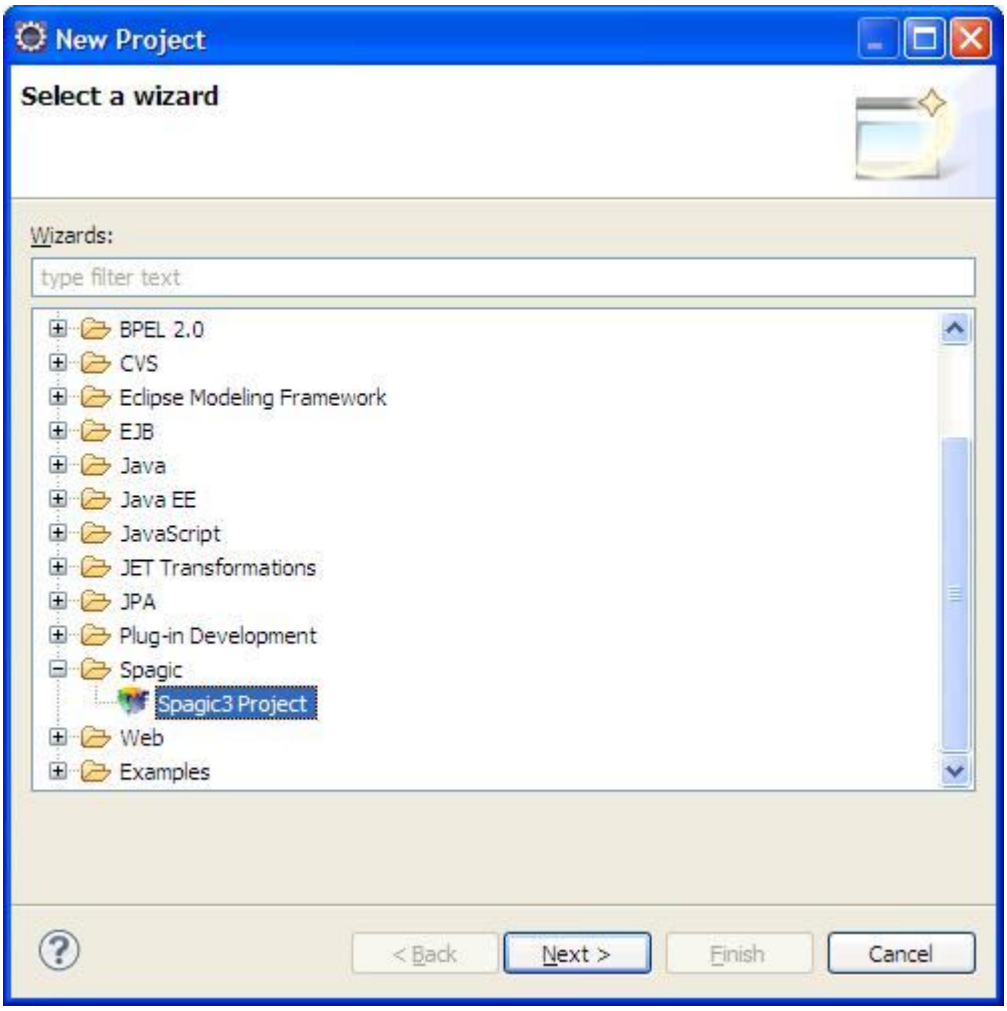
Spagic Studio is an Eclipse environment that allows the use of a single interface in order to create components and processes managing the entire development cycle. For the details, see the documents *Spagic Studio User Guide*.
Spagic Studio integrates the Bpel Designer to design BPEL processes.
In this section we'll shortly describe all steps to develop a BPEL process using a simple example.

5.1 Create a Spagic3 project

Open Spagic Studio and create a workspace for working, for example call it *SpagicDemo*.
As first step, configure the Spagic General Preferences from *Window->Preferences-> Spagic General Preferences*; set the attributes for:

- the graphviz application in the main page, section *Graph Viz*;
- the ODE urls in *Spagic BPEL* tab;
- the metaDB in *Spagic Meta Database* tab.

To create a new project, select *File->New->Project->Spagic->Spagic3 Project*, insert a Project Name, *StartSpagic*, click *Finish*.

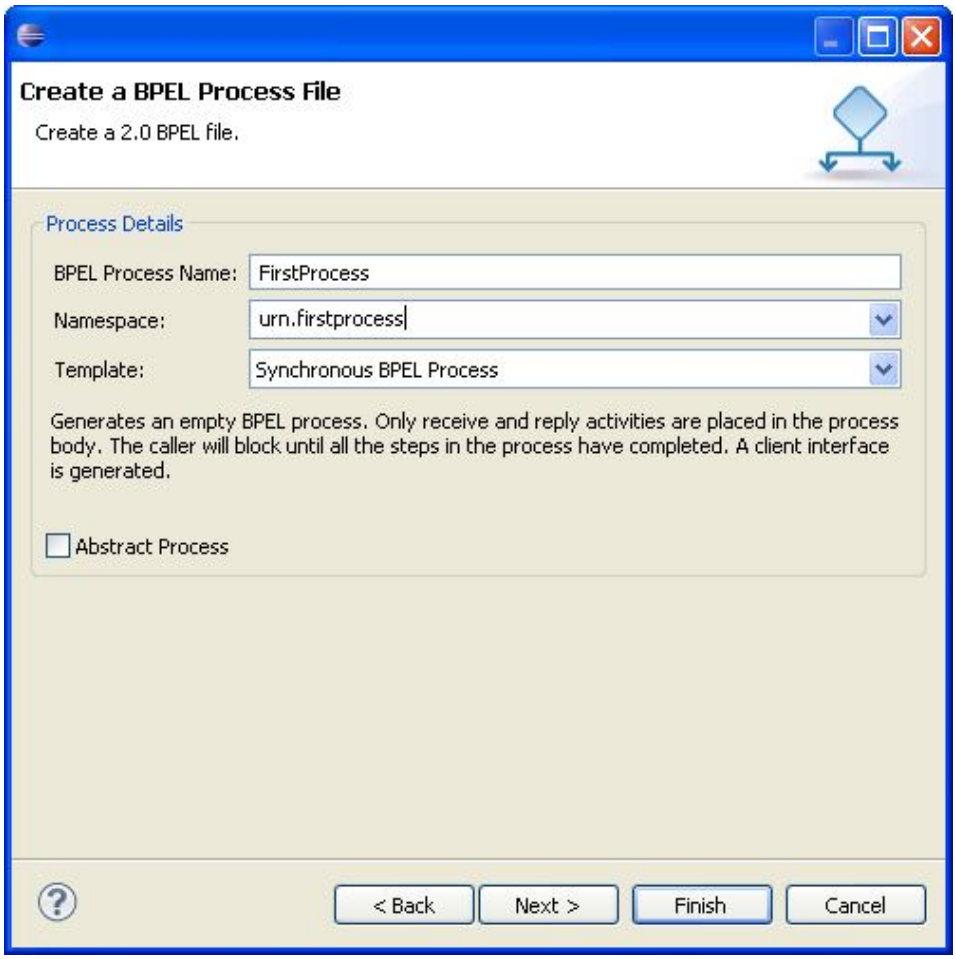


In the workspace, the project just created has a standard structure containing the followings directories:

Connectors, *Processes*, *Resources*, *Services.*

5.2 Design the process

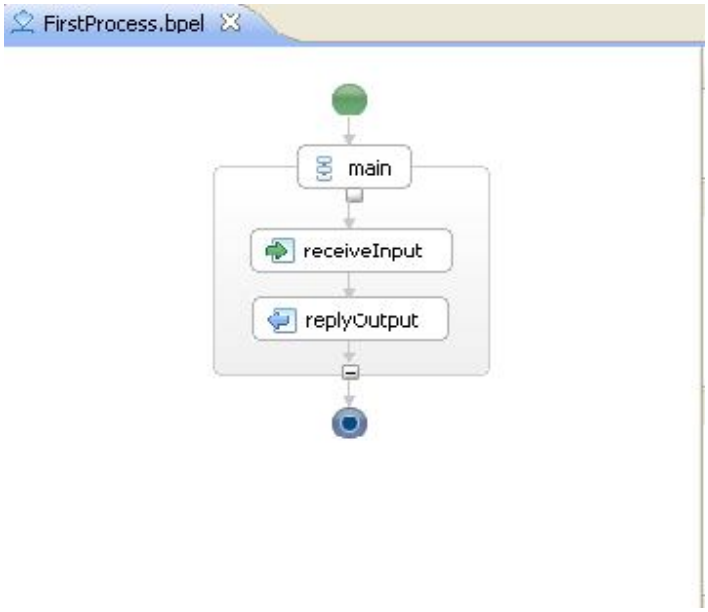
To define a new process, select the *Processes* folder, open the context menu and *New->Other->BPEL2.0->New BPEL Process File,* click *Next.* Insert the *BPEL Process Name*, for example *FirstProcess*, the *Namespace*, for example *urn.firstprocess* and select the template *Synchronous BPEL Process* as shown in the following screenshot.



Click *Next*, select *SpagicDemo* folder, then *Processes* folder, click *Finish*.

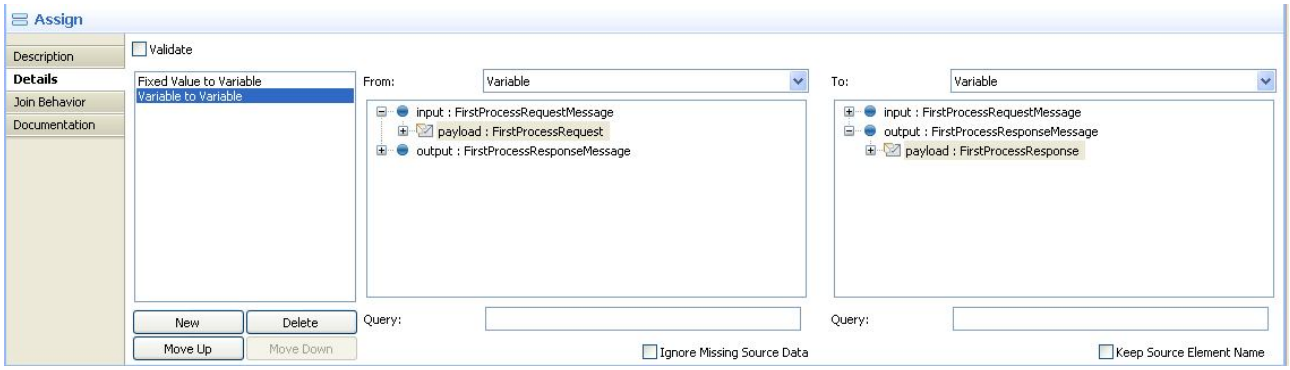
Spagic Studio will show in the BPEL Designer editor the process just created.

