Spagic3 Getting Started

Author Gianfranco Boccalon
Andrea Zoppello

1 Document Goal

The goal of this document is to provide you with an introduction on using Spagic3 platform looking at a demo application that should allow you to explore some of the most interesting features of the new platform.

This document is intended for users with knowledge of Spagic 2. If you don't know the Spagic platform, please read previously released documentation before proceeding.

2 Versions History

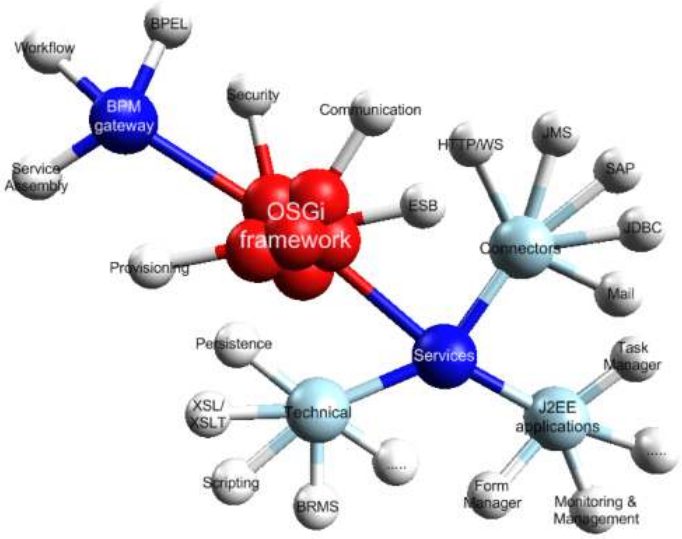
Version/Release n° :	1.0	**Date**	29/06/2007
Description	First release (English version)		

3 Requirements

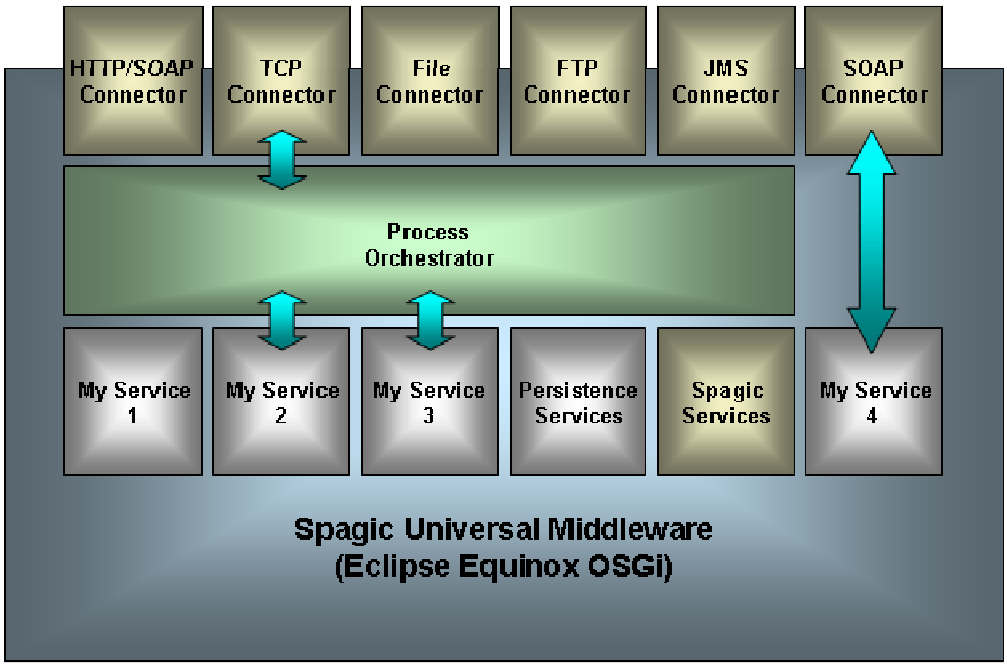
Required Tools	URL for download/Notes	Spagic Studio	Spagic Console
Database	MySQL Oracle	Required	Required
Eclipse Ganymede for Java EE Developers	http://www.eclipse.org/downloads/moreinfo/jee.php	Required	
GraphViz	http://www.graphviz.org/	Required	
JDK 1.6	http://java.sun.com/	Required	Required
Apache Tomcat 5.5.17	http://tomcat.apache.org		Required
Mozilla Firefox 2.0.0.x or later	http://www.mozilla.com		Required

