How to cook your Spagic

Author Gianfranco Boccalon
Andrea Zoppello

1 Document Goal

The goal of this document is to provide you the information necessary to transform a clean ServiceMix installation in a Spagic environment.

1.1 Document Conventions

In this document we will refer to the version of ServiceMix as *{SMX_VERSION}*, to the version of the Spagic platform as *{SPAGIC_VERSION}*, and to the version of Apache ODE as *{ODE_VERSION}*.

Current Spagic release is 2.4.0.

Current supported ServiceMix release is 3.3.

Current supported Apache ODE release is 1.2.

2 Versions History

Version/Release n° :	1.0	**Date**	24/07/2007
Description	First release (English version)		
Version/Release n° :	2.0	**Date**	15/10/2007
Description	Updates for ServiceMix 3.1.2		
Version/Release n° :	3.0	**Date**	21/01/2008
Description	Updates for Spagic 2.0.0		
Version/Release n° :	3.1	**Date**	11/02/2008
Description	Added the ServiceMix shared patch.		
Version/Release n° :	3.2	**Date**	03/03/2008
Description	Added the spagic-utils.jar library, used in Spagic 2.1.0.		
Version/Release n° :	3.3	**Date**	03/04/2008
Description	Removed the patches to the Groovy component. Added the changes for ODE 1.1.1.		
Version/Release n° :	3.4	**Date**	29/07/2008
Description	Changes for Spagic 2.2.0.		
Version/Release n° :	3.5	**Date**	10/10/2008
Description	Changes for Spagic 2.3.0.		
Version/Release n° :	3.6	**Date**	23/10/2008
Description	Changes for Spagic 2.3.1. Added the Spagic-rules component. Added a patch to the aggregator.		
Version/Release n° :	3.7	**Date**	22/12/2008
Description	Changes for Spagic 2.4.0		

3 Configuration Steps

3.1 Edit the file bin\servicemix

If you use Linux it's necessary to change the startup script *servicemix*.
In the function *locateHome* you should add the export of the variable *SERVICEMIX_HOME*.

```
locateHome() {
  if [ "x$SERVICEMIX_HOME" != "x" ]; then
     warn "Ignoring predefined value for SERVICEMIX_HOME"
  fi
   SERVICEMIX_HOME=`cd $DIRNAME/..; pwd`
  if [ ! -d "$SERVICEMIX_HOME" ]; then
     die "SERVICEMIX_HOME is not valid: $SERVICEMIX_HOME"
  fi
  export SERVICEMIX_HOME

  if [ "x$@" != "x" ]; then
     SERVICEMIX_DATA=.
  else
     SERVICEMIX_DATA=$SERVICEMIX_HOME
  fi
}
```

This change is necessary for the proper working of the feature "Automatic backup" of the Spagic Console.

3.2 Edit the file conf\servicemix.conf

You have to add some folders to the *servicemix.conf* file.
The file should contain the following lines (exactly in this order):

```
load ${servicemix.home}/conf
load ${servicemix.home}/lib/hib/*.jar
load ${servicemix.home}/lib/optional/*.jar
load ${servicemix.home}/lib/talend/*.jar
load ${servicemix.home}/lib/*.jar
load ${servicemix.home}/resources/xsd
load ${servicemix.home}/resources/keystores
load ${servicemix.home}/resources/properties
load ${servicemix.home}/resources/drl
load ${servicemix.home}/resources/templates
```

The folders highlighted in bold should be created.

In the folder *${servicemix.home}/resources/properties* create an empty text file called ***spagic.properties***. This property file is used for supporting the feature SPAGIC-232 (Support different deployment environments allowing using parameters when configuring the components).

3.3 Edit the file conf\servicemix.xml

In *servicemix.xml* file you have to configure the listeners namespaces. Add the following definitions at the beginning of the file (in the *beans* tag, the root element):

```
xmlns:syncListeners="java://org.spagic.smx.listeners.sync"
xmlns:debugListeners="java://org.spagic.monitoring.smx.listeners.debug"
xmlns:monitoringListeners="java://org.spagic.monitoring.smx.listeners"
```

and then add the listener for Synchronizer error management (for the details see *Spagic Studio Components.doc*) in the *sm:container* tag:

```
<sm:listeners>
  <!-- Spagic: listener for error management with Synchronizer -->
  <syncListeners:SynchronizerFaultListener/>
</sm:listeners>
```

3.4 Edit the file conf\component.properties

Edit the file *conf\component.properties* and add the following line:

servicemix-http.retryCount=0

This allows avoiding the retry mechanism enabled by default on HTTP Binding Component.