The result should be similar to the following screenshot.



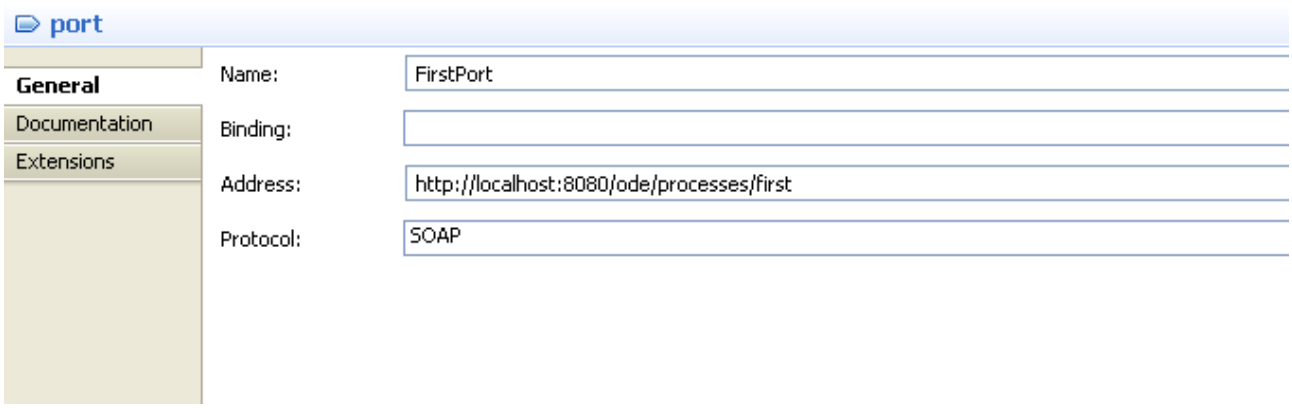
To complete the process, select, from the palette located at the right of the editor, under the category *Actions*, an *Assign* activity and put it to the editing area, between *receiveInput* and *replyOutput* .

Select the *Assign* activity in the editor area, right-click, select *Show in Properties*. In the *Details* page, we create an assignment, selecting *New*, then select *From* Variable and specify the payload for the *input.* Select *To* Variable and specify the payload for the *output*. Click *Yes*, when the dialog relative the Initialization variables is displayed. Save the process.



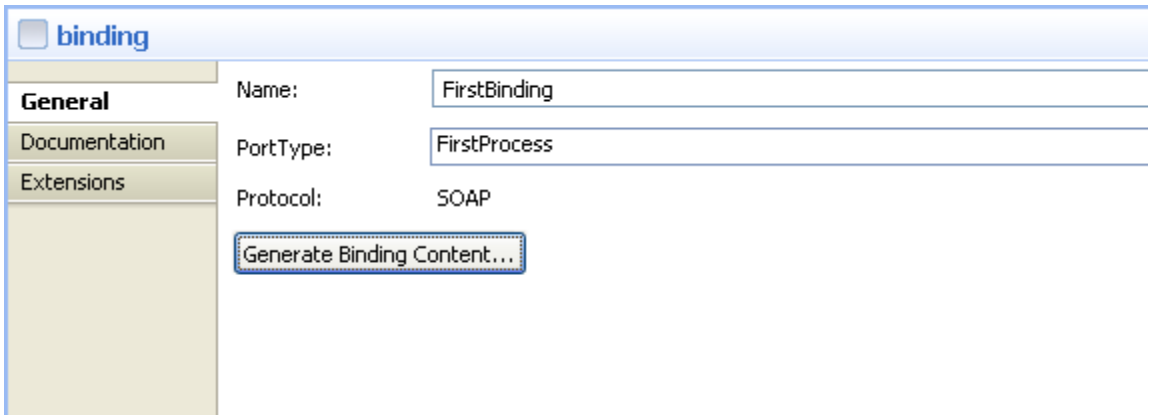
5.3 Configure Port and Binding

In order to execute the process, we have to define a Port and a Binding. Double-click on the file *FirtProcesArtifact.wsdl* to open the Wsdl editor. Select the editor area, open the context menu, and select *Add Service.* Configure the name as *FirstService*. Rename the *NewPort* in *FirstPort* and configure this as shown in the following screenshot.

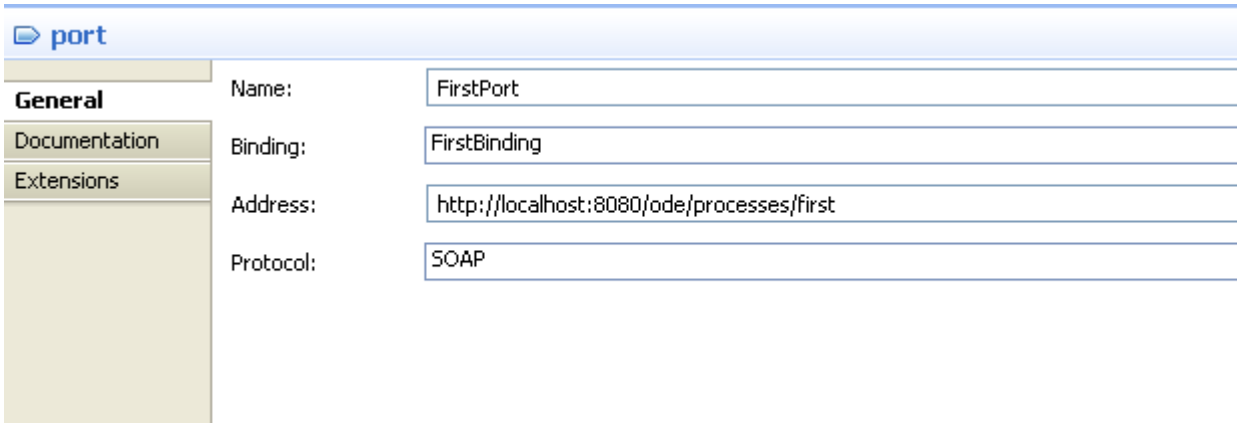


Save the wsdl file.

Select the editor area, open the context menu, and select *Add Binding.* Configure the name as *FirstBinding*, in *Port Type* select *FirstProcess*, click *Generate Binding Content*, in the dialog select the *Protocol* SOAP, and click *Finish*.

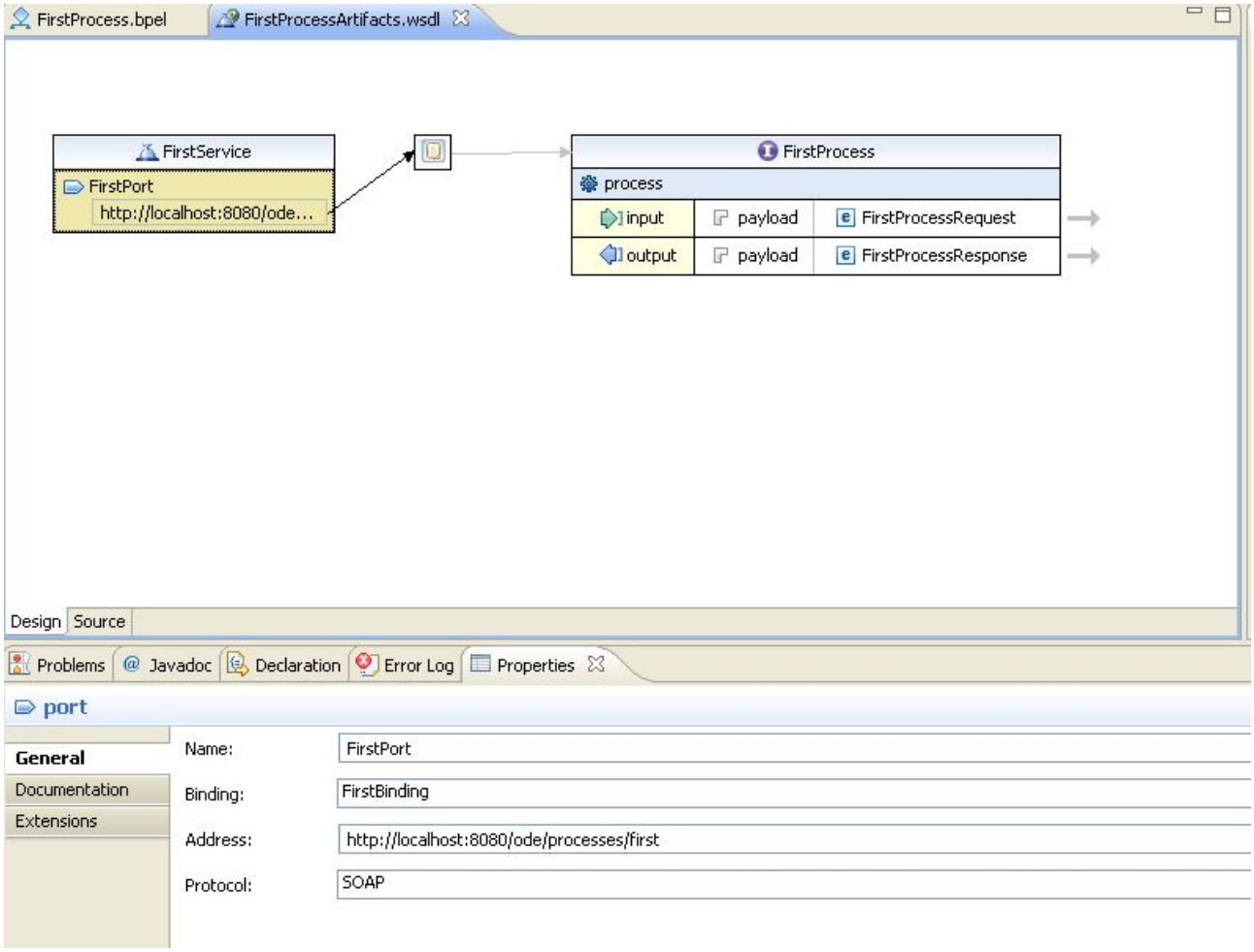


In the editor area, select the *FirstPort* and configure his *Binding* selecting *FirstBinding.*



Save the wsdl file.