4 Introduction

Spagic3 is a major release whose goal is proposing an Enterprise Universal Middleware OSGi based that enables the development of both single services and complex solutions including orchestration processes, workflows with human activities and features like support of rules engines, registries and multi-node distributions.



The key point of Spagic 3 is that the processes are built by composition of OSGi services that are hosted within an OSGi container (actually Equinox and in future also Swordfish).



The services can simply be configured with a connector (also hosted on the OSGi container) like “My Service 4” in the previous figure, or they can be orchestrated in a complex process by a service called “*Process Orchestrator*” like “My

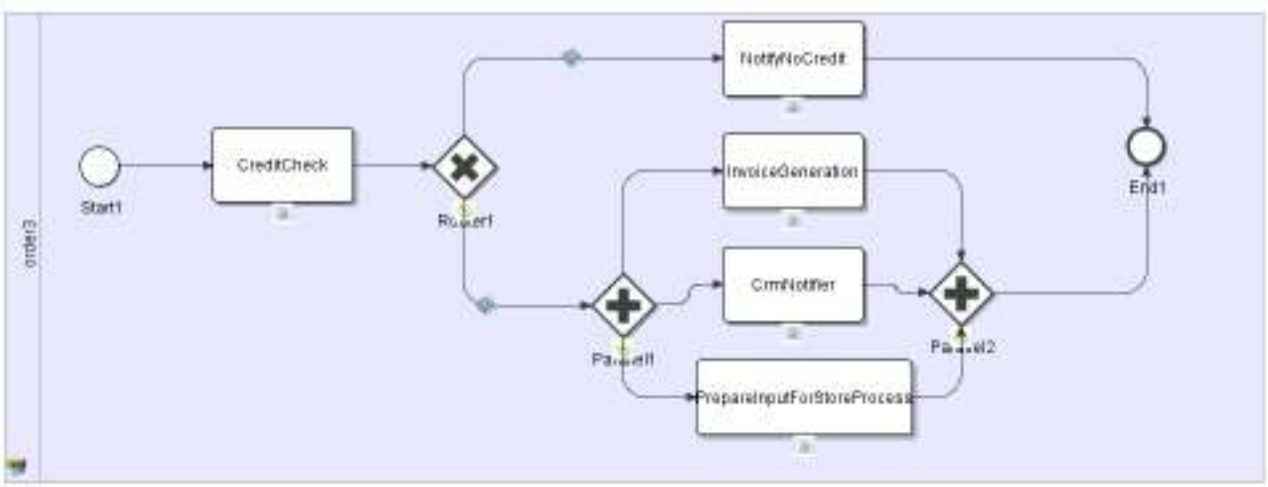
Service 2" and "My Service 3" in the previous figure. Also this kind of complex processes can be configured with a connector, for the activation of the process.

