

1. Introduction

1. Download These Instructions

If you wish to have an offline copy of these instructions, please [download the PDF version of this Installation Manual by clicking here.](#)

2. What is the relationship between SyMBA and the Object Model?

In the FuGE and SyMBA development processes, the master version of the FuGE structure is the Object Model, written with UML and, using Maven 2 and AndroMDA, translated into a variety of tools. SyMBA uses both the FuGE XSD STK, whose purpose is to create either a plain FuGE XSD or aid in the creation of specific community extensions of the FuGE-OM, and the FuGE Hibernate STK.

There are a number of changes made to the officially-sanctioned FuGE-OM as provided in the main download pages that were performed for both the Hibernate STK and SyMBA. These changes are generally to low-level AndroMDA tags and are required to make sure that access to the database is straightforward and clear. A full list of the changes can be found on the page within this documentation entitled [FuGE-OM Modifications for the Hibernate Implementation](#) .

2. What You'll Need

1. Recommendations

1.1. Operating Systems

SyMBA can be installed on either Windows or Linux, but we highly recommend a Linux system, specifically Ubuntu Gutsy Gibbon and higher.

If you use Ubuntu, it is simple to apt-get the components required. Alternatively, you could create a Virtual Machine and install the latest Ubuntu Server (or Standard) release (8.04, Hardy Heron) on it. This has the advantage that it has postgresql 8.3 and maven 2.0.8 in its package repository. During the installation it even asks whether you want to install a database server.

1.2. Database Choice

We also recommend PostgreSQL or MySQL for the database choice, as these are the only two that have been tested with this STK. Of these, PostgreSQL is the preferred choice as it is the one used by the main developer of this STK.

These instructions were originally and loosely based on those found in the [Credits Section](#), but have been modified to suit the needs of this latest version of the STK.

2. Installing SSH

While the STK does NOT require SSH, if your database server will be running on a different machine from where you have this project installed, you may wish to be able to log on to that remote machine.

- Ubuntu. In Ubuntu, you can just run the following command to install ssh if you don't have it yet on the appropriate two machines:

```
sudo apt-get install ssh
```

- Other OSs. You may already have ssh installed on the appropriate machines. You can use either commercial SSH or the open alternative, OpenSSH (this is what Ubuntu installs using the

above command).

- Windows. There are a couple of options for installing an OpenSSH server on your windows machine, though neither are particularly easy. (This is another reason we recommend running the webapp from a *nix environment.)
 - One option for OpenSSH is to install [OpenSSH for Windows](#) , which installs a minimal cygwin package .
 - Fully install cygwin, and then install OpenSSH within cygwin. We recommend installing cygwin to help you with this task. When running setup.exe during the cygwin installation, ensure that you select "openssh" for installation. This application lives underneath the "Net" section of the cygwin install. There are some useful instructions for setting up SSH via cygwin [here](#) . In short, however, once cygwin is installed and the users set up, you can set up the SSH service with two commands. Firstly, run ssh-host-config , which sets up the server information for SSH. This is an interactive setup program. Secondly, you will want to set up user private and public keys with ssh-user-config . Remember that the user you run ssh-user-config for is the user that you need to provide the details of within the net.sourceforge.symba.webapp.scp.* variables inside the top-level pom.xml.

3. Installing Sun Java 5 or Java 6

SyMBA has been tested on both Sun Java 5 and Java 6, so feel free to use whichever version suits you.

3.1. Install Sun J2SE Development Kit 6.0 (JDK 6.0)

Otherwise, if you are using Ubuntu 7.04 or higher (or similar), you can install Sun Java 6 simply with an apt-get command. The following commands show the installation of the binaries, and of the JDK (the latter is required for Tomcat):

```
sudo apt-get install sun-java6-bin  
sudo apt-get install sun-java6-jdk
```

and then optionally set the default version in your system with

```
sudo update-java-alternatives
```

Full details of the installation of Sun Java on Ubuntu 7.04 or higher is available at <https://help.ubuntu.com/community/Java> .

Alternatively, you can download the latest Java from [Sun directly](#) .

3.2. Install Sun J2SE Development Kit 5.0 (JDK 5.0)

If you are using Ubuntu 7.04 or higher, you can install Sun Java 5 simply with an apt-get command. The following commands show the installation of the binaries, and of the JDK (the latter is required for Tomcat):

```
sudo apt-get install sun-java5-bin  
sudo apt-get install sun-java5-jdk
```

and then optionally set the default version in your system with

```
sudo update-java-alternatives
```

Full details of the installation of Sun Java on Ubuntu 7.04 or higher is available at <https://help.ubuntu.com/community/Java> . These instructions are for java 6, so just replace any instance of the number "6" in the instructions with the number "5".

Alternatively, you can download Java 1.5.x from [Sun directly](#) .

3.3. Check Your Installation

Make sure that the JAVA_HOME environment variable is pointing to the directory where you installed the JDK.

If using the "apt-get install" method for Ubuntu described above combined with the update-java-alternatives command, you should already have this variable set.

4. Download and Install Maven 2.0.7 or Later

Download and install Maven 2.0.7 or later from this site: <http://maven.apache.org/download.html>

5. Maven Setup

Create a directory in your home directory called .m2 with a single file inside called settings.xml . This is what my settings.xml looks like:

```
<settings>
<localRepository>/media/share/synched/Documents/.m2/repository/</localRepository>

<proxies>

<proxy>
<active>>true</active>

<protocol>http</protocol>

<host>my.proxy.host</host>
<port>8080</port>
</proxy>

</proxies>
<mirrors>
<mirror>

<id>ibiblio.org</id>

<name>ibiblio Mirror of http://repo1.maven.org/maven2/
</name>

<url>http://mirrors.ibiblio.org/pub/mirrors/maven2</url>

<mirrorOf>central</mirrorOf>
</mirror>

</mirrors>
</settings>
```

All sections are optional. However, depending on your circumstances, you may wish to use one or more of these settings. The "localRepository" element names a location separate from the default home directory for the Maven2 repository. This may be beneficial if you have limited space on your home directory, as this repository directory can grow quite large. The "proxies" element should only be used for those developers who must access the internet via a proxy. The "mirrors" section is useful to have in case the primary central Maven2 server is offline for some reason.

6. Environment Variables and Settings for Maven

Set up the environment variable `M2_HOME` to point to your maven installation directory, and then ensure that both `$M2_HOME/bin` and `$JAVA_HOME/bin` are present in your `PATH`. Also set `M2_REPO`, which is the location of your Maven2 repository. Not only is it recommended by Maven, it is also a variable used by this project later on.

7. Test Maven (Part One)

Test that you've installed maven correctly by running `mvn --version`. You should see something similar (but not necessarily identical) to the following:

```
$ mvn --version
Maven version: 2.0.7
Java version: 1.5.0_08
OS name: "linux" version: "2.6.17-12-386" arch: "i386"
```

8. Test Maven (Part Two)

Check that Maven2 is working properly by creating a temporary, empty project with the following command:

```
mvn archetype:create -DgroupId=testapp -DartifactId=testapp
```

Check for the `BUILD SUCCESSFUL` message and, once you have received this message, please delete the created `testapp` folder.

9. Introduction

This is the only AndroMDA artifact that we will install explicitly. All other artifacts, such as AndroMDA cartridges, will be automatically downloaded by the Maven2 scripts generated by the plugin. Install the plugin by following the steps below.

10. AndroMDA Installation

10.1. Download and Install

- Click [here](#) to download the the AndroMDA plugin installer.
- Unzip the contents of the installer into your Maven repository at \$M2_REPO (or whatever you have set "localRepository" to be, or by default, it resides in your-home-dir/.m2/repository).
- Verify that the following directory was created (switch the slashes around for Windows):

```
$M2_REPO/org/andromda/maven/plugins/andromda-maven-plugin>>>
```

10.2. Test Installation

- Create a temporary directory, e.g. C:\andromda-temp or \$HOME/andromda-temp .
- Create a file called pom.xml in this directory with the following content:

```
<project>
<modelVersion>4.0.0</modelVersion>
<groupId>samples.test</groupId>
<artifactId>test</artifactId>
<version>1.0</version>
<packaging>jar</packaging>
<name>test</name>
<build>
<defaultGoal>compile</defaultGoal>
<plugins>
<plugin>
<groupId>org.andromda.maven.plugins</groupId>
<artifactId>andromdapp-maven-plugin</artifactId>
<version>3.2</version>
</plugin>
</plugins>
</build>
<repositories>
<repository>
<id>andromda</id>
<name>AndroMDA Repository</name>
<url>http://team.andromda.org/maven2</url>
</repository>
</repositories>
<pluginRepositories>
<pluginRepository>
<id>andromda</id>
<name>AndroMDA Repository</name>
```

```
<url>http://team.andromda.org/maven2</url>  
</pluginRepository>  
</pluginRepositories></project>
```

- Open a Command Prompt in the directory where you created this pom.xml and run the command mvn without any arguments. Make sure the command completes successfully by displaying the BUILD SUCCESSFUL message.
- You can now delete the temporary directory you just created.