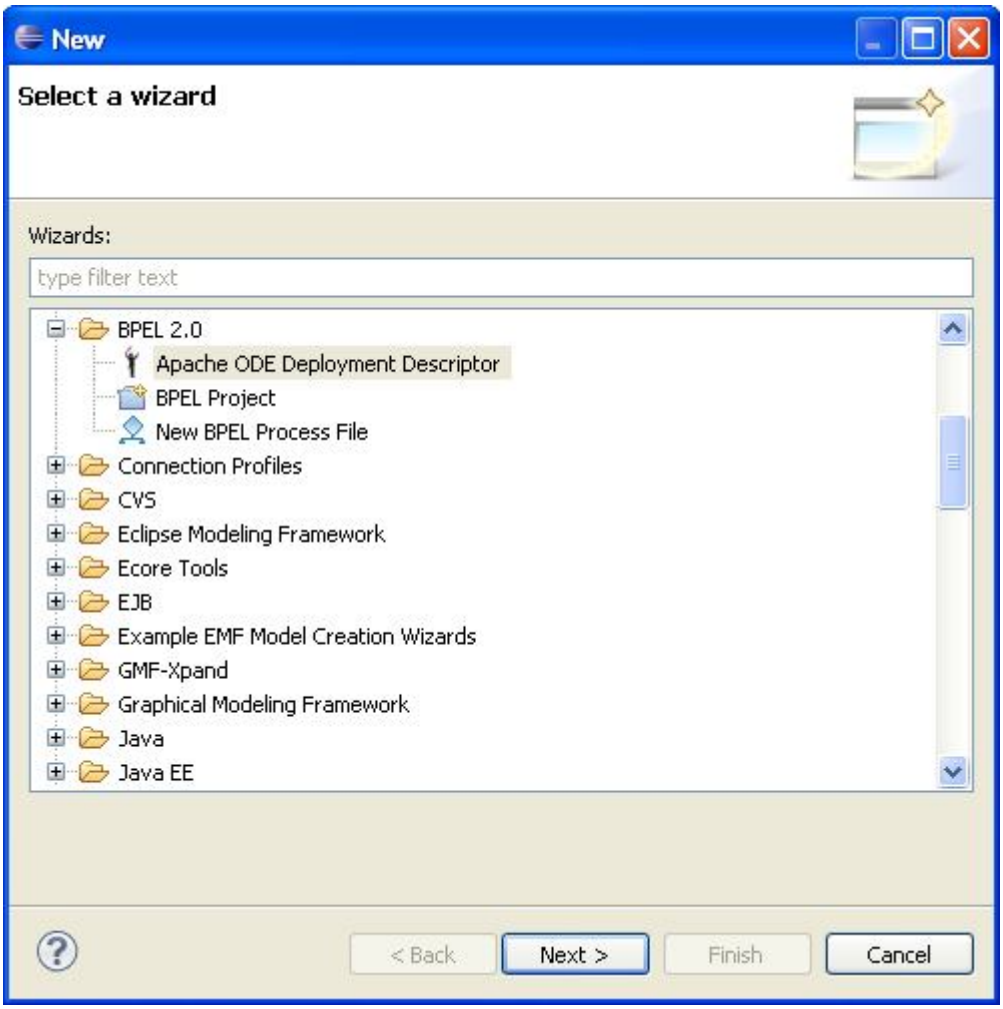
The result should be similar to the following screenshot.



5.4 Configure the deployment descriptor

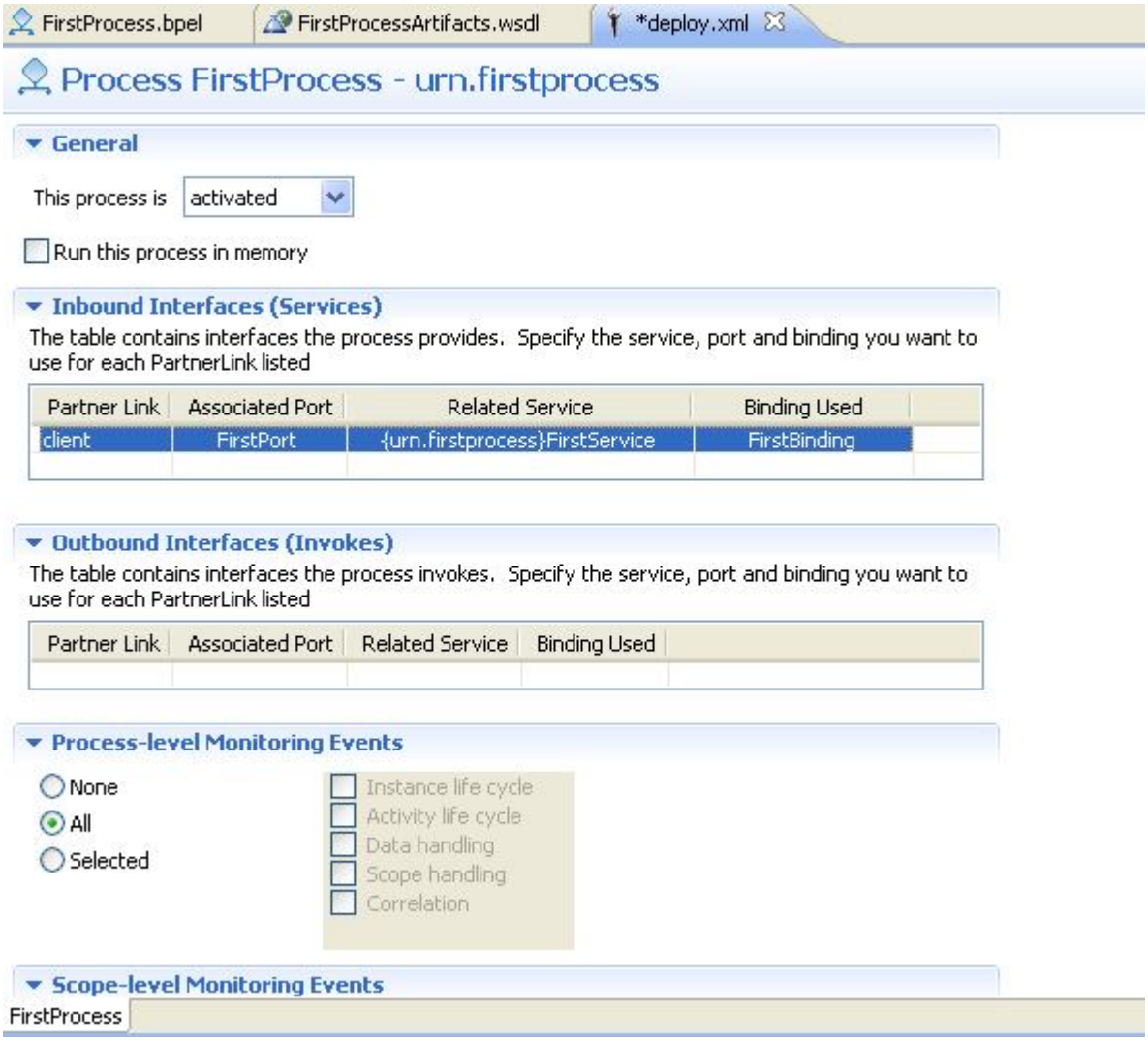
In order to deploy the process, we have to generate and configure an ODE deployment descriptor file.

Select the *Processes* folder, open the context menu, and select *New -> Other -> BPEL2.0 -> Apache ODE Deployment Descriptor*.



Click *Next*, check that the BPEL Project has been set to */SpagicDemo/Processes*. Click *Finish.*

In the *Processes* folder, the *deploy.xml* file has just been generated. Right-click on *deploy.xml* file, select *Open With -> ODE Deployment Descriptor Editor*. Associate the port *FirstPort* to *client* partner link, as shown in the following screenshot. Save the file.



5.5 Publish the process

After designing the process, you have to publish it: this operation execute the following steps:

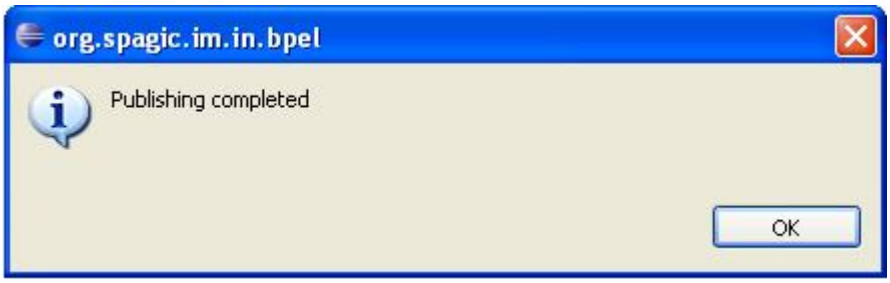
- generates the Intermediate Model associated to BPEL process,
- creates a .zip containing *.bpel*, *.wsdl*, *deploy.xml*,
- deploys the process into Apache ODE,
- retrieves the version process from ODE and set it into the Intermediate Model,
- saves the process information (properties, flow...) into the MetaDB.

Before the publishing, assure that Apache ODE 's running.

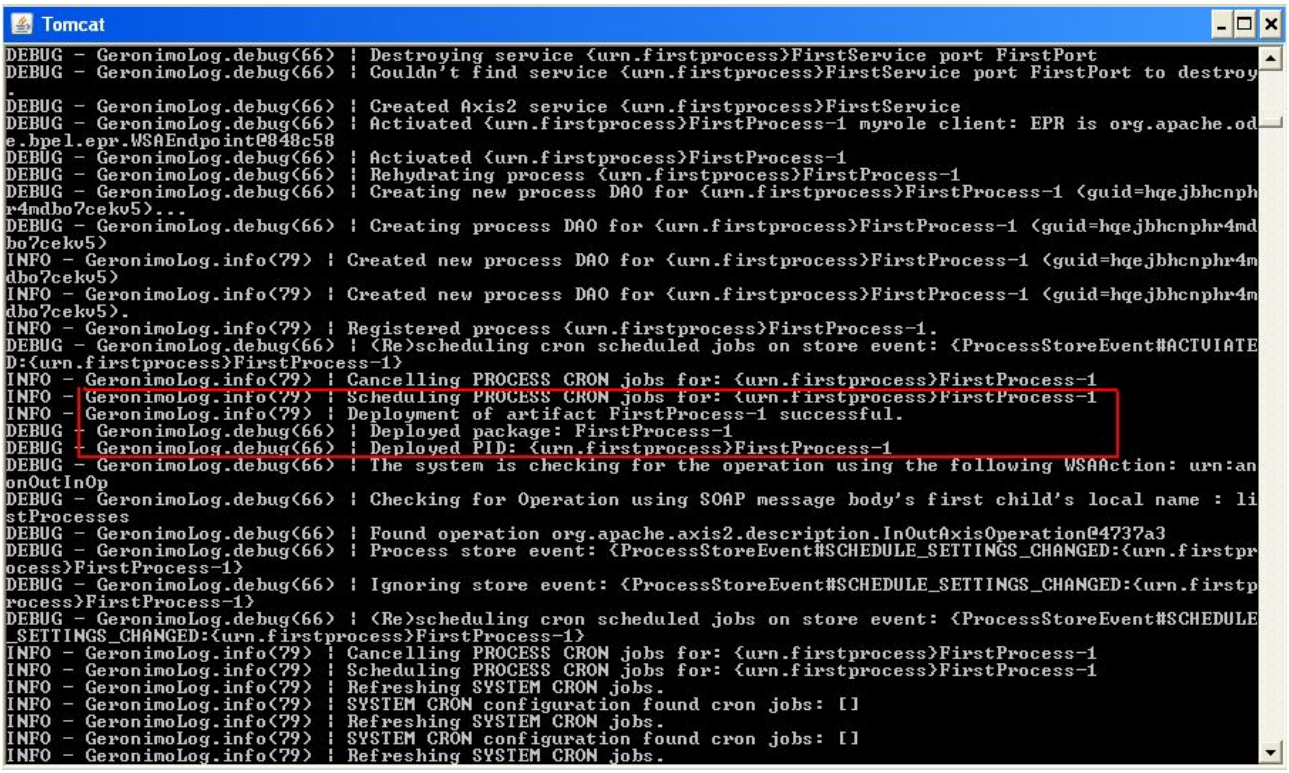
To publish the process, select the *.bpel* file, open the context menu and select *Spagic BPEL->ODE Deploy.*

When the dialog appears, click **Ok**.