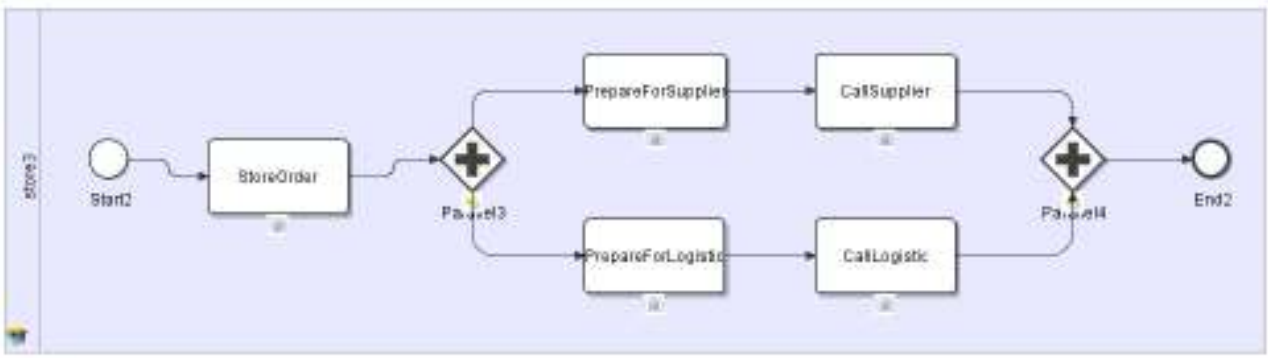
The steps for creating a new service in Spagic 3 are:

- Create a new BPMN process with Spagic Studio
- Configure all the services that will be used by the process steps. Each service is configured in a declaratively way.
- Define the connectors that will be used to activate the process. Also this step is done in a declaratively way.

To demonstrate the capabilities of the new Spagic3 platform we provide a demo Web application that activates a Spagic process that simulates the management of an order.
Technically this means executing two different Spagic processes: **the first one is exposed through an HTTP connector**, and simulates the management of the order.
The second one is exposed by a file connector, and is activated by a file produced at the end of the order process, and simulates the storing of the order.
These processes will be executed within the Equinox container by the Spagic bundle called "Process Orchestrator" (based on JBPM) that allows the orchestration of OSGi services.

Here you can see the two processes:





The tools that Spagic3 provides are:

- **Spagic Service Manager:** the biggest difference, compared to Spagic2, is that in Spagic3 the Service Manager is now an **OSGi container** (actually Equinox) while in the previous release it was based on Apache ServiceMix. This means that now you can gain the advantages of the OSGi architecture customizing your application "on demand", for example choosing the persistence of the messages only if your project requires it.
- **Spagic Studio**: Eclipse environment that allows the use of a single interface in order to create components and processes managing the entire development cycle: design, service registry configuration (UDDI & ebXML), metadata management, WSDL generation, rules definition, mapping, custom services and orchestration.
- **Spagic Console**: for the monitoring, through Ajax interface, of the system information (system monitoring), services & processes (services monitoring).
- **Persistence layer - MetaDB**: the different services and the rules of process and extraction of the relevant information are described and classified through a metadata system. Therefore the repository manages the entire life cycle of the components and traces their use in order to optimize the monitoring activities. The registration is made through Spagic studio and at runtime on Service Manager by means the specific listener.

The additional stuff that we provide for this demo:

- An Eclipse project containing the two Spagic processes and the configuration for the OSGi services used by the steps.
- A Web application that simulates the creation of a new order, and that activates the Spagic Order process.

5 Installation of the Demo

Before starting with the demo, you should install some Spagic tools:

- Spagic MetaDB
- Spagic Console
- Spagic Studio

The installation instructions for these tools are in the Appendix of this document and are a brief of the document "Spagic Getting Started".

In this document we will use the notation *{SPAGIC3_INSTALL_ROOT}* for referring to the folder where you installed Spagic3 (you should not declare the variable *SPAGIC3_INSTALL_ROOT* in your system).

Follow these steps:

- Unzip the file *spagic3-preview.zip* that contains the Equinox installation and all other artifacts necessary for the demo. This Equinox release contains also all Spagic bundles.
- Edit the *{SPAGIC3_INSTALL_ROOT}/ spagic-service-manager-3.0-preview /eclipse.ini* file within the Equinox folder.

In this prerelease we have some absolute paths that you should adjust before trying the demo. We will fix this on the final release.