3.5 Additional libraries

These libraries are used by the Spagic components and by the patched ServiceMix components.

You should retrieve them from the Spagic distribution and copy in your ServiceMix.

The libraries are:

- *{SERVICEMIX_HOME}\lib*
  - commons-codec-1.3.jar (Used for encoding and encoding in base64)
  - commons-dbcp-1.2.1.jar (Dependency of commons-dbutils)
  - commons-dbutils-1.1.jar (New code for simple jdbc component)
  - commons-lang-2.2.jar (Dependency of commons-dbutils)
  - commons-httpclient-3.1.jar (Used by sms component)
  - groovy-all-1.0-JSR-06.jar (Groovy Lightweight SE)
  - groovy-engine-20070112.jar (Groovy Lightweight SE): JSR-223 compliant Groovy Engine implementation. Retrieved from http://repo.open.iona.com/maven2/com/sun/script/groovy-engine/20070112/. Contains also the file for registering the Groovy engine in groovy-engine-20070112.jar/META-INF/services/javax.script.ScriptEngineFactory.
  - script-api.jar (Groovy Lightweight SE): retrieved from the reference implementation of JSR-223
  - mysql-connector-java-5.0.7-bin.jar (My sql connector driver)
  - mysql-connector-java-5.0.7-license.txt (MySQL driver license)
  - ojdbc14.jar (Oracle JDBC driver)
  - ojdbc14_license.txt (Oracle driver license)
  - postgresql-8.1-405.jdbc3.jar (PostgreSQL connector driver)
  - postgresql-8.1-405.jdbc3-license.txt (PostgreSQL driver license)
  - spring-jdbc-2.5.5.jar (used by JDBC advanced component)

- spring-tx-2.5.5.jar (used by JDBC advanced component)
- joda-time-1.5.2.jar (used by event management components)
- commons-io-1.4.jar (used by event management components)
- quartz-1.5.2.jar (used for installing Quartz component)
- velocity-1.5.jar (used by velocity component)

- *{SERVICEMIX_HOME}\lib\hib*: Hibernate libraries and dependencies
  - Hibernate and dependencies
    - ant-antlr-1.6.5.jar
    - antlr-2.7.6.jar
    - asm-attrs.jar
    - asm.jar
    - c3p0-0.9.1.jar
    - cglib-2.1.3.jar
    - dom4j-1.6.1.jar
    - ehcache-1.2.3.jar
    - hibernate3.jar
    - jaxen-1.1.1.jar
    - jta.jar
  - Spagic Persistence Layer library
    - metadb-model-{SPAGIC-VERSION}.jar

- *{SERVICEMIX_HOME}\lib\talend*: libraries for Talend integration (these libraries are usually included within Talend jobs)
  - ant.jar
  - axis.jar
  - commons-discovery-0.2.jar
  - db2jcc.jar
  - db2jcc_license_cu.jar
  - edtftpj-1.5.4.jar
  - file_delimited.jar
  - javacsv.jar
  - jaxrpc.jar
  - jconn3.jar
  - jtds-1.2.jar
  - jxl.jar
  - ldapjdk.jar
  - load_product.jar
  - saaj.jar
  - systemRoutines.jar
  - userRoutines.jar

- *{SERVICEMIX_HOME}\lib\optional*: libraries used by the lightweight components.
  - commons-collections-3.1.jar
  - commons-pool-1.2.jar
  - geronimo-annotation_1.0_spec-1.0.jar (used by servicemix-bean)

3.6 Spagic Service Assemblies

The HTTP BC (*servicemix-http-{SMX_VERSION}-installer.zip*) should be installed before installing Spagic service assemblies.

There are some Spagic Service Assemblies to deploy, to complete the Spagic installation.