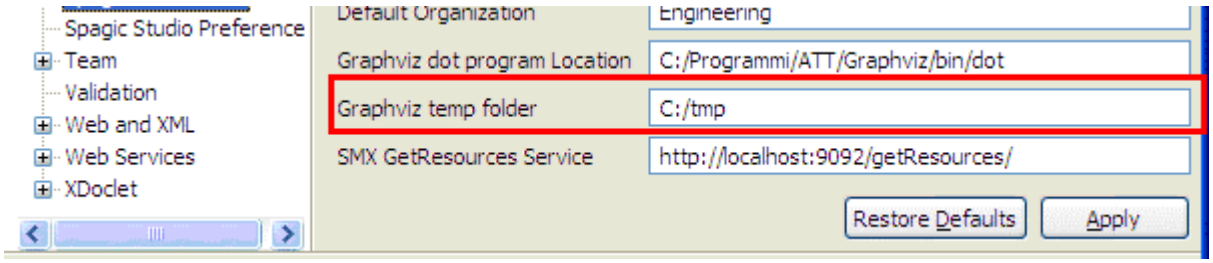
When the publishing is completed, the following dialog will be displayed.



In the Apache ODE console, the deployment is displayed in this way:

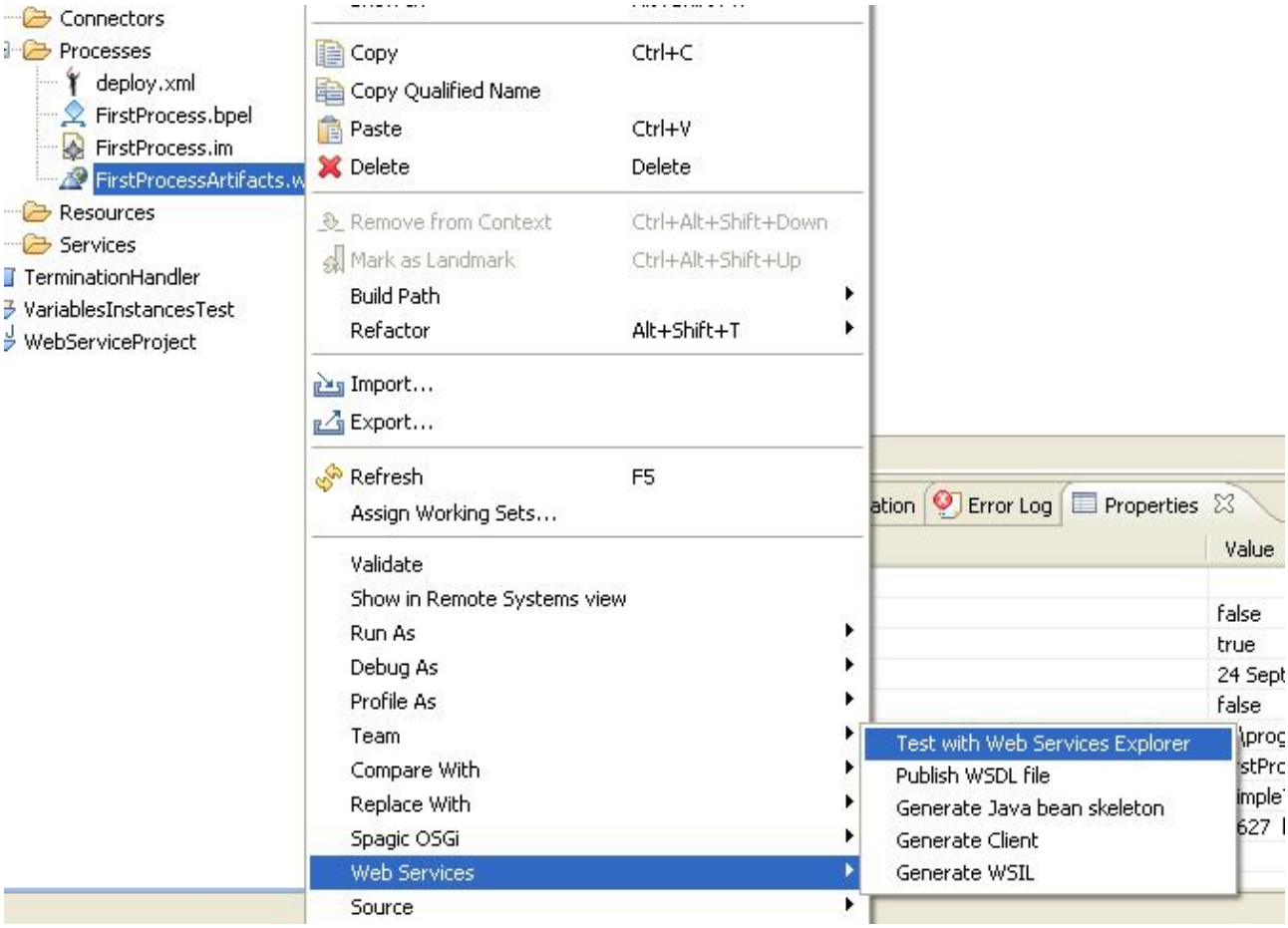


If you have some errors, verify that Apache ODE is running and you've installed *graphviz* tool and you've created the temporary directory that should be configured in the *Spagic General Preferences*, in the item *Graphviz temp folder*.