The *eclipse.ini* file contains:

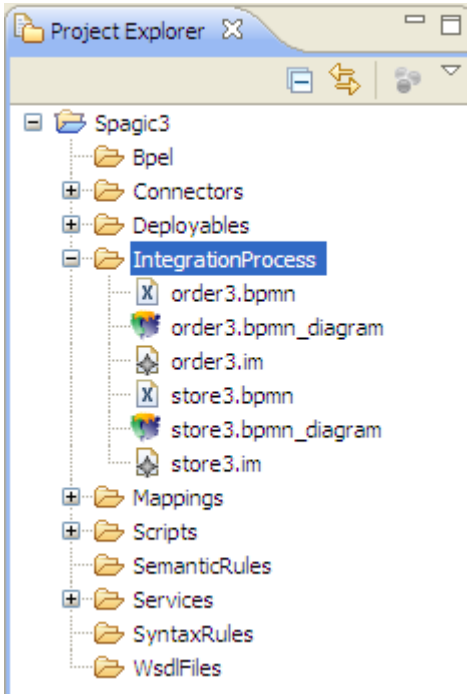
- The variable ***logback.configurationFile*** that contains the absolute path to the *logback.xml* file used for logging. The logback.xml file is actually within the Equinox folder, but you can put it anywhere.
- The variable ***spagic.home*** that contains the absolute path to the folder where the service definitions are searched.

- Start Equinox with the options "-debug –console":

  eclipse.exe -debug -console

- Import the Eclipse project within the folder *{SPAGIC3_INSTALL_ROOT}/Spagic3DemoProject* in a new workspace, with Spagic Studio.

The project has the following structure:



Within the *IntegrationProcess* folder there are the 2 Spagic processes used by the demo.

Before deploying the processes, we must activate all the services used by the processes.

The service definitions are read from Equinox from the folder ***{SPAGIC3_INSTALL_ROOT}\services***: you can "hot deploy" new service configurations putting the *.service* files within this folder.

Actually the .service files contain absolute paths to the resources necessary to the service activation: for example, a Groovy service definition contains the absolute path to the Groovy file.

This limitation will be removed on the final release.

We suggest to use the folder ***{SPAGIC3_INSTALL_ROOT}\resources*** for the services resources.

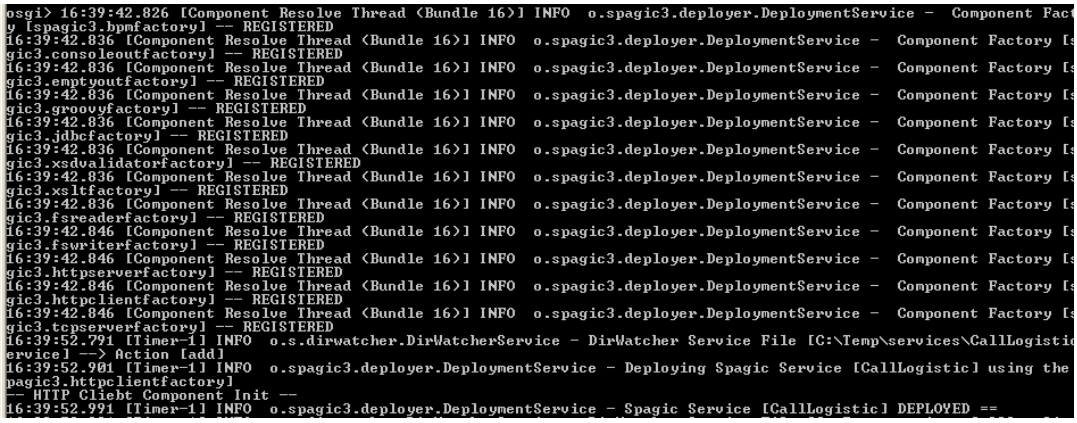
We provide already the ***{SPAGIC3_INSTALL_ROOT}\resources*** folder preloaded: this folder is the merge of the files contained in the folders *Mappings* and *Scripts* of the Eclipse project.

Let's proceed with the installation:

- Edit the services definitions (the .service files) within the ***Services*** folder of the Eclipse project, to substitute the variable {SPAGIC3_INSTALL_ROOT} with the absolute path of Spagic3 installation.