11. Choosing a UML Tool

You will probably want a UML tool to use if you wish to manipulate the UML at all. If this is important to you, AndroMDA has been tested with both [ArgoUML](#) and [MagicDraw](#), though the FuGE STK only recommends the MagicDraw line of editors. You must use MagicDraw 15.0 or higher. A free version of the Community Edition of this UML editor is available.

12. Installing a Supported Database

You must choose a database supported by Hibernate. Databases used in the creation of the Hibernate STK include PostgreSQL and MySQL. Hibernate STK has been extensively tested on PostgreSQL and less extensively tested on MySQL. In fact, many users (especially Windows users) have reported problems in the past with using MySQL together with earlier versions of this project on Windows. In these cases, we recommend PostgreSQL.

12.1. PostgreSQL

You can download PostgreSQL from <http://www.postgresql.org/download/>. It is the recommended database. It is not required, but it is recommended that you also install a graphical database browser such as pgAdminIII or PhpPgAdmin, both available from the url above. The JDBC driver should come with the download. Alternatively the PostgreSQL jdbc driver can be downloaded from <http://jdbc.postgresql.org/>

12.2. MySQL

You can download MySQL from <http://dev.mysql.com/downloads/>. phpMyAdmin (http://www.phpmyadmin.net/home_page/downloads.php) is suggested as the GUI for MySQL. If using MySQL, the jdbc driver can be downloaded at: <http://dev.mysql.com/downloads/connector/j/3.1.html>

Instructions for Subversion checkout are available on the [Source Repository](#) information page.

3. Setting Up, Compiling, and Running the Hibernate STK

1. Maven Profiles

SyMBA makes use of two of the profiles generated by AndroMDA within the top-level pom.xml : the default (local) profile and the validation (val) profile. The default profile should contain all of the connection details for your main database, while the validation profile should contain all of the details for your unit testing and general testing database.

In this way, you don't have to constantly be changing the values in your pom.xml when you are switching between a testing environment and your normal environment. This page shows you how to set the values for each of these profiles.

The default profile is identified within the pom.xml with the term "local" and the default activation setting:

```
[...]  
  <profiles>  
    <profile>  
      <id>local</id>  
      <activation>  
        <activeByDefault>true</activeByDefault>  
      </activation>  
    </profile>  
  </profiles>  
[...]
```

The testing profile is identified within the pom.xml with the term "validation" (short name "val"):

```
[...]  
  <profile>  
    <id>validation</id>  
    <activation>  
      <property>  
        <name>env</name>  
        <value>val</value>  
      </property>  
    </activation>  
  </profile>  
[...]
```

By default, no unit tests will be performed when the default (local) profile is in use. If you plan to run the unit tests, then please make both a default and a test database using the steps below.

If you wish to use any of the other two available profiles, then please do. Currently, the STK makes no use of them.

2. Create a database to hold the metadata

2.1. Default (Local) Profile

Create a database inside your chosen DBMS entitled, for example, "fuge". You will also need to have access to a database user that is the owner of this database, e.g create a new user called "fuge" that you use specifically for accessing and modifying the metadata database. Modify the following elements in the top-level pom.xml located at your-checkout/trunk/pom.xml for the profile with the "local" id. This will change the jdbc connection details to a value suitable for your database connection, and also provide the STK with your database username and password:

```
<jdbc.url>  
</jdbc.url>  
  
<jdbc.username></jdbc.username>  
<jdbc.password></jdbc.password>
```

2.2. Testing (Validation) Profile

Follow the same steps as above, but making the changes in the variables within the validation profile rather than the local profile.

3. Point Maven at the correct jdbc jar

3.1. Default (Local) Profile

Modify the following elements in the top-level pom.xml located at your-checkout/trunk/pom.xml for the profile with the "local" id. This will ensure that the STK is pointed at the appropriate location for your jdbc jar and the appropriate class name for your jdbc driver:

```
<jdbc.driver>
</jdbc.driver>

<jdbc.driver.jar>
</jdbc.driver.jar>
```

The values of these elements will depend on what database type you've chosen to use. For example, the value of `jdbc.driver` would be `org.postgresql.Driver` for PostgreSQL, and `com.mysql.jdbc.Driver` for MySQL.

3.2. Testing (Validation) Profile

Follow the same steps as above, but making the changes in the variables within the validation profile rather than the local profile.

4. Filling in SyMBA variables within the top-level pom.xml

While the database holds all experimental metadata, it does not hold the raw experimental data. Instead we chose to keep the raw data separate to reduce the size of the relational database. Create a directory on the computer where you plan to store the data, and then put the details of the machine - and the user that will be the owner of the data - into the fields below, which are present within `your-checkout/trunk/pom.xml`. These elements will be read by the web interface and stored.

Also included in the variables below are a variety of other items that are required within the web interface. Each variable has a comment stating what the variable is used for. The variable set below are represented once per profile within the pom, and therefore you will need to set the variables specifically for each profile you wish to use.

```
<!-- The help email address for your group -->
<net.sourceforge.symba.webapp.helpEmail>helpdesk@cisban.ac.uk</net.sourceforge.symba.
webapp.helpEmail>
  <!-- The name and verison of this software: there are already reasonable defaults, so
you don't need to
change this unless you want to -->
<net.sourceforge.symba.webapp.softwareName>${pom.artifactId}</net.sourceforge.symba.we
bapp.softwareName>
```

```
<net.sourceforge.symba.webapp.softwareVersion>${pom.version}</net.sourceforge.symba.webapp.softwareVersion>
```

<!-- The class name for the jdbc driver for the SyMBA security database. Should be the same

as jdbc.driver variable above, unless you are using a completely different database for the

security database. -->

```
<net.sourceforge.symba.webapp.security.driver>org.postgresql.Driver
```

```
</net.sourceforge.symba.webapp.security.driver>
```

<!-- jdbc connection details for the SyMBA security database -->

```
<net.sourceforge.symba.webapp.security.url>jdbc:postgresql://localhost:5432/symba_security
```

```
</net.sourceforge.symba.webapp.security.url>
```

<!-- The username and password for the SyMBA security database. -->

```
<net.sourceforge.symba.webapp.security.username>user</net.sourceforge.symba.webapp.security.username>
```

```
<net.sourceforge.symba.webapp.security.password>pass</net.sourceforge.symba.webapp.security.password>
```

<!-- The machine name of the remote machine where the data files will be stored -->

```
<net.sourceforge.symba.webapp.scp.hostname>localhost</net.sourceforge.symba.webapp.scp.hostname>
```

<!-- The full path to the actual directory where the files should reside. The data files will be put directly

into this directory. -->

```
<net.sourceforge.symba.webapp.scp.directory>/your/directory/for/data/</net.sourceforge.symba.webapp.scp.directory>
```

<!-- The username and password to use to connect to the remote machine where the data files are stored -->

```
<net.sourceforge.symba.webapp.scp.username>user</net.sourceforge.symba.webapp.scp.username>
```

```
<net.sourceforge.symba.webapp.scp.password>pass</net.sourceforge.symba.webapp.scp.password>
```

<!-- net.sourceforge.symba.webapp.global.remote.data.store.os MUST be either "unix" or "dos" -->

```
<net.sourceforge.symba.webapp.scp.remote.data.store.os>unix
```

```
</net.sourceforge.symba.webapp.scp.remote.data.store.os>
```

<!-- In DOS, you need to replace all of the colons in an LSID. Tell SyMBA want to replace

them with using this variable. This variable is ignored when the scp.remote.data.store.os is set to "unix"

If you don't provide anything at all and are using "dos", the default will be a double underscore, "__" -->

```
<net.sourceforge.symba.webapp.scp.lsid.colon.replacement></net.sourceforge.symba.webapp.scp.lsid.colon.replacement>
```

5. Compilation

5.1. Default (Local) Profile

Change into the top-level trunk/ directory. Generate all of the automatically-generated AndroMDA sources:

```
mvn install
```

You should see a "BUILD SUCCESSFUL" message at the end of it. You must be connected to the internet for this step to work, as there will be many jars that need to be downloaded.