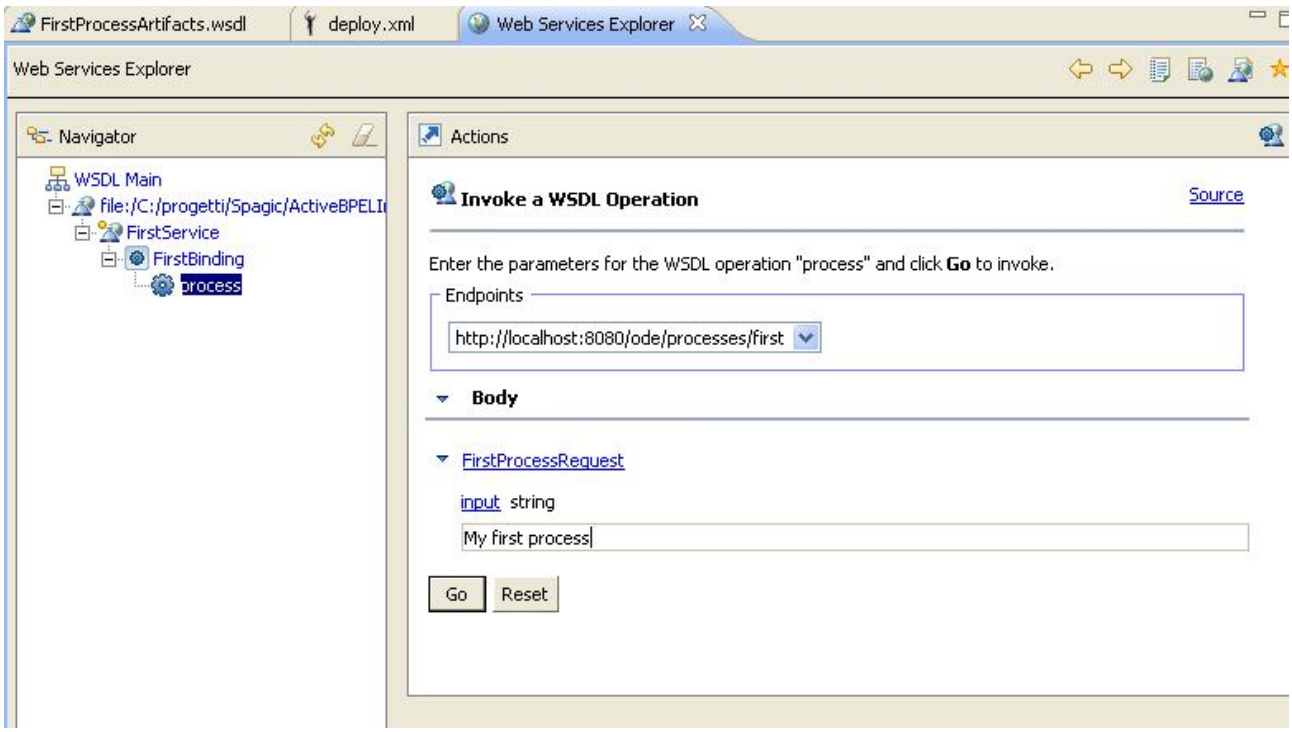


6 Run the Process

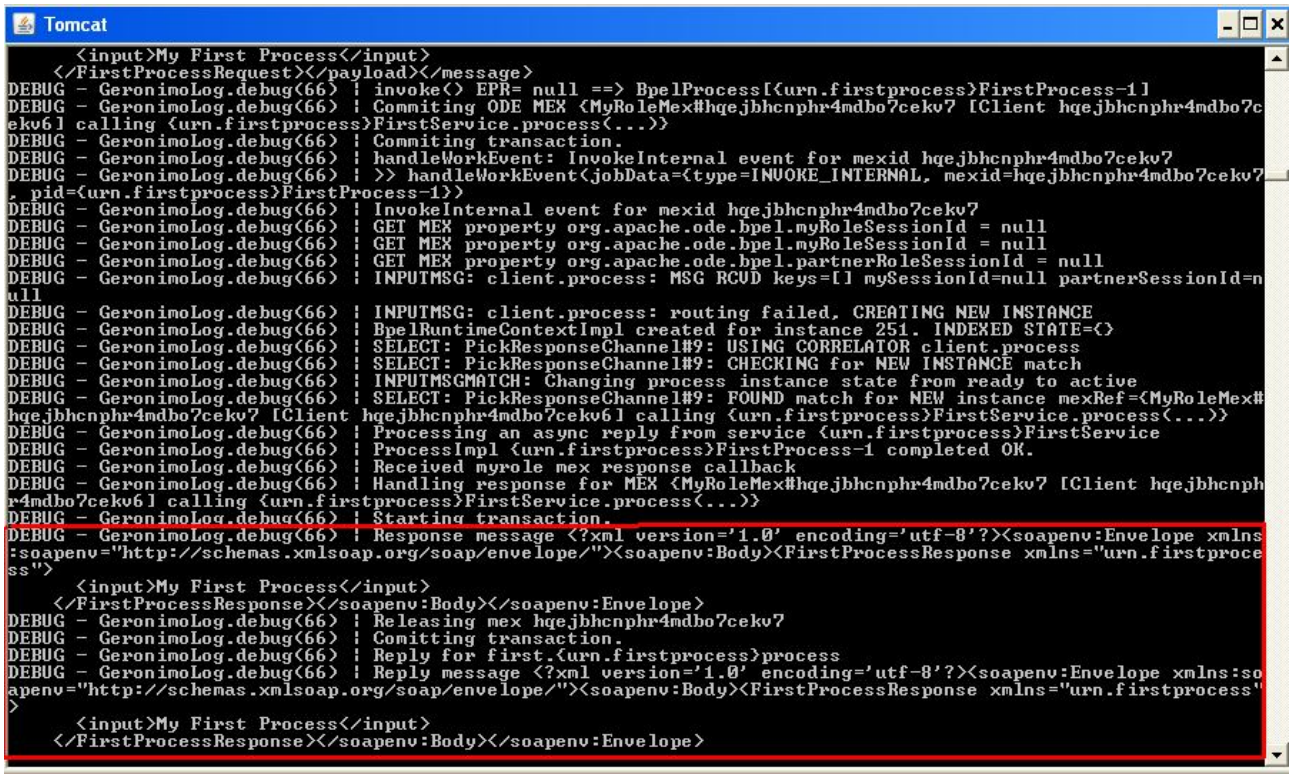
To run the process select the *FirstProcessArtifact.wsdl*, right-click, select *Web Services->Test with Web Services Explorer*.



Navigate the tree on the left until you find *process*, select it, and insert in the *input* field "*My first BPEL process*". Click *Go*.



In Apache ODE, the result should be similar to the following screenshot.

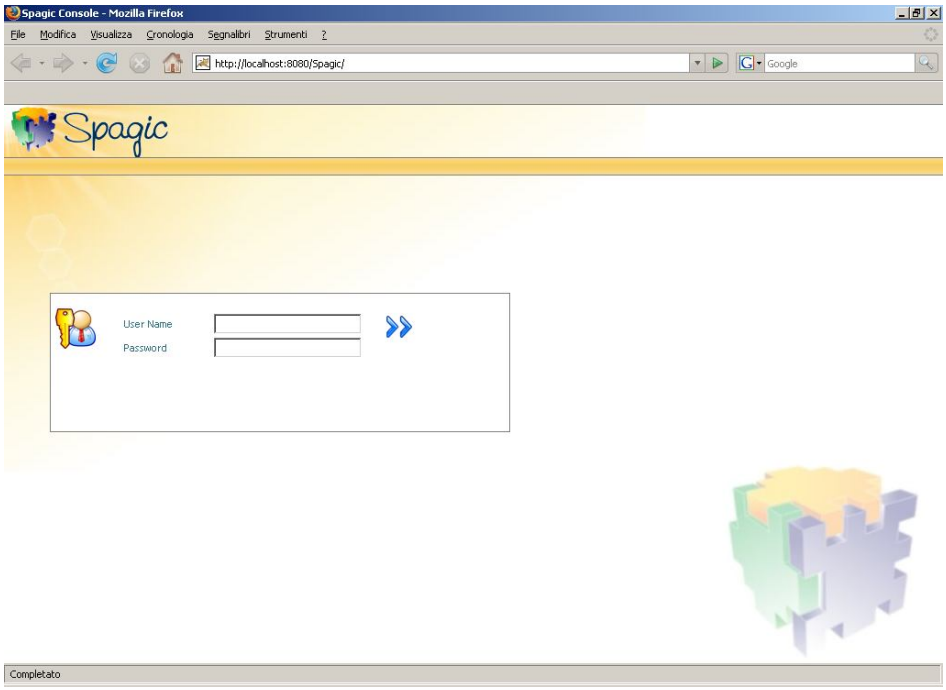


7 Monitor the Process

Spagic Console is a web application to monitor, through Ajax interface, the system information (system monitoring), the services, and the processes (services monitoring).
It's recommended to use the Firefox Mozilla 3.x.
For more details about Spagic Console please read the document *Spagic Studio User Guide*.doc.
Start the Tomcat where you've installed the Spagic Console.

7.1 Authentication

The URL to launch the Spagic Console is: http://localhost:8080/Spagic/.
The first page visualized by the web application is the authentication page.
Insert the user *spagic* and the password *spagic* and then click the button ».



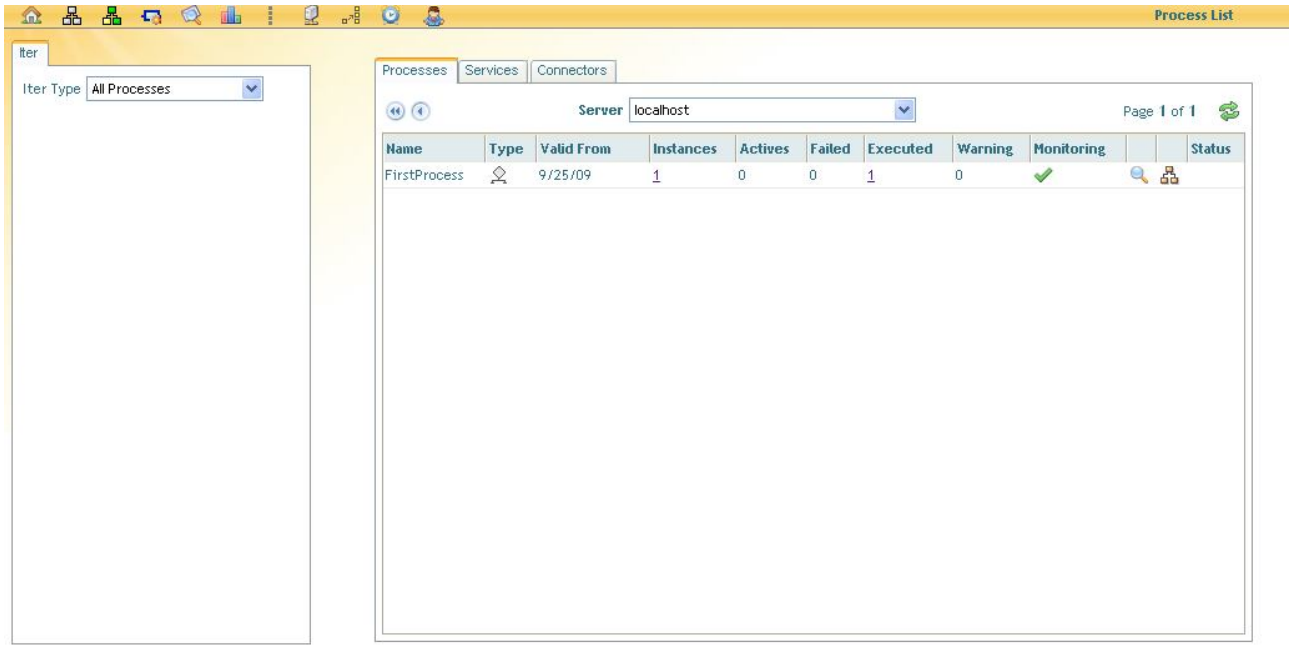
The next page contains the application menu.

7.2 Process List

Select the icon **Process List** () to view all processes and theirs configuration, properties, and static flow diagram. The list contains the following attributes:

- *Name*: process name.
- *Type*: process technology.
- *Valid From*: start validity date.
- *Instances*: number of process instances.
- *Actives*: number of active processes.
- *Failed*: number of failed processes.
- *Warning*:
-  : displays the process detail containing the *description*, *version*, *organization*, *service name*.
-  : displays the process graph.

The process list should contain only *FirstProcess*: the process just deployed and launched.