  These are the service definitions of the services used by the process steps.

- Edit the services definitions (the .service files) within the ***Connectors*** folder of the Eclipse project, to substitute the variable {SPAGIC3_INSTALL_ROOT} with the absolute path of Spagic3 installation.
  These are the service definitions of the connectors used to activate the processes.
- Copy all the services definitions from the project folders ***Services*** and ***Connectors*** to the folder *{SPAGIC3_INSTALL_ROOT}\services*. You should see something like:



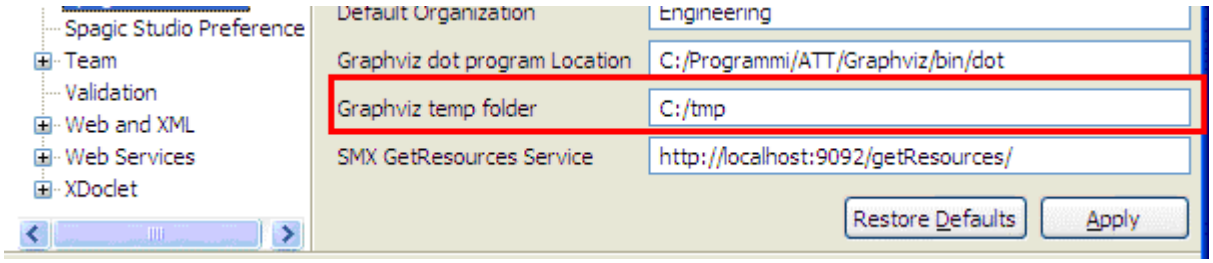
  You should see some rows like “Spagic Service [SERVICE NAME] DEPLOYED”.

Now you must deploy the Spagic processes on database.

- Select the BPMN files (one at a time) and choose the command *Spagic Workflow->Generate Workflow Definition (JPDL)*.

- Select the generated JPDL files (within the folder *Deployables*, one at a time) and deploy them with the command *Spagic Workflow->Deploy (JPDL)*.

  If you have some errors, verify that you installed *graphviz* tool and you created the temporary directory that should be configured in the Spago Preferences, in the item *Graphviz temp folder*.



- Install the Spagic3Demo web application on your servlet container.
  In the folder C:\Temp create a file called order.txt with a row containing the value 1.
- Now you can start the Spagic3Demo application. You should see this page:

- Now you are ready to start the demo: simply enter a value on the field Quantity and press “Order Submit”.
  This sample application starts the process order3 by a SOAP call.
  When the process order3 finish, it writes a file on a folder that is polled by another process, store3.
  Store3 starts, retrieves the file, and it uses the information to make some checks and simulating the processing of an order.
  You should see the newly created processes on the Spagic Console:

Id	Process	Iter	Instance	Start	End	State	
606	store3			7/3/09 5:03:27 PM	7/3/09 5:03:29 PM	✓	
605	order3			7/3/09 5:03:20 PM	7/3/09 5:03:23 PM	✓	

6 Limitations of this preview release

On this preview there are some limitations that will be removed on the final release.

The limitations are:

- The feature that allows restarting failed processes is not available.
- The feature that allows stopping the running processes is not available.

- The runtime actually doesn't support the definition of datasources.
- The runtime actually doesn't support the persistence of messages between services. In final release it'll be possible to choose between direct communication and using intermediate queues.
- To create new Spagic projects with Spagic Studio you must use the wizard "Spagic2 project". Actually there isn't a wizard "Spagic3 project".

7 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 Console.doc*: detail document about *Spagic Console* monitoring application.
2. *Spagic Studio User Guide*.doc: detail document about *Spagic Studio* environment.