You now have all auto-generated code.

5.2. Testing (Validation) Profile

This is the same as the Default profile, except you have to explicitly state that you want to compile using the Validation profile. Further, during the initial install, you don't want the unit tests to be run, as you want all jars to be made first.

```
mvn -Denv=val -DskipTests install
```

The Validation profile has now been used to make all FuGE Hibernate STK jars.

5.3. Fixing a Bug in the AndroMDA plugin

This step must be run only once, directly after the first time you run the first maven install command. Follow this step and then run the same maven install command you ran the first time to pick up the modified plugin.

When you downloaded and installed the AndroMDA plugin for Maven 2, there was a small bug in one of the jars. (Details of this bug can be found on the [AndroMDA forum](#), and thanks to Olga Tchuvatkina for including the instructions for fixing it.)

Without this fix, the XSDs generated will be virtually empty, containing only the top-level

namespace information. Therefore, the first time you run maven install, the AndroMDA plugin files are automatically downloaded, but with the incorrect xmlschema cartridge. This copy step only needs to happen once.

1. Go into the top-level directory (directly underneath the trunk/ directory) of SyMBA. You'll find a file called andromda-xmlschema-cartridge-3.2.jar there.
2. Copy this into the correct part of your Maven 2 Repository. Below the environment variable \$M2_REPO is used to mark the location of your local Maven 2 Repository. Overwrite the default jar file with the fixed one, making the directory first, if required:

```
mkdir -p $M2_REPO/org/andromda/cartridges/andromda-xmlschema-cartridge/3.2/  
cp andromda-xmlschema-cartridge-3.2.jar  
$M2_REPO/org/andromda/cartridges/andromda-xmlschema-cartridge/3.2/
```

6. Optional Modification of the UML

Once you have run the maven install at least once, and fixed the bug in the AndroMDA plugin as stated above, you can start modifying the UML, if you so wish.

OPTIONAL: If you wish to make changes to the FuGE UML prior to generating the code, the UML may be found in trunk/symba-mds/src/main/uml/SyMBA-FuGE-v1.mdzip . Any changes you make in the UML must be converted from this MagicDraw proprietary file type to a standard UML2 file type.

- Open up MagicDraw 15.0. Before you open any of the FuGE mdzip files, you need to set the maven2.repository path variable. This is equivalent to the \$M2_REPO environment variable you will have already set when installing maven. It points to the location of the andromda 3.2 profiles, which are inside the maven repository. This value should be wherever you have your maven repository: if you installed Maven2 using the default repository location, then maven2.repository would be \$HOME/.m2/repository for linux, or the equivalent in windows.
- If you are having problems, you can manually copy all of the andromda profiles from the maven repository (\$M2_REPO/org/andromda/profiles) into the same directory as the model file, or into the profiles/ directory of the MagicDraw installation location.
- Test that your setup works after setting this variable by opening trunk/symba-mds/src/main/uml/SyMBA-FuGE-v1.mdzip . If it loads without error, you MagicDraw installation is complete.

You can auto-generate the Java IDE project files using Maven 2. The auto-generation step should be performed for the first time after successfully installing and compiling the STK. If you need to update the auto-generated files at any time, you can just re-run the command.

For example, to autogenerate IntelliJ files (), change directory to within the top-level trunk directory and run the following command:

```
mvn idea:idea
```

There are other [similar commands](#) for IDEs such as Eclipse. Details can be found on the [Maven](#) website.

If you are using IntelliJ 7.x or higher, the first time you use the project files, IntelliJ will tell you it needs to upgrade the project files. Just agree to all dialog boxes.

7. Default (Local) Profile

There are many useful AndroMDA-specific maven commands in the trunk/readme.txt generated by AndroMDA, including the commands necessary to create the tables in the database, also shown below. You will need to create the database before using the STK.

```
mvn -f symba-core/pom.xml andromdapp:schema -Dtasks=create
```

If you need to re-create the database from scratch at any time, you can add the drop task to the task list as shown below:

```
mvn -f symba-core/pom.xml andromdapp:schema -Dtasks=drop,create
```

8. Testing (Validation) Profile

If you wish to create the test database (used in the unit tests), then you will need to run this command after having run the [Compiling](#) step for the Validation profile:

```
mvn -f symba-core/pom.xml -Denv=val andromdapp:schema -Dtasks=create
```

9. Assigning Identifiers

By default SyMBA utilizes its own LSID assigner Web Service for assigning new LSIDs to the FuGE objects. There is much more information on using the LSID Web Services (WS) provided by SyMBA in the [LSID section](#) of the [General Information](#) documentation pages.

This installation comes expecting to connect to the LSID Assigning Service available in the

SyMBA Sandbox. You may use this for testing, but there is no guarantee of availability or reliability, and therefore you should really maintain your own version of this war once you are ready to use SyMBA in a more production-style environment. This is straightforward to do, and the instructions are available in the [LSID section](#) mentioned above. Just remember you need an active internet connection to be able to use this WS.

10. Checking the Connection to Your LSID Assigner

To fill the metadata database with either people or template protocols, you need to be able to connect to an active LSID Assigner. When you ran "mvn install", you not only created the `symba.war` file needed for the web interface, but also two Web Service wars: `symba-lsid-ws-assigner.war` and `symba-lsid-ws-resolver.war`. It is the first of these that creates new LSIDs for SyMBA.

You may choose to use your own assigner WS or the one provided for the SyMBA sandbox. If you are using the sandbox WS, there is no guarantee of availability or reliability, and therefore you should really maintain your own version of this WS once you are ready to use SyMBA in a more production-style environment. This is straightforward to do, and the instructions are available in the [LSID section](#).

The SyMBA subversion version of the `symba-mapping` already expects to connect to the Sandbox Assigner WS, so you don't need to change anything to get that to work. However, if you are using your own version of the WS, you will need to remember to modify `client-beans.xml` found in `symba-mapping/src/main/resources`. The address property needs to be updated to the address of your own WS.

You can check your connection to the assigner WS by running the following command from within the `symba-lsid-ws-client` directory:

```
mvn exec:java -
Dexec.mainClass="net.sourceforge.symba.lsid.webservices.client.LsidAssignerClient"
```

The response from the program should look something like (but not necessarily identical to) this:

```
28-May-2008 11:56:00 org.apache.cxf.service.factory.ReflectionServiceFactoryBean
buildServiceFromClass
INFO: Creating Service
{http://service.webservices.lsid.symba.sourceforge.net/}LsidAssignerService from class
net.sourceforge.symba.lsid.webservices.service.LsidAssigner
[...]
Response assignLSID: urn:lsid:cisban.cisbs.org:TestClassName:13b36013-a768-4d2a-8302-
64e1e9952ece
```

The last line is the response from the web service, and contains the test LSID.

11. Preparing Your List of Users

The symba-mapping module contains a file of sample people to load as users in your database (xml/samples/SamplePeople.xml). Open SamplePeople.xml in an editor, and make any modifications/additions you want, based on that template, to add the users you need. You may wish to save this file with a different name, and in a safe place for future reference.

12. Pre-loading Users into the Metadata Database

First, we'll store your users into the FuGE metadata database. You can do this with the following command, replacing the SamplePeople.xml filename with the correct file and path for your own user XML file:

```
mvn exec:java -
Dexec.mainClass="net.sourceforge.symba.mapping.hibernatejxb2.UnmarshalPeople" -
Dexec.args="xml/samples/SamplePeople.xml"
```

13. Pre-loading People into the Login/Security Database

Once you have loaded your people into the SyMBA database, you will also need to tell your symba_security database about those people. Remember, a FuGE Person's Endurant LSID is what goes into the security database in the "lsid" column.

There are two methods of loading usernames and passwords into the security database:

- Automatically. If you would like SyMBA to generate your passwords and usernames for you, there are a number of Java classes provided with main() methods that will create usernames and passwords for you from your user XML file made in the previous step. The first time you ever generate usernames/passwords, run the following command, replacing the SamplePeople.xml filename and path with your own user XML file, if different:

```
mvn exec:java -
Dexec.mainClass="net.sourceforge.symba.webapp.util.security.CreateUserPassAndLoad" -
Dexec.args="xml/samples/SamplePeople.xml"
```

You should double-check that the users have been added to the USERS table by conventional

means (e.g. pgadmin III or psql). Also, save the STDOUT to a safe place, as these are the generated usernames and passwords. If you look in the same package as CreateUserPassAndLoad, you will see a number of other similar classes. CreateUserPass will create the file of usernames and passwords, but won't load in the database.