The service assemblies are:

- ***spagic-getResources-sa{SPAGIC_VERSION}.zip***: service assembly used by Spagic Studio to retrieve the ServiceMix datasource configuration.
  This SA is built with Spagic Studio and then renamed with the Spagic version.
- ***spagic-restart-sa{SPAGIC_VERSION}.zip***: service assembly used by the Spagic Console to restart the failed processes.
  This SA is built with Spagic Studio and then renamed with the Spagic version.
- ***spagic-bckdyndata-sa{SPAGIC_VERSION}.zip*****:** service assembly used by the Spagic Console to backup historic data from the metadabase, to the backup metadabase.
  This SA is built with Spagic Studio and then renamed with the Spagic version.
- ***spagic-deldyndat-sa{SPAGIC_VERSION}.zip*****:** service assembly used by the Spagic Console to delete dynamical data from the metadabase.
  This SA is built with Spagic Studio and then renamed with the Spagic version.
- ***spagic-autoBckDynData-sa{SPAGIC_VERSION}.zip*****:** service assembly used by the Service Manager to automatically delete/backup dynamical data from the metadabase.
  This SA is built with Spagic Studio and then renamed with the Spagic version.
- ***spagic-configureAutoBck-sa{SPAGIC_VERSION}.zip*****:** service assembly used by the Spagic Console to configure the process spagic-autoBckDynData.
  This SA is built with Spagic Studio and then renamed with the Spagic version.
- ***spagic-signalreceiver-sa{SPAGIC_VERSION}.zip*****:** service assembly used to receive a signal (human task event) outside spagic and send to a waiting process.
  This SA is built with Spagic Studio and then renamed with the Spagic version.
- ***spagic-killProcessInstance-sa{SPAGIC_VERSION}.zip*****:** service assembly used to kill a waiting process instance without resuming it's execution.
  This SA is built with Spagic Studio and then renamed with the Spagic version.
- ***spagic-handleWaitingProcess-sa{SPAGIC_VERSION}.zip:*** service assembly that implements a chron job to control expired waiting process instance.
  This SA is built with Spagic Studio and then renamed with the Spagic version.
- ***spagic-deployer-sa{SPAGIC_VERSION}.zip:*** service assembly for publishing the processes on the MetaDB using a Spagic command line tool.
  This SA is built with Spagic Studio and then renamed with the Spagic version.

3.7 Configuration without LAN

If you want to disable ActiveMQ multicast because you are working not connected to a LAN, change the *activemq.xml* file removing the *discoveryUri* attribute from the <*amq:transportConnector*> tag:

```
<amq:transportConnector uri="tcp://localhost:61616" discoveryUri="multicast://default"/>
```

becomes

```
<amq:transportConnector uri="tcp://localhost:61616"/>
```

And remove also the tag <*amq:networkConnector*>.

3.8 Configuration to improve performance

You have to modify the *servicemix.xml* file:

- Comment the flows differents from sedaFlow;
- Comment JDBCAuditor;
- Comment statistics service.

3.9 Configuring the process to handle expiring waiting process instances

In Spagic 2.2 a mechanism for event handling has been introduced.
One of the new features is to have a “chron” process that with a scheduling logic will start and it automatically resumes, or kills the expired wait process instances.

The scheduling logic is implemented with a Spagic process (spagic-handleWaitingProcess) that has a Quartz binding component as its start component. In Quartz chron job are configured using chron expression:
(http://www.opensymphony.com/quartz/wikidocs/TutorialLesson6.html)

The chron expression for the (spagic-handleWaitingProcess) job must be configured in the file:
{SERVICEMIX_HOME}\resources\properties\spagic-handleWaitingProcess_v_0.properties

This file is a simple property file with the chron expression in quartz format.
Some examples:

- *0 0 12 ? * WED (Every Wednesay at 12 o clok)*
- 0 0/10 * * * ? (Every 10 minute)

3.10 Configuring the process to make automatically delete/backup dynamical data from the metadabase

The possibility to make automatically delete/backup data is implemented by two Spagic processes: *spagic-configureAutoBck* and *spagic-autoBckDynData*.
The *spagic-autoBckDynData* process has a Quartz binding component as its start component: the component is configured using chron expression in the file:
{SERVICEMIX_HOME}\resources\properties\spagic-autoBckDynData_v_0.properties
The configuration is managed by the Spagic Console.

The *spagic-configureAutoBck* process uses the file *{SERVICEMIX_HOME}\resources\properties**spagic.properties*** containing the following parameters:

- expiredTime=3MM //chron expression
- autoBackupDynamicData=true
- autoDeleteDynamicData=false

4 Spagic components

We describe the Spagic components included in the Spagic platform. You should include these components in your ServiceMix environment to extend its base features with the Spagic features.

4.1 Prerequisites

Before installing Spagic components you have to:

- Install the component ServiceMix Bean (*servicemix-bean-{SMX_VERSION}-installer.zip)*. Then you need to extract the jar *called geronimo-annotation_1.0_spec-1.0.jar* in *{SERVICEMIX_HOME}\lib\optional*.