8 Appendix

8.1 Persistence layer – MetaDB

To setup the metadatabase follow these steps:

1. Install MySQL Server (release 5.0 or higher) or Oracle (release 9i or 10g) or PostgreSQL (release 8.1)
2. Create a new schema "*spagic"* and a user "*spagic"* with password "*spagic*" and the permissions for writing into the schema.
3. Generate the tables using the "*spagic-metadb-<database type>.ddl"* released in the *spagic-metadb* package.
4. Launch the scripts "*setup-<database type>.sql"* released in the *spagic-metadb* package, to load all the configuration tables.
5. **Optional step**: install the backup database for monitoring data, if you want to clean periodically the monitoring data. Create a new schema "*spagic-bck"* and a user "*spagic-bck"* with password "*spagic-bck*" and the permissions for writing into the schema.
6. Generate the tables using the *"spagic-bck-<database type>.ddl"* released in the *spagic-metadb* package.
7. If you installed the MySQL Server and the processes's messages contain the attachments bigger than 1 MB, you have to configure the variable max_allowed_packet = 16M into the file: {*MySQL Server path*}*/my.ini* and restart the server.
8. You must define a user and a scheme in your database to contain jBPM tables, possibly but not necessarily different to the scheme and user of the metadb.
9. There's no need to run a DDL script to create jBPM tables, because they are created at the first run or deploy of a jBPM process (hibernate driven creation). The jBPM user must have the grants to create tables on his schema at least at the first deploy or run of a JBPM process.
10. Unfortunately when the JBPM database is automatically created the table JBPM_VARIABLE_INSTANCE is defined to contain a maximum of 255 characters for String variables. In Spagic jbpm variables could countain xml_payloads so we need modify that table with an alter command to remove this limitation. Simply run the alter_jbpm.ddl on database instance once the db has been created.

8.2 Spagic Studio

To install Spagic Studio on client machine follow these steps:

1. Get graphviz installation package from http://www.graphviz.org/ and install it. If you're using windows, simply run graphviz executable file and follow the wizards. During the installation steps take in mind the location where the executable dot program was installed.

   In Windows (Italian language) default installation graphviz will install the dot program in C:\Programmi\ATT\Graphviz\bin\dot.exe

2. Spagic Studio is distributed as an Eclipse application, so you have simply to get it from Spagic distribution (from the package *spagic-studio*) and unzip it.
   In this document we will refer to the Eclipse folder as *SPAGIC_STUDIO_HOME*.
   If you want to create Spagic Studio from scratch, instead of using the distribution package, please refer to the document "*How to cook your Spagic Studio*".

3. Spagic Studio needs to connect to Spagic metadabase. Ensure that the database is running and that permissions are configured properly on database server.

To start you only need to launch eclipse.exe in *SPAGIC_STUDIO_HOME.*

8.3 Spagic Console

To install Spagic Console follow the next steps:

1. Install the Apache Tomcat 5.5.17; to install it refer to its installation documentation (http://tomcat.apache.org).
2. Install the database JDBC driver for the database you are using in the Tomcat *common\lib* folder:
   - If you are using MySQL, please use *mysql-connector-java-5.0.7-bin.jar* or later.
   - If you are using Oracle 9i/10g use the JDBC driver for JDK 1.4 and Oracle 10g (it works also on 9i). The driver name is *ojdbc14.jar*.

   You can retrieve both the drivers from the *lib* folder of the ServiceMix released in Spagic.
3. Install the web application *SpagicConsole*: you should copy into *apache-tomcat-5.5.17\webapps* folder the *Spagic.war* released with Spagic.

4. **If you used a configuration different from the default suggested in this document**, before starting Tomcat it 's necessary to verify the following Spagic Console **configuration files** and update them:
   - **If you installed the database on a different machine from the Tomcat machine**, update the file *\SpagicConsole\META-INF\context.xml* so it could link to the Spagic database (with the JNDI name *jdbc/spagic*).
   - **If you installed an Oracle database instead of MySQL** then configure the "*\SpagicConsole\WEB-INF\classes\hibernate.cfg.xml*" file so that the property "***hibernate.dialect***" has the value "*org.hibernate.dialect.OracleDialect*".