LoadUserPassIntoDB takes an already-generated user/pass file from one of the two previously-mentioned classes and loads it into the database. You may find these classes useful later on, for maintenance or re-creation of the security database.

- Manually. Using your favorite method of connecting to your database, run the appropriate SQL INSERT statements directly into the USERS table, putting in whatever usernames and passwords you want that match your user XML from the previous step.

14. Pre-loading Protocol Templates into the Metadata Database

There are a number of sample protocol templates available for you to load into your SyMBA installation. You can either start by loading these, or you can create your own based on these templates. This section details how to load up the pre-provided sample template protocols. Change into the symba-mapping directory and run the following command:

```
mvn exec:java -
Dexec.mainClass="net.sourceforge.symba.mapping.hibernatejxb2.UnmarshalWorkflow" -
Dexec.args="xml/samples/SampleMicroarray.xml \
xml/samples/SampleMicroscopy.xml xml/referenceTemplates/CarmenElectrophysiology.xml"
```

This loads all of the currently-extant sample protocols.

15. Further Reference

There are some commands prepared for you in trunk/symba-mapping/README.txt that will help you load sample people and sample protocols into your new database. You can use these samples to guide the creation of your own people and template protocols. These can be used as a handy reference, so you don't have to go back to these docs each time.

4. Known Problems

1. The Chicken and the Egg

Within the FuGE-OM, the AccessRight association (visible within the AuditMain class diagram) links the SecurityAccess entity to the OntologyTerm entity with a 0...1 relationship. You will find within the XmlDbRoundtripTest that any reference to the "accessRight" association end is removed before comparing the original XML file and the XML file that was generated from the database information. This is because of a basic chicken-and-the-egg problem between AuditCollection and OntologyCollection.

This attribute of SecurityAccess links to an item in the OntologyCollection. However, the AuditCollection *must* be loaded in the database before we can proceed to parsing the OntologyCollection, as the OntologyCollection may have Audit Trails that are referenced by the AuditCollection. However, if the ontology term that is referenced by the accessRight association is not yet in the database, that association simply cannot be set within the SecurityAccess class.

In normal use, such ontology terms would already be present in the database, however for a unit test round trip which is generating a new random XML file for each unit test, the ontology term is NOT yet in the database. Hence the chicken-and-the-egg situation. Therefore, this association is purposely removed from the XML file prior to running the comparison of the before and after XML files, so that a known bug won't shield an unknown one.

Any suggestions on resolving this situation would be most appreciated.

2. Linking Table Between Software and Equipment is Not Getting Filled

Within the FuGE-OM, the Software entity is linked to the Equipment entity with a Many2Many relationship. For some reason, the spring-controlled Hibernate session is not properly flushing the session after setting up an association of this type. Because of this, the Software2Equipment table within the database, meant to hold such links, is never populated. This behaviour is shown in the GenericSoftwareMappingHelperTest class, which currently fails.

A full explanation of this error is present in the [Andromda forums](#), where I have yet to receive a response. This part of the code will be fixed as soon as I can figure out how to add the appropriate session.flush() statement.

Any suggestions on resolving this situation would be most appreciated.

5. Testing the STK

This STK uses [TestNG](#) for its test suite. Maven has an integrated test suite plugin called Surefire, which has [Testng support](#) .

There are a number of unit tests already available with this STK, and more are planned. These tests are currently limited to the symba-mapping and symba-querying sub-projects, and can be found within that directory. Tests are disabled by default within the default (local) profile and within the production profile so that production databases are not accidentally modified with test data.

You should run the tests after re-running the Maven install command with the correct profile, thus ensuring that the database connection details are the right ones for the test setup.

To run the tests, simply run these commands from the top-level trunk directory (the maven install command just shows you the appropriate install command for the test commands below it: you do not need to run it before each test, unless you have modified code outside of the module whose tests you are calling):

```
mvn install -Denv=val -DskipTests

mvn -f symba-mapping/pom.xml test -Denv=val
mvn -f symba-querying/pom.xml test -Denv=val
```

All should pass except two (`testSoftware2EquipmentLink` and `testSoftware2EquipmentLinkWithFlush`), which are demonstrating a bug discovered within the auto-generated Hibernate and Spring code. Full details of this bug are available in the [Andromda forums](#) , and as soon as I have a fix, the unit tests should all pass.

6. Tips and Tricks

1. Using the Maven 2 exec:java Command

When running main() methods within SyMBA, you have a variety of choices for how they can be called. You can use java from the command-line, but that requires that you specify the huge classpath that will be needed for that class. You could also run from within your IDE, but then you are using the IDE's classpath, and not the identical one that is created by Maven for the various SyMBA jars.

How, then, can you ensure that you run these Java classes using the same classpath that was used to build SyMBA in the first place? The answer is to use Maven to run your classes, in the same way that you use Maven to build your project. You do this with the mvn exec:java plugin. Details of this plugin are available from the <http://mojo.codehaus.org/exec-maven-plugin/java-mojo.html> Developer's website .

Let's take the example of loading the Sample People into the SyMBA Database. The command should be run from within the symba-mapping/ subdirectory. It is as found in the symba-mapping/README.txt file is:

```
mvn exec:java -
Dexec.mainClass="net.sourceforge.symba.mapping.hibernatejxb2.UnmarshalPeople" -
Dexec.args="xml/samples/SamplePeople.xml"
```

1.1. Specifying the Full Name for exec:java

Sometimes, you may get an error message from Maven stating that it cannot find the exec plugin. This especially seems to happen in Windows systems. In these cases, you can call the plugin using its full name, as in the example below:

```
mvn org.codehaus.mojo:exec-maven-plugin:1.1:java -
Dexec.mainClass="net.sourceforge.symba.mapping.hibernatejxb2.UnmarshalPeople" -
Dexec.args="xml/samples/SamplePeople.xml"
```

If you still have problems, please contact symba-devel@lists.sourceforge.net.

2. Configuring Subversion to Automatically Enable Keywords in New Files

When Subversion keywords are enabled for a particular file under version control, Subversion will automatically update the file with information about the last modifier of the file, the revision number, and more. It can be applied to all sorts of different file types. You can enable keywords in a particular file type one-by-one with commands like this:

```
find . \( -name '.svn' -prune \) -o -name '*.java' -exec svn propset svn:keywords "Date Rev Author HeadURL" {} \;
```

The above command will enable the specified keywords for all *.java files in the directory.

However, if (like me) you constantly are forgetting to run such a command with each svn add of a new file, you can make modifications to your .subversion/config file (which lives in your home directory) so that subversion performs the tagging for you.

Open .subversion/config and un-comment the following line:

```
enable-auto-props = yes
```

Next, add a line (or un-comment the appropriate line if already present) in the [auto-props] section for each file extension you wish to contain svn keywords. An example auto-props section is shown below:

```
*.xml = svn:eol-style=native; svn:keywords=Author Date Id Revision  
*.java = svn:eol-style=native; svn:keywords=Author Date Id Revision  
*.apt = svn:eol-style=native; svn:keywords=Author Date Id Revision
```

If you have any problems, please contact symba-devel@lists.sourceforge.net.

3. Using the Tomcat Manager

One method of re-starting the SyMBA web application on Tomcat is as follows:

- Stop Tomcat
- Delete the old symba.war and associated symba/ directory from Tomcat's webapps/ directory

- Copy the new symba.war into Tomcat's webapps/ directory
- Start Tomcat

However, this involves stopping and starting Tomcat each time you make a change. There is an easier way: using your Tomcat Manager. With the manager, you can re-deploy a war file without stopping the Tomcat server.

The first step is to make a user for Tomcat that has permissions to use the Tomcat Manager. Go into the directory containing your Tomcat installation. Find the conf/tomcat-users.xml file and add a user with the "manager" role. Below is an example (complete) tomcat-users.xml file:

```
<?xml version='1.0' encoding='utf-8'?>
<tomcat-users>
  <role rolename="manager"/>
  <user username="tomcat" password="tomcat" roles="manager"/>
</tomcat-users>
```

Ensure that you've restarted Tomcat to have the changes take effect, and then go to the root URL for your Tomcat installation (e.g. if Tomcat is installed on your local machine on port 8081, then go to <http://localhost:8081>). You should get a welcome to Tomcat screen, which states "If you're seeing this page via a web browser, it means you've setup Tomcat successfully.

Congratulations!". On the left-hand side is a menu. The second item on the menu is "Tomcat Manager". Click on this. The following screen allows you to deploy and undeploy wars to your heart's content, without having to restart Tomcat. However, you need to watch for dragons when using the Tomcat Manager. Details of possible problems with using it are in the next section.