4.2 Utilities Classes

These libraries contain utilities classes used usually by other lightweight or servicemix bean component, and are located in *{SERVICEMIX_HOME}\lib\optional* folder:

These libraries are:

- **spagic-utils-{SPAGIC_VERSION}.jar**: Utilities Classes for working with databases and attachments
- **spagic-event-{SPAGIC_VERSION}.jar**: Marshallers, Filters, some classes to model events, and some utilities for event handling

4.3 Spagic Bean Components

These components are installed in the *{SERVICEMIX_HOME}\lib\optional* folder.

You should retrieve them from the Spagic distribution and copy in your ServiceMix.

The components are:

- **spagic-advanced-jdbc-{SPAGIC_VERSION}.jar**: Advanced JDBC SE. For performing any type of operation on database.
- **spagic-attachment-{SPAGIC_VERSION}.jar**: attachment management SE. Allow to process attachments in several ways
- **spagic-console-out-{SPAGIC_VERSION}.jar**: a servicemix-bean component to write to the console.
- **spagic-empty-bc-{SPAGIC_VERSION}.jar**: empty bc binding components, to let a process to terminate without to do nothing.
- **spagic-manage-{SPAGIC_VERSION}.jar**: Management SE. Used by Spagic Studio to retrieve information about a running ServiceMix.
- **spagic-restart-{SPAGIC_VERSION}.jar**: Restart SE. Used by Spagic Studio to manage process restart.
- **spagic-synchronizer-{SPAGIC_VERSION}.jar**: Synchronizer SE. Allow to build in-out processes using In-Only components.
- **spagic-talend-{SPAGIC_VERSION}.jar**: Talend SE. Allow to call Talend processes.
- **spagic-velocity-{SPAGIC_VERSION}.jar**: used by the *VelocityTemplate* component.
- **spagic-wait-{SPAGIC_VERSION}.jar**: The bean components used to manage the wait signal events to let a process to be in wait status until some external event occurs.

Please note that this components require the patch to deploy servicemix-bean component as plain pojo described in chapter 5.

4.4 Servicemix Lightweight components migrated to Servicemix Bean

Starting from the version 3.3, the lightweight container, and so the old lightweight components are not supported anymore.

Unfortunately not all the new introduced standard jbi components, have the same features of the old lightweight components, and in some case the backward compatibility is not guaranteed, for this reason the best solution was to make a porting of the old lightweight components to servicemix-bean.

The components we've ported are:

- Groovy/Script Component
- XSLT Component
- XSDValidateComponent
- JDBC (The simple one)

The components are packaged in the folder *{SERVICEMIX_HOME}\lib\optional*:

- **smx-bean-components-{SPAGIC_VERSION}.jar**

Please note that these components require the patch to deploy servicemix-bean components as plain POJO as we previously do with the old lightweight components. This patch is included in the files:

- **smx-bean-support-{SPAGIC_VERSION}.jar**
- **smx-bean-common-utils-{SPAGIC_VERSION}.jar**

The problem refers to this issue: https://issues.apache.org/activemq/browse/SM-1745 where there's a detailed description.

4.5 Standard Binding Components

These components are installed in the *{SERVICEMIX_HOME}\hotdeploy* folder.

You should retrieve them from the Spagic distribution and copy in your ServiceMix.

The components are:

- **spagic-tcp-{SPAGIC_VERSION}-installer.zip**: TCP binding component for reading and writing through TCP sockets.
- **spagic-jdbc-{SPAGIC_VERSION}-installer.zip**: JDBC poller component. Allow polling on a database table, sending the new inserted data through normalized messages.

4.6 Standard Service Engines

These components are installed in the *{SERVICEMIX_HOME}\hotdeploy* folder.

You should retrieve them from the Spagic distribution and copy in your ServiceMix.

The components are:

- **spagic-deployer-{SPAGIC_VERSION}-installer.zip**: internal Spagic component for managing the processes deploy.

5 ServiceMix patched components

In this chapter we describe the patched ServiceMix components: the patches are necessary for fixing unresolved bugs, or for extending the components with features necessary to Spagic.

5.1 Standard Service Engines

These components are released in the *{SERVICEMIX_HOME}\hotdeploy* folder.

You should retrieve them from the Spagic distribution and copy in your ServiceMix.