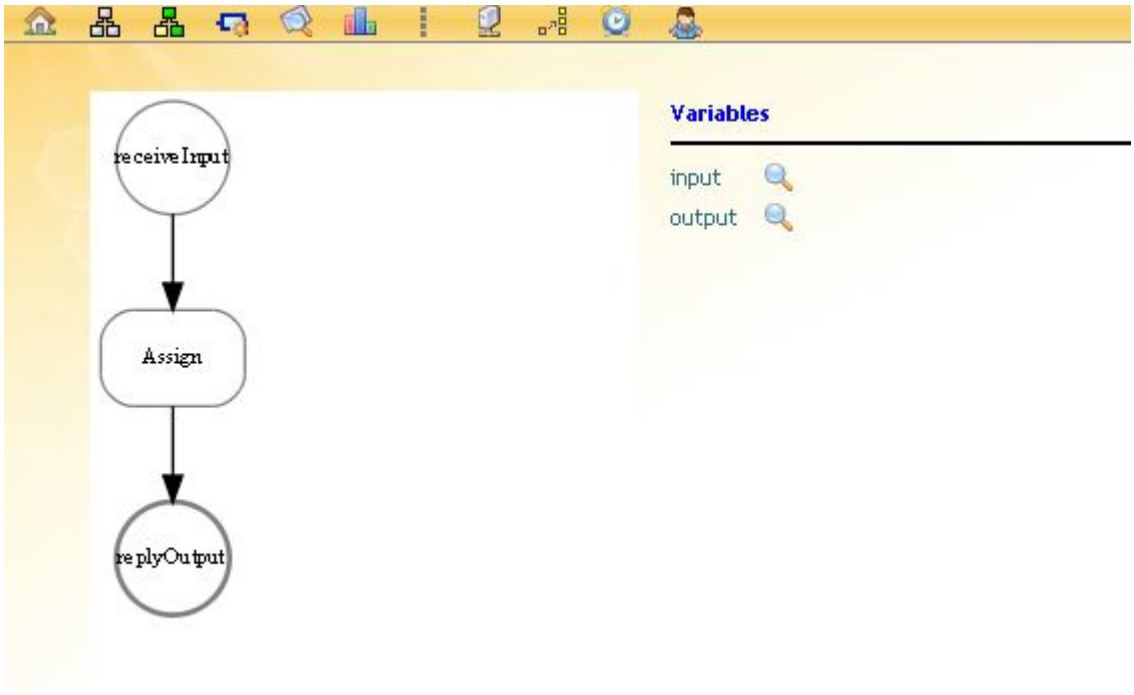


7.2.1 Process Graph

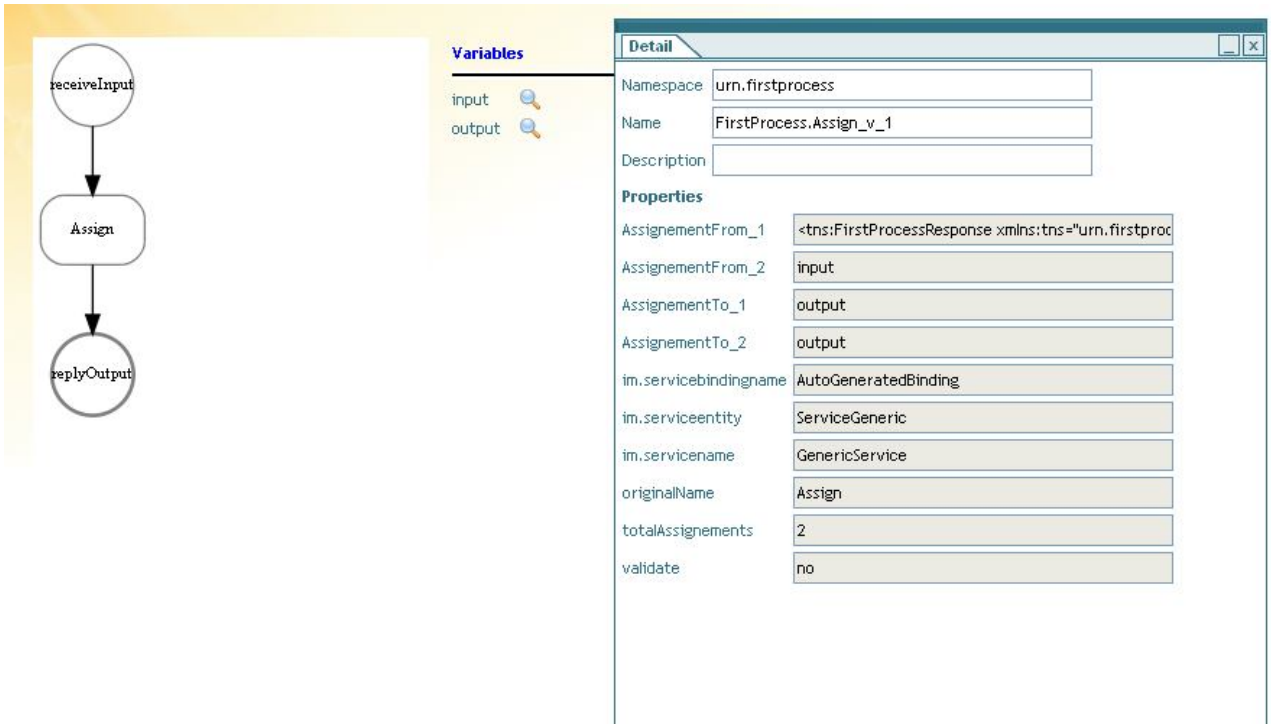
Selecting the icon from the process list, it's possible to display the process graph and on right side the list of global variables defined in the process. In this case *input* and *output*.

In most cases, the activities designed with the BPEL designer editor correspond to the steps visualized in the console, excepting the sequence and the scope activities; these are displayed only if an event handler or a fault handler has been associated to them. In these cases, the step names have been renamed using a prefix *start-* and a suffix *end-* specifying the start and the end activity. See the Samples section for more details.

Selecting the elements on the graph it's possible to display a window containing *Name,* step *Description*, *Service* and its *Properties*.



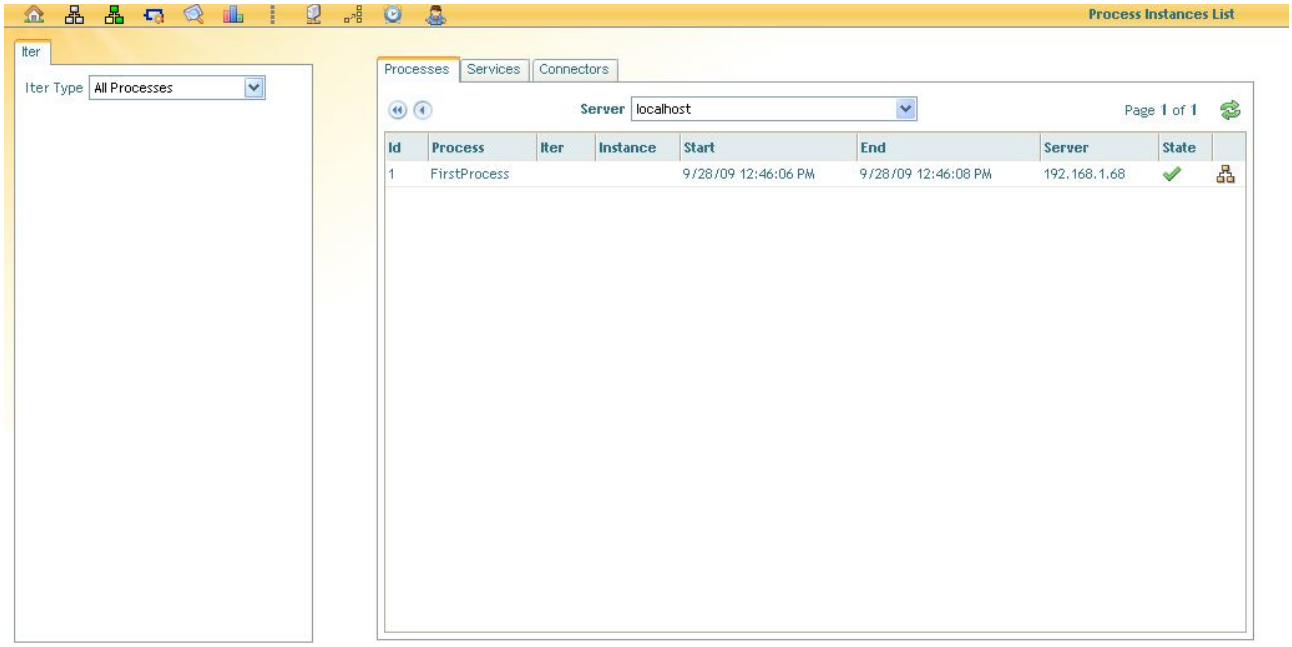
Selecting the elements on the graph it's possible to display a window containing *Name,* step *Description*, *Service* and its *Properties*.



7.3 Process Instances List

Selecting the icon **Process Instances List** ( ) the process instances list is showed. The list contains:

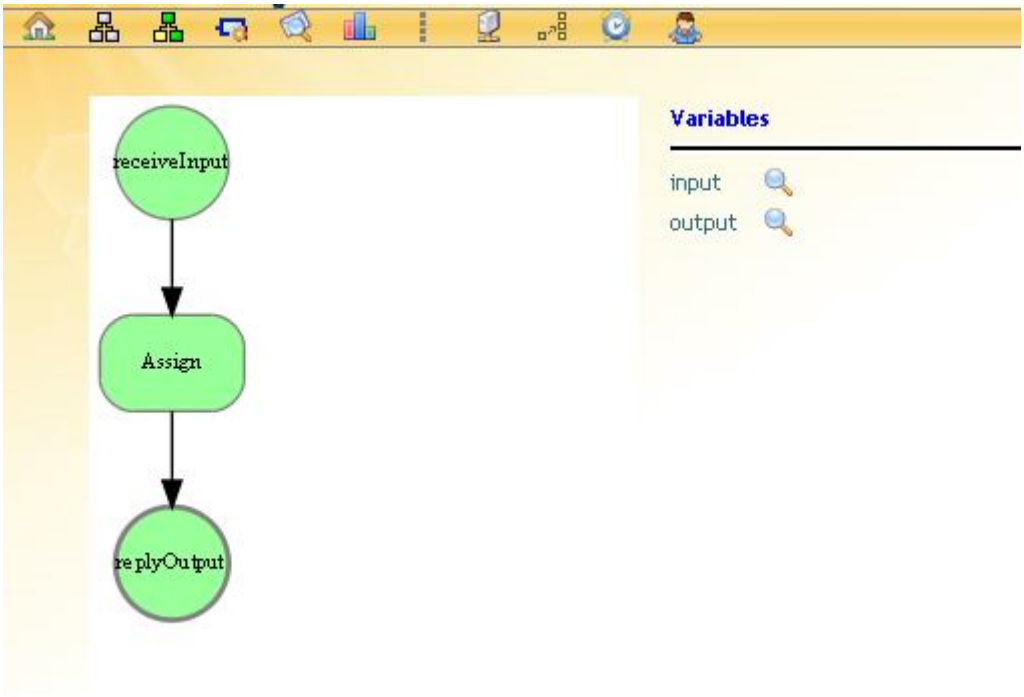
- *Id*: identifier of process instance.
- *Process*: process name.
- *Iter*: name of the iter (optional) associated to the process instance.
- *Instance*: attributes identifying the iter instance (optional).
- *Start*: start execution time.
- *End*: end execution time.
- *Server*: server IP address that executed the process.
- *State*: instance state. It can be *active* (yellow √), *executed* (green √) or *fault* (red ✕).
- : visualizes the process execution detail.