4. "PermGen Space" Error

As described [here](#), when you undeploy a webapp using Tomcat Manager, the classes aren't necessarily garbage collected. This means that, over time, when you undeploy/deploy multiple times in a single Tomcat instance using the manager, you will eventually get an OutOfMemory Exception, with a mention of "PermGen Space". This may happen with a properly-controlled exception, or may show itself with a blank jsp that just doesn't seem to load properly. As SyMBA uses CGLIB, commons-logging, spring and hibernate (all culprits mentioned in the web page below), it will be tough going through with a profiler and finding all instances where it's not being garbage collected. Until there are the resources to do this, there is a simple answer: Just stop and restart your Tomcat Server. This sort of error always happens early on, so you can test after a deploy to see if it is happening. You don't need to re-deploy or anything after such an error, just stop and start Tomcat.

If you have any problems, please contact symba-devel@lists.sourceforge.net.

5. MySQL on Windows: java.sql.SQLException: Can't create/write to file 'C:\WINDOWS\TEMP\#sql_318_0.MYD' (Errcode: 13)

If you are using MySQL on Windows, you may intermittently encounter an exception similar to

```
java.sql.SQLException: Can't create/write to file 'C:\WINDOWS\TEMP\#sql_318_0.MYD'
(Errcode: 13)
```

There is a lot of chatter on the web about such errors, without any real clear answers as to what is happening. You'll get an error like this when you try to load content into the SyMBA database, e.g. via UnmarshalPeople, UnmarshalWorkflow or other similar classes.

There are a variety of thoughts as to the reason. Even though MySQL seems to have the permissions to write to the directory in question (as there are other temp files present there when you look), it seems to have a sudden problem with one of the files, namely the one that is mentioned in the exception. A temporary fix is to delete all of the tmp files in that directory. However, this does NOT solve the problem, and it will just occur again. Various forums have suggested defragging your hard disk, changing permissions in directories, and turning off the virus checking in the directory where MySQL keeps its temp files (the directory mentioned in the exception).

We have not found a satisfactory solution to this problem. The following was tried, but in the particular case was not effective.

- Ensure that you have a specific directory set aside for the MySQL temp files. You can create such a directory by following the instructions [here](#).
- Following the suggestions made [here](#), turn off the virus scanning for that directory.

6. Windows: Escaping Characters

If you have any problems, please contact symba-devel@lists.sourceforge.net.

7. Getting SyMBA Built Behind a Proxy

More here soon!

7.1. cxf.xml

7.2. Maven Settings

7.3. Subversion Settings

7. Would You Like to Contribute?

1. Do you wish to contribute to SyMBA?

If you have enjoyed using this STK, but also have some ideas for improvement, or perhaps wish to include some of your own code in this project, please begin by contacting symba-devel@lists.sourceforge.net. Explain what things you would like to see included, and whether or not you would like to help add them.

If appropriate, the SyMBA Administrators could add you as a developer, allowing you to commit code back to the SyMBA subversion repository. There are some guidelines to follow, however, and please read on to see them.

2. Introduction

This section is for SyMBA developers only: that is, those who are identified on the SourceForge project site as developers or administrators. It offers guidelines in modifying and documenting this STK. You will only be able to modify the website or the code in Subversion if you are a SyMBA developer.

3. What to include in new Java Classes

If you're contributing new Java classes to the STK, please ensure that you have put in useful javadoc for all methods. If you've created a new package or sub-project, you'll also need to check the top-level pom.xml in the reporting element, and ensure that your source code will get picked up by the javadoc plugin when the `mvn site:site` command is run.

Also, please copy the LGPL license section from one of the existing Java classes and include it in your file.

4. Subversion Best-Practices

You can commit changes if one of the SyMBA SourceForge administrators adds you as a SyMBA developer. You will need to contact symba-devel@lists.sourceforge.net if you wish to be added.

- Run "svn update" Every time you start a programming session with the STK, and also just prior to running "svn commit". This will ensure that you resolve any potential conflicts prior to committing changes.

- Please only commit when you have a working STK: do not commit code that will break the build. We need to ensure that anyone who checks out our STK from the trunk/ can at least build the project, even if the trunk is in a snapshot state (which it will normally be in).
- Each time we create a point or full release, it will be tagged and marked in the tags/ subdirectory of the project. This means that users can always choose to stay with a particular release. If we need to bug-fix a particular release separately from what we're already building in the trunk, then we can create a branch for that release in the branches/ subdirectory and work from there. The tags/ subdirectory is never to be committed to, as that would be against Subversion best-practices.

More information on Subversion can be found in its [Manual](#) .

5. Beyond Javadoc: Documenting Your Work

If any of the changes you make change how a user would install or use this STK, please ensure that you update the Hibernate documentation, which we make available via the website both in html and pdf format. The actual documentation is built using the APT (Almost Plain Text) Format and then generated by maven into other formats.

The APT files are made into a book using the doxia maven plugin - this is a plugin that, in a more generic usage, also builds the maven site documentation when you run the command `mvn site:site` . The benefit of using the Doxia book plugin is that it can create the book in multiple versions: currently we build the book in xdoc format, which is used by the `mvn site:site` command, and in pdf, which is manually copied to the FuGE website for download and offline use. Please see the [Doxia Plugin Manual](#) and information on [writing in APT format](#) .

You build the new version of the book by going into the books/ sub-directory and running the book-generation command:

```
cd symba-books/  
mvn doxia:render-books
```

This builds the book in the `symba-books/target/generated-site` directory.

6. Re-Building the SyMBA Website

When you are ready to re-build the entire website, go into the top-level directory and generate the site docs in each of the sub-projects:

mvn site:site

This actually builds all of the html files. However, they are still in their individual sub-directory's target/site directory. To put them all together and copy them to the FuGE website, you need to use the mvn site:deploy command. However, to prevent accidental copying, the shell.sf.net site element is commented out. Please ensure this remains the case when you are committing changes to SVN.

Also edit your \$M2_HOME/settings.xml file by adding the following:

```
<server>
<id>shell.sf.net</id>
<username>your sf username</username>
<password>your sf password</password>
</server>
```

mvn site:deploy JUST copies the already-generated site docs. If you make a change to any of the APT files, you must run mvn site:site FIRST, to re-generate the site docs, THEN mvn site:deploy to copy them to the final location described in the distributionManagement element of the pom.xml

It is a good idea to test the maven-generated site before publishing to the SyMBA website. Change the url element of the local-test site element below to local directory that is right for you, then deploy locally:

```
mvn site:deploy
```

and check the resulting site. If all looks OK, comment-out the local-test site element and un-comment the shell.sf.net site element. Only developers with write-access to the shell.sf.net server and the symba group area will be able to perform this goal. Run the deploy command again to copy to the real SourceForge webserver:

```
mvn site:deploy
```

The only problem with the site deployment at the moment is that it doesn't copy the non-html books properly. This means that the pdf version of the installation instructions doesn't get put on the server. So, as a final step, you need to scp the pdf manually:

```
scp books/target/generated-site/pdf/symba/installation.pdf your-
username@shell.sf.net:/home/groups/s/sy/symba/htdocs/
scp books/target/generated-site/pdf/symba/general-information.pdf your-
username@shell.sf.net:/home/groups/s/sy/symba/htdocs/
scp books/target/generated-site/pdf/symba/deviations-from-fuge.pdf your-
```

username@shell.sf.net:/home/groups/s/sy/symba/htdocs/

Remember, you can only run `site:deploy` successfully if you are down as a SyMBA developer on the SF site.

8. Modifications to the FuGE-OM

1. Modifications to the FuGE-OM for the Hibernate and the EJB3 STK

This document contains the changes to the reference FuGE v1 UML Object model required to properly generate the Hibernate code using the AndroMDA Hibernate+Spring cartridge. It is based extensively on Leandro Hermida's similar document for the FuGE EJB3 STK, so that the two UML documents will be similar. Deviations from his modifications are noted.