The components are:

1. **servicemix-eip-{SMX_VERSION}-installer.zip.**
   Added more capabilities to SplitAggregator (envelope name, create a unique envelope).
   The timer of aggregators must start always see: https://spago.eng.it/jira/browse/SPAGIC-303

2. **servicemix-quartz-{SMX_VERSION}-installer.zip.**
   Solved this issue: https://issues.apache.org/activemq/browse/SM-1701

3. **servicemix-drools-{SMX_VERSION}-installer.zip.**
   Solved this issue: https://issues.apache.org/activemq/browse/SM-1748
   Added support for XLS files: https://spago.eng.it/jira/browse/SPAGIC-311, https://issues.apache.org/activemq/browse/SM-1751

5.2 Standard Binding Components

These components are released in the *{SERVICEMIX_HOME}\hotdeploy* folder.

You should retrieve them from the Spagic distribution and copy in your ServiceMix.

The components are:

1. **servicemix-http-{SMX_VERSION}-installer.zip.**
   Added the http compression support to http endpoint:
   https://spago.eng.it/jira/browse/SPAGIC-333

2. **servicemix-cxf-bc-{SMX_VERSION}-installer.zip:**
   Solved a lot of issue we have with cxf binding component
   https://spago.eng.it/jira/browse/SPAGIC-334
   https://issues.apache.org/activemq/browse/SM-1734
   https://issues.apache.org/activemq/browse/SM-1711

3. **servicemix-mail-{SMX_VERSION}-installer.zip:** We need to provide a custom marshaller to support an Italian standard called PostaElettronicaCertificata (PEC).
   Also provided the automatic support for the mail subject directly by Spagic Studio, without the need for providing the "org.apache.servicemix.mail.subject" property on the Normalized message. See https://spago.eng.it/jira/browse/SPAGIC-355.

If you need to use the "*freebXML Registry 3.0 (**OMAR**)*" you should install also the patched ServiceMix Shared Libraries (*servicemix-shared-{SMX_VERSION}-installer.zip*). This is not part of the Spagic binary distribution; it's released only as source code.
This patch enforces the creation of the envelope *soap:Header* also when it should be not necessary because it's empty: this allows avoiding some errors when using Omar.

5.3 Deprecated ServiceMix Components

These components are deprecated and not anymore supported by Spagic.
They are released in the *{SERVICEMIX_HOME}\hotdeploy* folder: you should create a new folder *undeployed* and move them within it.
The components are:

- servicemix-lwcontainer-*{SMX_VERSION}*-installer.zip

5.4 Unsupported ServiceMix Components

These components are actually not supported by Spagic.
If you create the service assemblies with Spagic Studio, you cannot use them, so we suggest to move them from the *{SERVICEMIX_HOME}\hotdeploy* folder to the *undeployed* folder.
The components are:

- servicemix-camel-2008.01-installer.zip
- servicemix-cxf-se-2008.01-installer.zip
- servicemix-jsr181-2008.01-installer.zip
- servicemix-osworkflow-2008.01-installer.zip
- servicemix-saxon-2008.01-installer.zip
- servicemix-script-2008.01-installer.zip
- servicemix-scripting-2008.01-installer.zip
- servicemix-truezip-2008.01-installer.zip
- servicemix-validation-2008.01-installer.zip
- servicemix-wsn2005-2008.01-installer.zip
- servicemix-xmpp-2008.01-installer.zip

6 BPEL components

Spagic supports execution and monitoring of BPEL processes through usage of the BPEL Service Engine Apache ODE.
After downloading the Apache ODE JBI SE, and **before installing**, you have to modify it:

1. The first change is necessary to support monitoring: open the *ode.jbi.properties* file within the component and modify the value of the property *ode-jbi.event.listeners*:

   *ode-jbi.event.listeners=org.spagic.monitoring.ode.listeners.**OdeListener***

2. Add to the *lib* folder of the component the following libraries retrieved from the Spagic distribution:

- *spagic-monitor-{SPAGIC_VERSION}.jar*
- *spagic-ode-listener-{SPAGIC_VERSION}.jar*

3. Add to the *META-INF\jbi.xml* file the following libraries:

   <path-element>lib/spagic-monitor-{SPAGIC_VERSION}.jar</path-element>

   <path-element>lib/spagic-ode-listener-{SPAGIC_VERSION}.jar</path-element>

After saving the zip file, it can be deployed in the *hotdeploy* folder.

7 Monitoring components

This section explains how to install the Spagic monitoring components on ServiceMix.

7.1 Prerequisites

The JMS BC (*servicemix-jms-{SMX_VERSION}-installer.zip*) and the ServiceMix Shared Libraries (*servicemix-shared-{SMX_VERSION}-installer.zip*) should be installed before installing Spagic components.