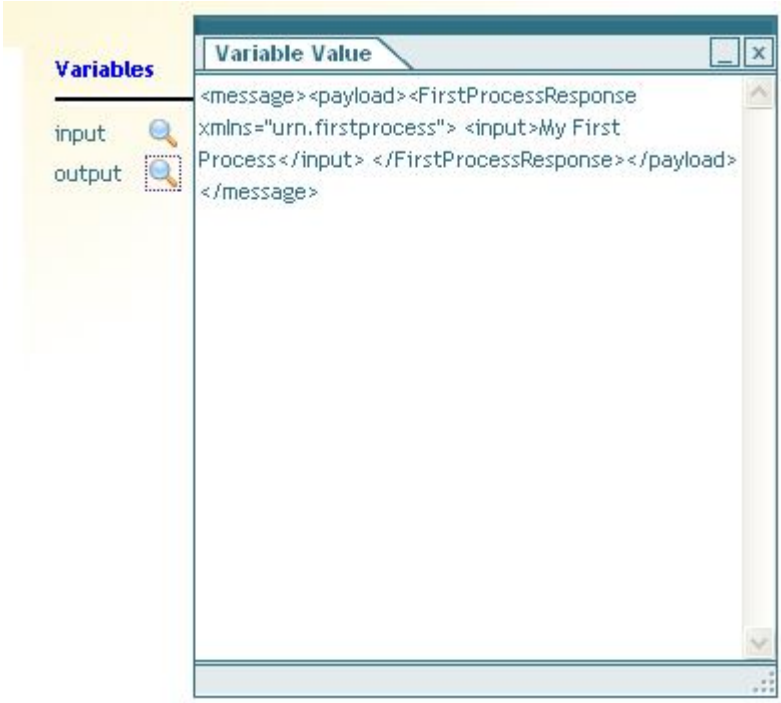
7.3.1 Process Instance Graph

Selecting the icon from the process instances list it's possible to display the graph representing the execution of the process.

The color of the component represents the component state: the green color means *executed*, the yellow color means *active* and the red means error.



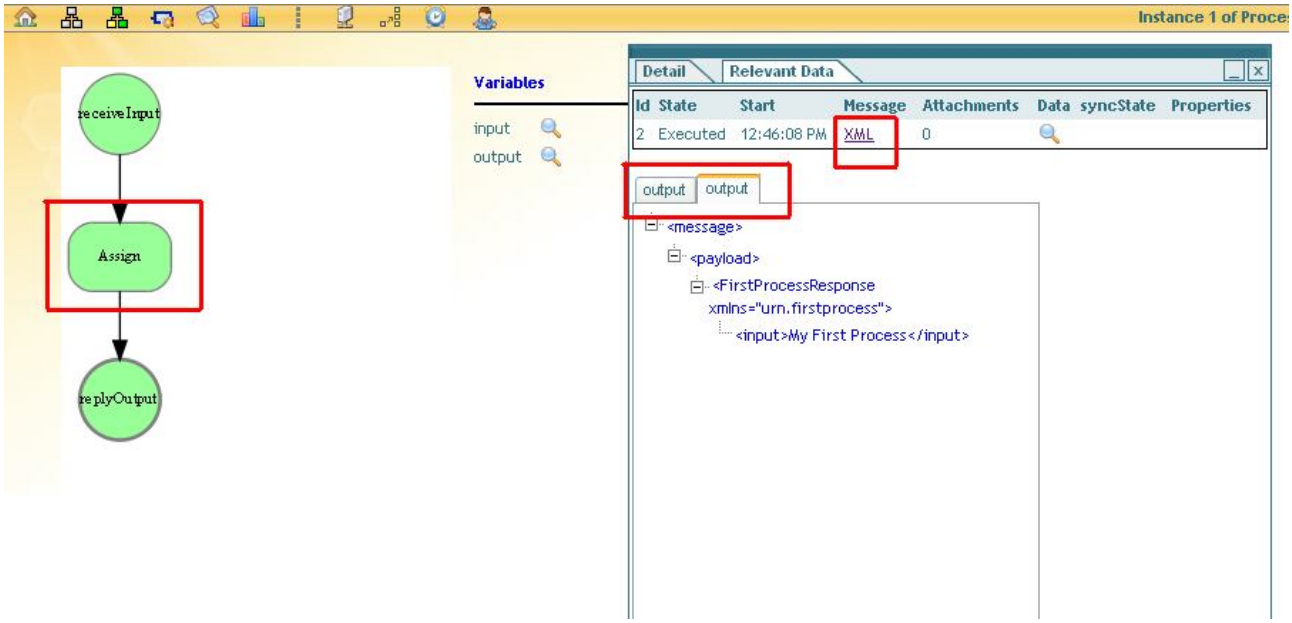
Selecting the icon corresponding to the variable on the right side, the variable runtime value will be displayed. If the process has been executed, the last value of the variable will be displayed.



Selecting the elements on the graph it's possible to visualize the window containing *Detail* and *Relevant Data.*

The *Relevant Data* tab contains the messages having as destination the component selected. Selecting the *XML* link in the list, all the variable modifications will be visualized. For example, selecting the *Assign* step, then the *Relevant Data*

tab and *XML* message, the console will display the two changes to variable *output* executed in this step: the first for initialization, the second for the assignment from *input* variable to *output* variable.

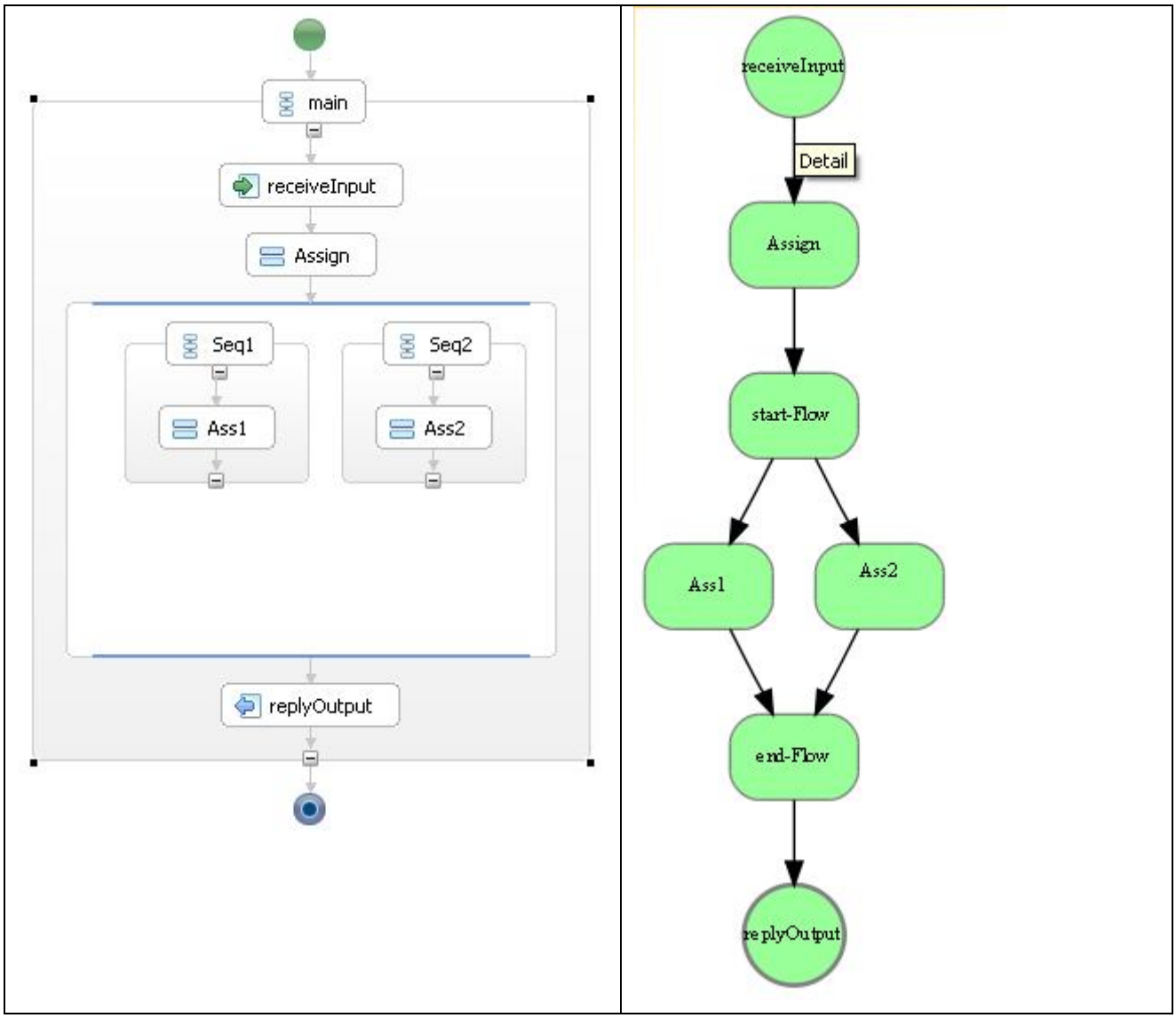


8 Samples

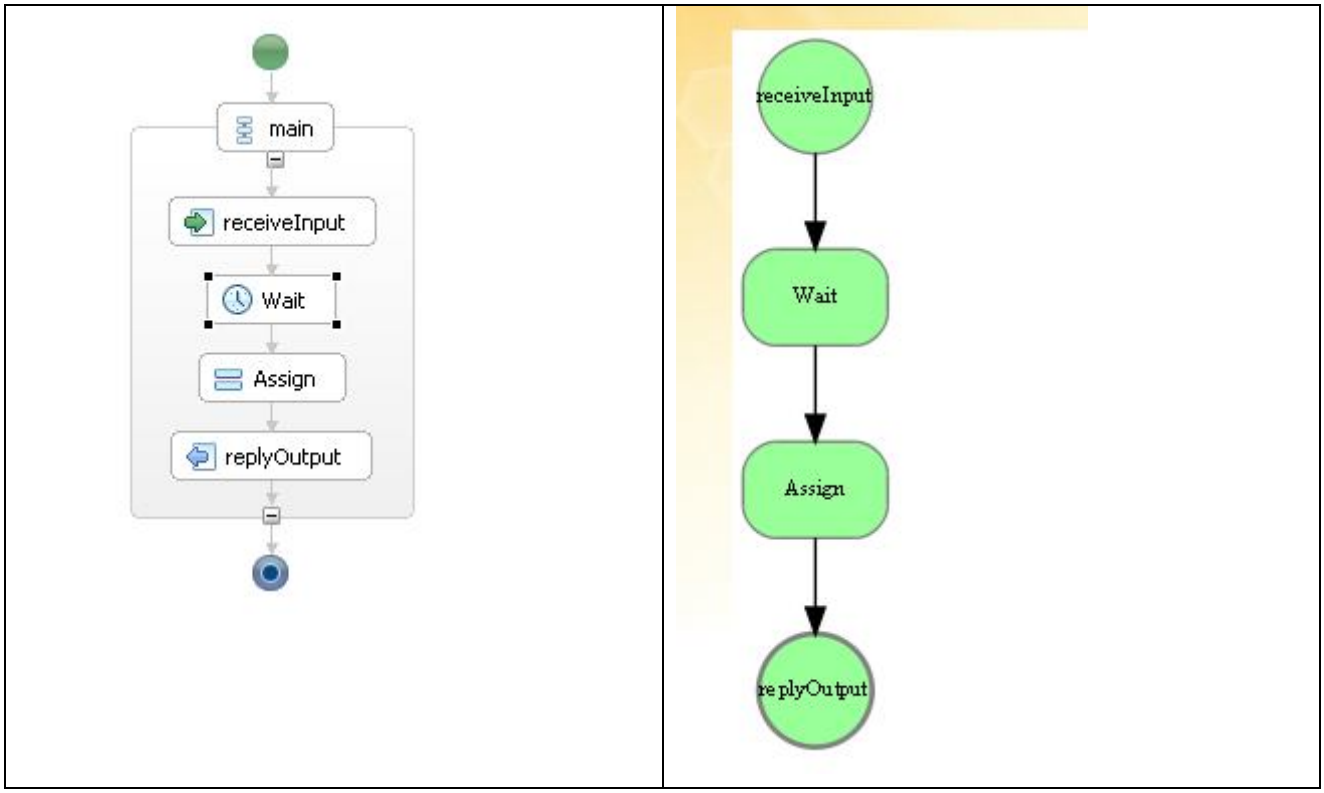
This section will show some examples of the BPEL constructs and their corresponding mapping in the console.