- Rename the copied package to net.sourceforge.fuge
- Within the net.sourceforge.fuge package change ALL the packages in the entire FuGE hierarchy to lowercase so that it will later produce JavaBeans standard package names (e.g. ConceptualMolecule --> conceptualmolecule)
- (This step was not needed in the Hibernate STK UML Diagram, as it is already fixed there: Fix typos in UML model, where +referenceAbleCollection between FuGE and ReferenceableCollection should read +referenceableCollection)
- (This step was also not needed within the Hibernate FuGE-OM) Search and remove all AbstractAssociation stereotyped associations (20 of 26 instances) EXCEPT the following (which don't seem to have a concrete partner - am still checking with developers about this) *
 - +dimensionElements : net.sourceforge.fuge::bio::data::DimensionElement
 - +dimensions : net.sourceforge.fuge::bio::data::Dimension
 - +inputPartitions : net.sourceforge.fuge::bio::data::DataPartition
 - +outputPartitions : net.sourceforge.fuge::bio::data::DataPartition
 - +partitionPairs : net.sourceforge.fuge::bio::data::PartitionPair
 - +supportingData : net.sourceforge.fuge::bio::data::Data
- Look through all of the associations in the FuGE model and remove any association name numbers that got added during copying (e.g. Provider1 --> Provider)
 - net.sourceforge.fuge.Provider1
 - net.sourceforge.fuge.bio.data.DimensionElementSet1
 - net.sourceforge.fuge.bio.data.PartitionedData1
 - net.sourceforge.fuge.bio.investigation.Conclusion1
 - net.sourceforge.fuge.bio.investigation.Hypothesis1
 - net.sourceforge.fuge.bio.material.Components1
 - net.sourceforge.fuge.bio.material.MeasuredMaterial1
 - net.sourceforge.fuge.common.protocol.Actions1
 - net.sourceforge.fuge.common.protocol.ChildProtocol1
 - net.sourceforge.fuge.common.protocol.Equipment1
 - net.sourceforge.fuge.common.protocol.EquipmentParts1

- net.sourceforge.fuge.common.protocol.InputCompleteMaterials1
 - net.sourceforge.fuge.common.protocol.ParameterPairs1
 - net.sourceforge.fuge.common.protocol.Parameters1
 - net.sourceforge.fuge.common.protocol.Parameters2
 - net.sourceforge.fuge.common.protocol.Parameters3
 - net.sourceforge.fuge.common.protocol.Parameters4
 - net.sourceforge.fuge.common.protocol.Parameters5
 - net.sourceforge.fuge.common.protocol.Protocol1
 - net.sourceforge.fuge.common.protocol.Software1
 - net.sourceforge.fuge.common.protocol.Software2Equipment1
 - net.sourceforge.fuge.common.protocol.Value1
- All many-to-many associations with two-way navigability MUST have an aggregation (or composition) end
 - (added to end labelled "equipment", though it then appears on the end labelled "software" in the diagram!) Software2Equipment (between GenericSoftware and GenericEquipment) should be "shared" on +equipment end
 - Fix database table name clashes between certain associations and classes using @andromda.persistence.table tagged value

Association	Tagged Value	Location	Reason
Equipment	PROTOCOL_EQUIPMENT	between GenericEquipment and GenericProtocol	clashes with Equipment class
Software	PROTOCOL_SOFTWARE	between GenericSoftware and GenericProtocol	clashes with Software class
Provider	PARAMETERIZABLE_PROVIDER	between Parameterizable and ContactRole	clashes with Provider class

- Create custom table names for classes and associations using @andromda.persistence.table tagged value

Class	Tagged Value	Reason

FuGE	F_U_G_E	instead of FU_G_E
Software2Equipment (between GenericSoftware and GenericEquipment)	SOFTWARE2EQUIPMENT	instead of SOFTWARE2_EQUIPMEN T
Database	EXTERNAL_DATABASE	DATABASE is SQL reserved word

- Certain associations need better/corrected table names using @andromda.persistence.table tagged value

Association	Tagged Value	Location
Actions	PROTOCOL_ACTION	between GenericProtocol and Action
ActionApplications	PROTO_APPL_ACTION_A PPL	between ProtocolApplication and ActionApplication
Affiliations	AFFILIATION	between Person and Organization
AllBibliographicReferences	REF_COLLECT_BIBLIO_R EFERENCE	ReferenceableCollection and BibliographicReference
AllContacts	AUDIT_COLLECT_CONTA CT	between AuditCollection and Contact
AllData	DATA_COLLECT_DATA	between DataCollection and Data
AllDatabases	REF_COLLECT_EXT_DAT ABASE	between ReferenceableCollection and Database
AllDataPartitions	DATA_COLLECT_DATA_P ARTITION	between DataCollection and DataPartition
AllDimensions	DATA_COLLECT_DIMENS ION	between DataCollection and Dimension
AllEquipment	PROTO_COLLECT_EQUIP MENT	between ProtocolCollection and Equipment

AllProtocolApplications	INVEST_COMPONENT_P ROTO_APPL	between InvestigationComponent and ProtocolApplication
AllSequenceAnnotations	CONCEP_MOL_COLLECT _SEQ_ANNOT	between ConceptualMoleculeCollecti on and SequenceAnnotationSet
AllSoftware	PROTO_COLLECT_SOFT WARE	between ProtocolCollection and Software
Annotations	DESCRIBABLE_ANNOTAT ION	between Describable and OntologyTerm
BibliographicReferences	IDENT_BIBLIO_REFEREN CE	Identifiable and BibliographicReference
Characteristics	MATERIAL_CHARACTERI STIC	between Material and OntologyTerm
Components	MATERIAL_COMPONENT	between GenericMaterial and GenericMaterial
ComponentDesignTypes	COMPONENT_DESIGN_T YPE	between InvestigationComponent and OntologyTerm
ConceptualMolecules	CONCEP_MOL_COLLECT _CONCEP_MOL	between ConceptualMoleculeCollecti on and ConceptualMolecule
Contacts	MATERIAL_CONTACT	between Material and ContactRole
DatabaseReferences	IDENT_DB_REFERENCE	between Identifiable and DatabaseReference

DataPartitions	FACTOR_VALUE_DATA_PARTITION	between FactorValue and DataPartition
Descriptions	DESCRIBABLE_DESCRIPTION	between Describable and Description
Dimensions	DATA_DIMENSION	between Data and Dimension
DimensionElements	DIMENSION_DIMENSION_ELEMENT	between Dimension and DimensionElement
EquipmentApplications	PROTO_APPL_EQUIP_APPL	between ProtocolApplication and EquipmentApplication
EquipmentParts	EQUIPMENT_PART	between GenericEquipment and GenericEquipment
Factors	INVEST_COMPONENT_FACTOR	between InvestigationComponent and Factor
FactorCollection	INVEST_COLLECT_FACTOR	between InvestigationCollection and Factor
FactorValues	FACTOR_FACTOR_VALUE	between Factor and FactorValue
HigherLevelAnalyses	DATA_COLLECT_HIGHER_LEV_ANAL	between DataCollection and HigherLevelAnalysis
InputCompleteMaterials	INPUT_COMPLETE_MATERIAL	between GenericProtocolApplication and Material

InputMaterials	INPUT_MATERIAL	between GenericProtocolApplication and GenericMaterialMeasurement
InputPartitions	INPUT_PARTITION	between PartitionPair and DataPartition
InputTypes	INPUT_TYPE	between Protocol and OntologyTerm
Investigations	INVEST_COLLECT_INVESTIGATION	between InvestigationCollection and Investigation
InvestigationComponents	INVEST_INVEST_COMPONENT	between Investigation and InvestigationComponent
InvestigationTypes	INVESTIGATION_TYPE	between Investigation and OntologyTerm
Materials	MATERIAL_COLLECT_MATERIAL	between MaterialCollection and Material
Members	SECURITY_GROUP_MEMBER	between SecurityGroup and Contact
OntologySources	ONTOLOGY_COLLECT_SOURCE	between OntologyCollection and OntologySource
OntologyTerms	ONTOLOGY_COLLECT_TERM	between OntologyCollection and OntologyTerm

OutputMaterials	OUTPUT_MATERIAL	between GenericProtocolApplication and Material
OutputPartitions	OUTPUT_PARTITION	between PartitionPair and DataPartition
OutputTypes	OUTPUT_TYPE	between Protocol and OntologyTerm
Owners	SECURITY_OWNER	between Security and Contact
Parameters	EQUIPMENT_PARAMETER	between GenericEquipment and GenericParameter
Parameters	SOFTWARE_PARAMETER	between GenericSoftware and GenericParameter
Parameters	PROTOCOL_PARAMETER	between GenericProtocol and GenericParameter
Parameters	ACTION_PARAMETER	between GenericAction and GenericParameter
ParameterPairs	ACTION_PARAMETER_PAIR	between GenericAction and ParameterPair
ParameterValues	PARAM_APPL_PARAMETER_VALUE	between ParameterizableApplication and ParameterValue
PartitionPairs	PROTO_APPL_PARTITION_PAIR	between ProtocolApplication and PartitionPair