7.2 Components

The components necessary for monitoring the Spagic platform are:

- **Monitoring listener**: this listener monitor all the exchanges handled by ServiceMix and copy them to a queue for further processing.
  The listener code is in the file "**spagic-listener-{SPAGIC_VERSION}.jar**" stored in the *lib/optional* folder.
- **Monitoring process**: this is a Service Assembly composed of a single JMS BC, which listens for the queue populated by the Monitoring listener.

  The JMS BC is the standard JMS component with a custom message processor.

  The process is stored in the file "***spagic-monitorService-sa*{SPAGIC_VERSION}*.zip***" and should be copied in the ServiceMix "*hotdeploy*" folder.

  This SA is built with an Ant script in the Spagic workspace (because actually it's not possible creating it with Spagic Studio).

7.3 Configuration

To configure the monitoring you have to:

- Configure the queue for the listener. This is done by the file "***activemq.xml***":

  ```
  <!-- Spagic: queue for SMX monitor service -->
  <amq:queue id="smxMonitorQueue"  physicalName="org.spagic.monitor.SMXQueue"/>
  ```

  This configuration should be inserted after the tag *<amq:broker>* (**after, not within**).

- Configure the JMS connection factory. This is done in the "***jndi.xml***" file:

  ```
  <!-- Spagic jms connection factory -->
  <entry key="java:comp/env/jms/SpagicJmsConnectionFactory">
    <amq:connectionFactory brokerURL="${activemq.url}" />
  </entry>
  ```

- Configure the listener. This is done in the "***servicemix.xml***" file and it is composed of several steps:
  - Configure the JMS factory for the listener (before the *sm:container* tag):

```
<!-- Spagic: JMS factory for monitor listener -->
<bean id="spagicJmsFactory" class="org.springframework.jndi.JndiObjectFactoryBean">
  <property name="jndiName" value="java:comp/env/jms/SpagicJmsConnectionFactory"/>
</bean>
```

  - Configure the listener itself (in the *sm:container* tag):

```
<sm:listeners>
  <!-- Other listeners -->
  ……

  <!-- Spagic: listener for monitor service -->
  <monitoringListeners:AuditingExchangeASyncListener>
    <property name="connectionFactory" ref="spagicJmsFactory"/>
    <property name="destinationQueue" ref="smxMonitorQueue"/>
  </monitoringListeners:AuditingExchangeASyncListener>
</sm:listeners>
```

- In ServiceMix you should have also the ***hibernate.cfg.xml*** in the "*conf* " folder, and the *metadb-model{SPAGIG-VERSION}.jar* in the "*lib/hib*" folder.

7.4 Datasources

Spagic needs two datasources:

- ***MetaDB Datasources (Mandatory)***: Is the datasource that configure the connection pool with the metadatabase where Spagic store all monitoring informations.
  It must be configured in ***${servicemix.home}/conf/jndi.xml*** with the key ***java:comp/env/jdbc/metadb***

- ***Backup MetaDB (Optional):*** Is the datasource that configure the connection pool with the backup metadb where spagic move historic data using the backup service.
  It must be configured in ***${servicemix.home}/conf/jndi.xml*** with the key ***java:comp/env/jdbc/metadb-bck***

Here an example of the jndi.xml fragment defining the datasources:

```
<entry key="java:comp/env/jdbc/metadb">
        <bean id="metadb-ds"
              class="org.apache.commons.dbcp.BasicDataSource"
              destroy-method="close">
              <property name="driverClassName" value="com.mysql.jdbc.Driver"/>
              <property name="url" value="jdbc:mysql://localhost:3306/spagic"/>
              <property name="username" value="spagic"/>
```

```
            <property name="password" value="spagic"/>
        </bean>
    </entry>

    <entry key="java:comp/env/jdbc/metadb-bck">
        <bean id="metadb-ds"
            class="org.apache.commons.dbcp.BasicDataSource"
            destroy-method="close">
            <property name="driverClassName" value="com.mysql.jdbc.Driver"/>
            <property name="url" value="jdbc:mysql://localhost:3306/spagic-bck"/>
            <property name="username" value="spagic-bck"/>
            <property name="password" value="spagic-bck"/>
        </bean>
    </entry>
```

8 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 Studio Components.doc*: detail document about *Spagic Studio* environment.
2. *Spagic Console.doc*: detail document about *Spagic Console* monitoring application.