8.1 Flow activity

The flow activity has been represented with a step *start-Flow* that identifies the start end a step *end-Flow* that identifies the end. The names *start-Flow* and *end-Flow* have been built from the original activity name adding the prefix *start-* and the suffix *end-*.

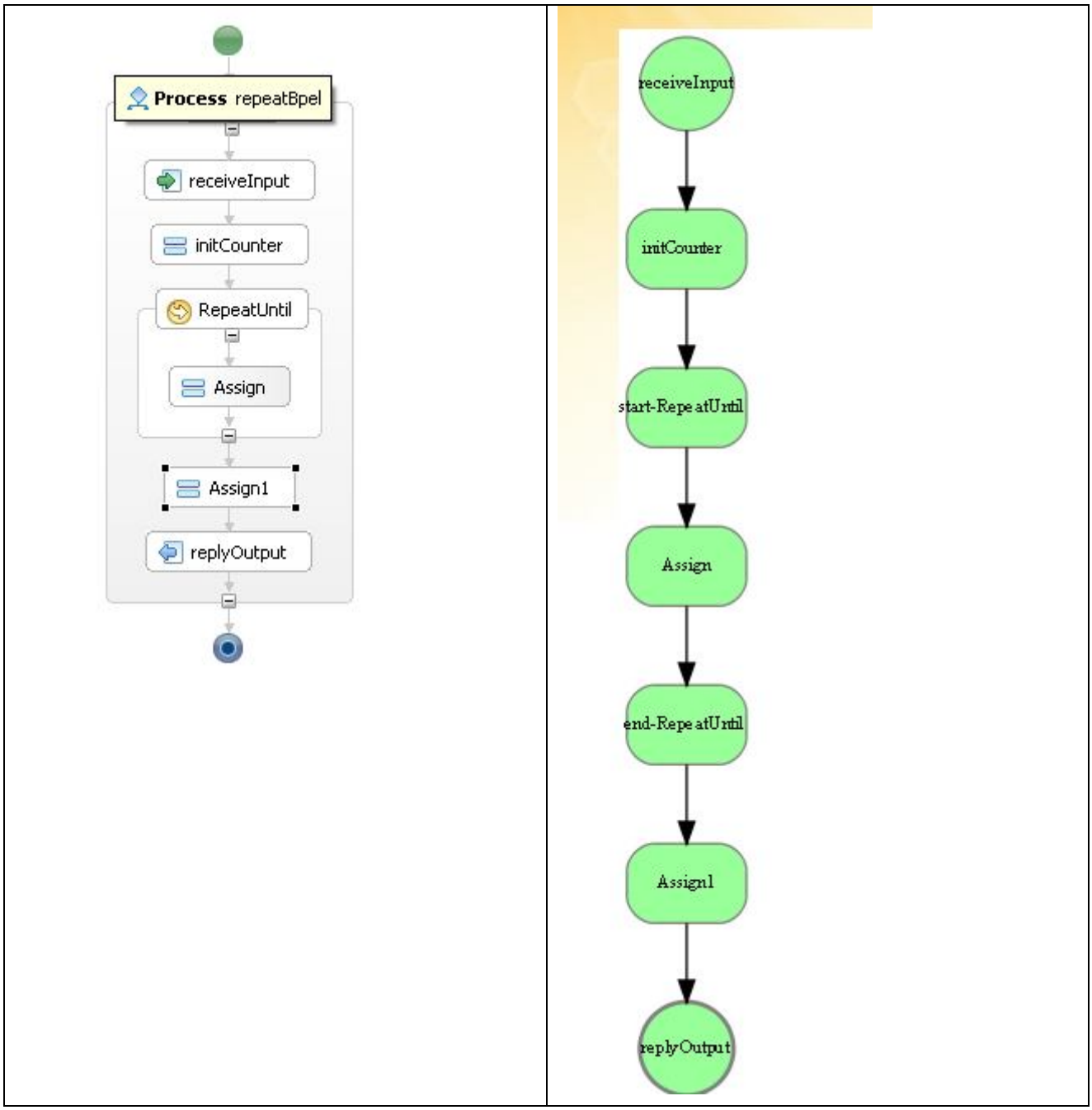


8.2 Wait activity

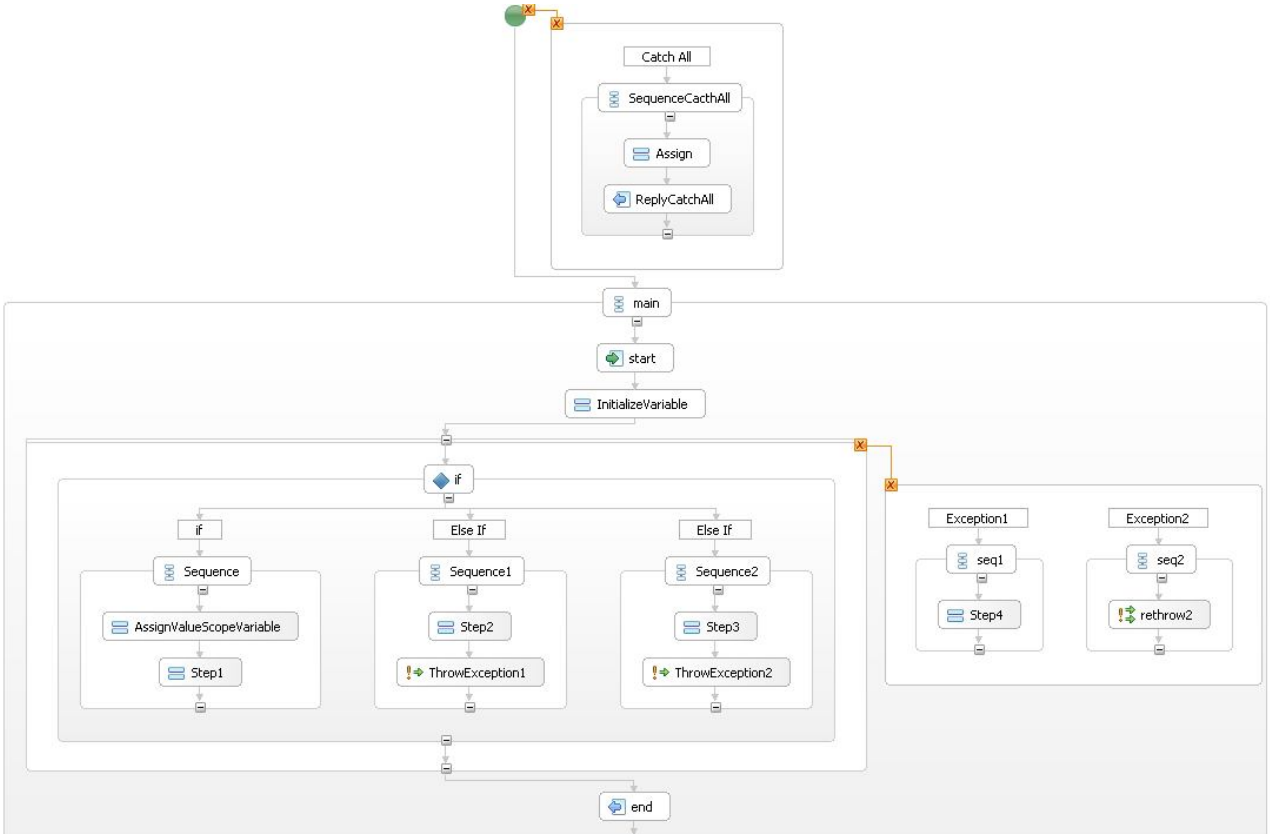


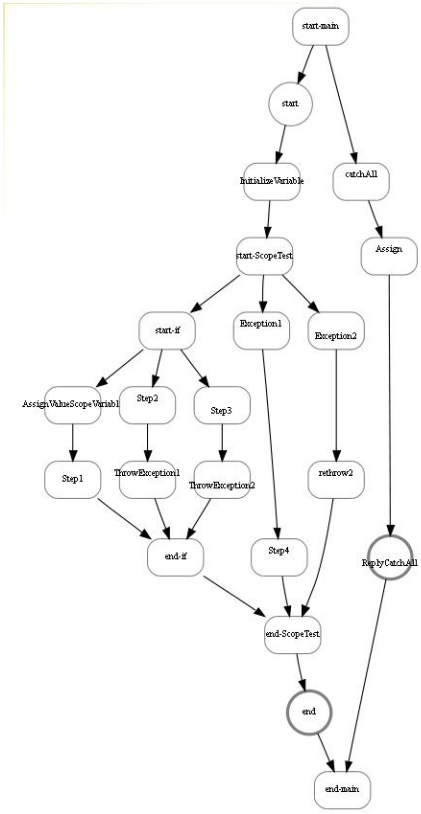
8.3 Repeat Until activity

The *Repeat Until* has been mapped with two steps that identify the start and the end of the cycle. The names *start-Repeat* and *end-RepeatUntil* have been built from the original activity name adding the prefix *start-* and the suffix *end-*.



8.4 IfElse, Throw, Fault Handler





9 Related documents

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this document.

1. *Spagic JBI Components.doc*: detail document about ServiceMix and Spagic JBI components.
2. *Spagic Console.doc*: detail document about *Spagic Console* monitoring application.
3. *Spagic Studio User Guide*.doc: detail document about *Spagic Studio* environment.