Performers	PROTO_APPL_PERFORMER	between ProtocolApplication and ContactRole
Properties	ONTOLOGY_INDIVIDUAL_PROPERTY	between OntologyIndividual and OntologyProperty
PropertySets	DESCRIBABLE_PROPERTY_SET	between Describable and NameValueType
Protocols	PROTO_COLLECT_PROTOCOL	between ProtocolCollection and Protocol
ProtocolApplications	PROTO_COLLECT_PROTOCOL_APPL	between ProtocolCollection and ProtocolApplication
Providers	INVESTIGATION_PROVIDER	between Investigation and ContactRole
QualityControlStatistics	QUALITY_CONTROL_STATISTIC	between Material and OntologyTerm
RangeDescriptors	RANGE_DESCRIPTOR	between Range and OntologyTerm
SecurityGroups	AUDIT_COLLECT_SECURITY_GROUP	between AuditCollection and SecurityGroup
SecurityRights	SECURITY_RIGHT	between Security and SecurityAccess
SoftwareApplications	PROTO_APPL_SOFTWARE_APPL	between ProtocolApplication and SoftwareApplication

SourceMaterials	SOURCE_MATERIAL	between Investigation and Material
SummaryResults	SUMMARY_RESULT	between Investigation and HigherLevelAnalysis
Types	PARAMETERIZABLE_TYPE	between Parameterizable and OntologyTerm
Types	SEQUENCE_ANNOTATION_SET_TYPE	between SequenceAnnotationSet and OntologyTerm

- Certain associations ends need better/corrected column names using @andromda.persistence.column tagged value

Association End	Tagged Value	Location	Reason
+parameters	PARAMETER_ID	Parameters association between GenericAction and GenericParameter	override plural association end name
+parameterPairs	PARAMETER_PAIR_ID	ParameterPairs association between GenericAction and ParameterPair	override plural association end name
+affiliations	AFFILIATION_ID	Affiliations association between Person and Organization	override plural association end name
+allContacts	CONTACT_ID	AllContacts association between AuditCollection and Contact	override plural association end name
+securityGroups	SECURITY_GROUP_ID	SecurityGroups association between AuditCollection and SecurityGroup	override plural association end name
+componentDesignTypes	COMPONENT_DESIGN_TYPE_ID	ComponentDesignTypes association between InvestigationComponent and OntologyTerm	override plural association end name

+conceptualMolecules	CONCEPTUAL_MOLECULE_ID	ConceptualMolecules association between ConceptualMoleculeCollection and ConceptualMolecule	override plural association end name
+allSequenceAnnotations	SEQUENCE_ANNOTATION_ID	AllSequenceAnnotations association between ConceptualMoleculeCollection and SequenceAnnotationSet	override plural association end name
+allData	DATA_ID	AllData association between DataCollection and Data	override plural association end name
+allDataPartitions	DATA_PARTITION_ID	AllDataPartitions association between DataCollection and DataPartition	override plural association end name
+allDimensions	DIMENSION_ID	AllDimensions association between DataCollection and Dimension	override plural association end name
+higherLevelAnalyses	HIGHER_LEVEL_ANALYSIS_ID	HigherLevelAnalyses association between DataCollection and HigherLevelAnalyses	override plural association end name

+dimensions	DIMENSION_ID	Dimensions association between Data and Dimension	override plural association end name
+annotations	ANNOTATION_ID	Annotations association between Describable and OntologyTerm	override plural association end name
+descriptions	DESCRIPTION_ID	Descriptions association between Describable and Description	override plural association end name
+propertySets	PROPERTY_SET_ID	PropertySets association between Describable and NameValueType	override plural association end name
+dimensionElements	DIMENSION_ELEMENT_ID	DimensionElements association between Dimension and DimensionElement	override plural association end name
+parameters	PARAMETER_ID	Parameters association between GenericEquipment and GenericParameter	override plural association end name
+equipmentParts	EQUIPMENT_PART_ID	EquipmentParts association between GenericEquipment and GenericEquipment	override plural association end name

+factorCollection	FACTOR_ID	FactorCollection association between InvestigationCollection and Factor	override plural association end name
+factorValues	FACTOR_VALUE_ID	FactorValues association between Factor and FactorValue	override plural association end name
+dataPartitions	DATA_PARTITION_ID	DataPartitions association between FactorValue and DataPartition	override plural association end name
+bibliographicReferences	BIBLIOGRAPHIC_REFERENCE_ID	BibliographicReferences association between Identifiable and BibliographicReference	override plural association end name
+databaseReferences	DATABASE_REFERENCE_ID	DatabaseReferences association between Identifiable and DatabaseReference	override plural association end name
+inputCompleteMaterials	INPUT_COMPLETE_MATERIAL_ID	InputCompleteMaterials association between GenericProtocolApplication and Material	override plural association end name

+inputMaterials	INPUT_MATERIAL_ID	InputMaterials association between GenericProtocolApplication and GenericMaterialMeasurement	override plural association end name
+inputPartitions	INPUT_PARTITION_ID	InputPartitions association between PartitionPair and DataPartition	override plural association end name
+inputTypes	INPUT_TYPE_ID	InputTypes association between Protocol and OntologyTerm	override plural association end name
+providers	PROVIDER_ID	Providers association between Investigation and ContactRole	override plural association end name
+investigationTypes	INVESTIGATION_TYPE_ID	InvestigationTypes association between Investigation and OntologyTerm	override plural association end name
+investigations	INVESTIGATION_ID	Investigations association between InvestigationCollection and Investigations	override plural association end name

+factors	FACTOR_ID	Factors association between InvestigationComponent and Factor	override plural association end name
+allProtocolApplications	PROTOCOL_APPLICATION_ID	AllProtocolApplications association between InvestigationComponent and ProtocolApplication	override plural association end name
+investigationComponents	INVESTIGATION_COMPONENT_ID	InvestigationComponents association between Investigation and InvestigationComponent	override plural association end name
+characteristics	CHARACTERISTIC_ID	Characteristics association between Material and OntologyTerm	override plural association end name
+materials	MATERIAL_ID	Materials association between MaterialCollection and Material	override plural association end name
+components	COMPONENT_ID	Components association between GenericMaterial and GenericMaterial	override plural association end name
+contacts	CONTACT_ID	Contacts association between Material and ContactRole	override plural association end name

+sources	SOURCE_ID	Sources association between OntologyCollection and OntologySource	override plural association end name
+ontologyTerms	ONTOLOGY_TERM_ID	OntologyTerms association between OntologyCollection and OntologyTerm	override plural association end name
+properties	PROPERTY_ID	Properties association between OntologyIndividual and OntologyProperty	override plural association end name
+outputMaterials	OUTPUT_MATERIAL_ID	OutputMaterials association between GenericProtocolApplication and Material	override plural association end name
+outputPartitions	OUTPUT_PARTITION_ID	OutputPartitions association between PartitionPair and DataPartition	override plural association end name
+outputTypes	OUTPUT_TYPE_ID	OutputTypes association between Protocol and OntologyTerm	override plural association end name

+types	TYPE_ID	Types association between Parameterizable and OntologyTerm	override plural association end name
+parameterValues	PARAMETER_VALUE_ID	ParameterValues association between ParameterizableApplication and ParameterValue	override plural association end name
+actions	ACTION_ID	Actions association between GenericProtocol and Action	override plural association end name
+parameters	PARAMETER_ID	Parameters association between GenericProtocol and GenericParameter	override plural association end name
+actionApplications	ACTION_APPLICATION_ID	ActionApplications between ProtocolApplication and ActionApplication	override plural association end name
+equipmentApplications	EQUIPMENT_APPLICATION_ID	EquipmentApplications between ProtocolApplication and EquipmentApplication	override plural association end name

+partitionPairs	PARTITION_PAIR_ID	PartitionPairs association between ProtocolApplication and PartitionPair	override plural association end name
+performers	PERFORMER_ID	Performers association between ProtocolApplication and ContactRole	override plural association end name
+softwareApplications	SOFTWARE_APPLICATION_ID	SoftwareApplications association between ProtocolApplication and SoftwareApplications	override plural association end name
+allEquipment	EQUIPMENT_ID	AllEquipment association between ProtocolCollection and Equipment	override plural association end name
+protocols	PROTOCOL_ID	Protocols association between ProtocolCollection and Protocol	override plural association end name
+protocolApplications	PROTOCOL_APPLICATION_ID	ProtocolApplications association between ProtocolCollection and ProtocolApplications	override plural association end name

+allSoftwares	SOFTWARE_ID	AllSoftwares association between ProtocolCollection and Software	override plural association end name
+qualityControlStatistics	QUALITY_CONTROL_STATISTIC_ID	QualityControlStatistics association between Material and OntologyTerm	override plural association end name
+rangeDescriptors	RANGE_DESCRIPTOR_ID	RangeDescriptor association between Range and OntologyTerm	override plural association end name
+allBibliographicReferences	BIBLIOGRAPHIC_REFERENCE_ID	AllBibliographicReferences association between ReferenceableCollection and BibliographicReference	override plural association end name
+allDatabases	DATABASE_ID	AllDatabases association between ReferenceableCollection and Database	override plural association end name
+members	MEMBER_ID	Members association between SecurityGroup and Contact	override plural association end name
+owners	OWNER_ID	Owners association between Security and Contact	override plural association end name

+securityRights	SECURITY_RIGHT_ID	SecurityRights association between Security and SecurityAccess	override plural association end name
+types	TYPE_ID	Types association between SequenceAnnotationSet and OntologyTerm	override plural association end name
+parameters	PARAMETER_ID	Parameters association between GenericSoftware and GenericParameter	override plural association end name
+sourceMaterials	SOURCE_MATERIAL_ID	SourceMaterials association between Investigation and Material	override plural association end name
+summaryResults	SUMMARY_RESULT_ID	SummaryResults association between Investigation and HigherLevelAnalyses	override plural association end name

- datatype::Object[] within GenericInternalData.storage has been changed to datatype::Blob. Some mapping of this type is required to temporarily fix the inability of PostgreSQL to deal with Object arrays, as there is no default mapping.

2. Hibernate-only FuGE-OM Modifications

These modifications were only performed on the Hibernate STK's version of the FuGE-OM.

2.1. Changes to the standard PostgreSQL Mapping

There is a file, PostgreSQLExtension.xml, within symba-mda/src/main/uml/config/mappings, which specifies any additional mappings between UML data types and Postgres types. Below are a summary of the contents of the file:

- datatype::Date is mapped to TIMESTAMP WITH TIME ZONE\ I originally tried to map datatype::URI to CHARACTER VARYING(1024) (this datatype::URI mapping has been created to force URIs to be stored as a string, as there is no default mapping). However, this didn't work as it was still being passed as a java.net.URI and ultimately caused exceptions that looked a bit like: "org.springframework.orm.hibernate3.HibernateSystemException: could not deserialize; nested exception is org.hibernate.type.SerializationException: could not deserialize at net.sourceforge.fuge.service.EntityServiceBase.getIdentifiable(EntityServiceBase.java:1113)". Therefore I changed both the PostgreSQL and Java mapping so that both in the database and in Java, the datatype::URI is mapped to CHARACTER VARYING(1024) (for PostgreSQL) and java.lang.String (for Java). This isn't ideal, but until I can figure out a way to have the URI dealt with correctly, at least it doesn't cause problems. This meant that the following URIs are actually strings within the API, though the UML has not changed:
 - net.sourceforge.fuge.common.references.Database.URI
 - net.sourceforge.fuge.common.description.Uri.location
 - net.sourceforge.fuge.bio.data.ExternalData.location
 - net.sourceforge.fuge.common.ontology.OntologySource.ontologyURI

2.2. Changes to column attributes

- Had to set the Unique stereotype for Identifiable.identifier, as this hadn't been done, and without it we cannot retrieve objects based on their identifier.

2.3. Changes to entity names

- The "URI" entity is problematic for Spring, as any class name whose second letter is upper case can cause problems. The best way to solve it is to change the name of this entity to "Uri".

2.4. Changes to entity column/property names

- Certain property names of entities needed to be changed because they were PostgreSQL reserved words. This can be done with the `@andromda.persistence.column` tagged value.

Property/Column Name	Tagged Value	Comments
+end (hibernate implementation only)	END_DATE	It is the +end property of the Investigation entity
+end (hibernate implementation only)	END_POSITION	It is the +end property of the Sequence entity

2.5. Changes from lazy to eager loading

- The association between AuditCollection and SecurityGroup is composition, but it is still lazily fetched. This is perhaps because it is a 0...* association, and can therefore have more than one security group? In any case, have set UML to be eager fetched. This means setting `@andromda.hibernate.lazy` to false within the tag for the association end "securityGroups".
- In the same way, the following association ends were set to `@andromda.hibernate.lazy` to false:
 - association end (AE): "securityGroups" (the first instance above)
 - AE: "securityCollection"
 - AE: "securityRights"
 - AE: "annotations" for Describable
 - AE: bibliographicReferences for Identifiable
 - AE: parent for Organization
 - AE: affiliations for Person
 - AE: members for SecurityGroups
 - AE: outputMaterials, inputData, inputCompleteMaterials, outputData for GenericProtocolApplication
 - AE: performer for Audit
 - AE: equipment for GenericSoftware
 - AE: equipment for GenericProtocol
 - AE: software for GenericProtocol

- AE: inputTypes, outputTypes for Protocol
- AE: contact for ContactRole
- AE: ontologySource for OntologyTerm
- AE: rangeDescriptors for Range
- AE: measuredMaterial for GenericMaterialMeasurement
- AE: childProtocolApplication for ActionApplication

2.6. Change to Cardinalities

The back cardinality (the side with the diamond in it) of GenericAction to Protocol (named childProtocol) should be 0...* rather than 0...1 . It doesn't affect the XSD because these aren't checked but will affect the database. This is something that probably should be fixed in the UML.

9. Example Java Code Provided With the original Hibernate STK

The mapping between the JAXB2 classes and persistence classes generated by the cartridges must currently be done manually. This STK provides the handwritten mapping code between the JAXB2 classes and the Hibernate cartridge-generated POJOs.

The code that performs this mapping is present within `net.sourceforge.symba.mapping.hibernatejaxb2` package inside the `symba-mapping` sub-project. It is [tested](#) with a round trip from the XML to the database and back again.

The `DatabaseObjectHelper` class contains some convenience methods to help access to the database services. These are described below:

- `getOrCreate` : Use when you're not sure if your object is already in the database. It will either retrieve your object or create a new one. Checks to see if the identifier provided is already in the database. If it is, return the associated object. If it isn't, then create the object with the provided identifier and return it. If no identifier was provided, then create a fresh identifier and return the new object. This does NOT create any object in the database.
- `assignAndSave` : Should only be used if you want to re-assign an identifier to an already existing object and then load that object into the database, as it does NOT create a new object but assumes a pre-existing one. You may pass a `Person` object as the person who should be marked as the audit trail owner. This person must already be loaded in the database, if used. However, the method will deal properly with null values in the person argument, therefore if you don't have person information, just pass a null value for that argument and the audit information will be created without it.

There are various helper classes available to help you load and unload XML into the database. These classes make use of the `net.sourceforge.symba.mapping.hibernatejaxb2.helper` mapping classes.

1. `PeopleUnmarshaler` (Example Main in `UnmarshalPeople`)

Allows you to load the contents of a FuGE-ML `AuditCollection` into the database. This is useful if you wish to pre-fill the database with the members of an organization or group.

2. `XMLUnmarshaler` (Example Main in `UnmarshalXML` and `XMLRoundtrip`)

Allows you to load the contents of a full FuGE-ML file into the database.

3. XMLMarshaler (Example Main in MarshalXML and XMLRoundtrip)

Allows you to extract a full FuGE experiment from the database and write it out in FugE-ML.

The GenerateOntologyIndividuals class within the net.sourceforge.symba.mapping.hibernatejaxb2 package allows the user to generate quickly a FuGE-ML OntologyCollection file that can then be loaded into the database as part of a FuGE-ML file.

The input file is of the type:

```
accession1::name1  
accession2::name2  
[...]  
accessionN::nameN
```

And you'll get out some FuGE-ML to load into your database. The ontologySource URI and the URN namespace are hard-coded right now, so you will need to modify that to suit your needs.

10. Further Reading

1. Contributors to Documentation

Special thanks goes to Frank Gibson, Rainer Scoepf, and Joerg Servos for helping find bugs in and suggesting additions to the installation docs.

2. Resources used in the creation of this document

1

<http://wiki.ficcs.org/ficcs/FuGE-to-XSD> , accessed 6 August 2007.

2

http://galaxy.andromda.org/index.php?option=com_content&task=view&id=105&Itemid=89 , accessed 6 August 2007.

3. FuGE

- : FuGE website
- : FuGE Wiki
- [FuGE paper in Nature Biotechnology](#)

4. Subversion

- : Subversion Documentation

5. Maven 2

- : Maven Documentation