GUS Installation Guide

Michael Saffitz

GUS Installation Guide

Michael Saffitz

Table of Contents

List of Examples

Chapter 1. GUS Overview

GUS Overview

The Genomics Unified Schema (GUS) is an extensive relational database schema and associated application framework designed to store, integrate, analyze and present functional genomics data. The GUS schema supports a wide range of data types including genomics, gene expression, transcript assemblies, proteomics and others. It emphasizes standards-based ontologies and strong-typing.

The GUS Application Framework offers an object-relational layer and a Plugin API used to rapidly create robust data loading programs for diverse data sources. The GUS distribution includes plugins for standard data sources. The GUS Web Development Kit (WDK) is a rich environment for efficiently designing sophisticated query-based websites with little programming required.

Chapter 2. Installing GUS

System Requirements

Hardware

While the GUS Application framework has primarily been developed on Linux, it should in principle run on any UNIX-based operating system, including Mac OS X, provided the software requirements below can be satisfied.

Note

Users have reported difficulty running GUS on Solaris.

As noted below, GUS requires the use of either Oracle or PostgreSQL as a relational database management system (RDBMS). PostgreSQL support is much more recent, and as such, Oracle is recommended whenever practical. For installations with more than 3-5GB of data, Oracle is highly recommended. PostgreSQL, however, is an open source and freely available database system, and thus makes an attractive option for smaller GUS installations, or where Oracle is not economical. The choice of an RDBMS will affect your system and hardware requirements. Refer to you RDBMS's documentation for further details and specific requirements.

GUS's hardware requirements depend largely on the size and complexity of the data you intend to load and analyze using GUS. Most installations will have a separate server for the database management system, for which you should refer to your RDBMS's documentation for requirements. For the GUS Application framework, any modern desktop PC satisfying the above architecture requirements should suffice. A 2.4Ghz Pentium with 512MB of RAM (or equivalent machine) would serve as a good basic machine. For parsing large datafiles, a 3.2Ghz Pentium with as much as 4GB of RAM may be more appropriate. For guidance specific to your situation, refer to the mailing list archives or post a new message to the list.

Software

GUS requires the following third party software packages:

- Perl 5.8.1 or above (www.perl.com [http://www.perl.com])
- Perl Modules: DBI, DBD-Oracle and/or DB-Pg, Parse-Yapp, XML-Simple, and XML-Parser or XML-SAX
- Java 1.4.2 or above, including the SDK (java.sun.com [http://java.sun.com])
- Oracle 9i or above (www.oracle.com [http://www.oracle.com]) *or* PostgresSQL 7.4 or above (www.postgresql.org [http://www.postgresql.org])
- Apache Ant 1.6.2 or above (ant.apache.org [http://ant.apache.org])

Preparing the RDBMS System

Note

If the GUS Schema has already been installed in the RDBMS, you may skip this step and proceed directly to the section called “Downloading GUS”

Once you've satisfied the above software requirements and installed and configured your database system, you should create or identify an existing database to install GUS into. As part of this process, more advanced users of GUS may consider tuning various database settings such as a data block size and memory allocations, as well as custom tablespace creation. Many of these settings will vary greatly depending on your usage of GUS and database system; refer to your RDBMS's documentation for further details and options.

Once the database to hold GUS has been created and configured, you must create or identify a user that the GUS installer will use to install the GUS Schema. When using Oracle, this user should be granted the `create session` and `dba` privileges for the duration of the install (after installation these privileges may be revoked). For PostgreSQL, it is recommended that this user be the owner of the database or be granted all privileges on all objects within the database. For both database systems, the installer will connect as this user, and as this user create the necessary schemata, tables, sequences, indexes, constraints, and the other objects that compose GUS.

Downloading GUS

The most recent release of GUS is available from the GUS Project website (www.gusdb.org [http://www.gusdb.org]) as a single distributable. From time to time, the GUS Project website may also provide less stable development releases of GUS as a single distributable. In addition, the GUS source repository is available for "bleeding edge" releases.

Warning

Since the 3.5 release of GUS, the source repository is no longer the preferred or recommended method for downloading GUS. The source repository contains the most recent additions and changes to GUS, which may be inconsistent with this and other documentation, and which may include changed APIs that could cause data loss.

Once downloaded, the GUS distribution should be moved to the location of your preference, and expanded:

```
$ tar -zxvf gus-x.x.x.tar.gz
```

Note

Throughout this documentation, the dollar (`$`) symbol will be used at the beginning of a line to denote the shell prompt. You should not type the dollar symbol as part of your input.

Configuring and Preparing GUS

Once you have downloaded and expanded the GUS distribution, you must configure your environment. These settings are required for all uses of GUS, and as such is it highly recommended that you add them in such a manner that they be set upon creating a new session (i.e. in your `.bash_profile` file). The following examples are for the Bash shell, and assume you have expanded the GUS distribution in the directory `/home/smith`:

Example 2.1. Configuring the Environment for GUS

```
$ export GUS_HOME=/home/smith/GUS/gus_home
$ export PROJECT_HOME=/home/smith/GUS/project_home
$ export GUS_CONFIG_FILE=/home/smith/GUS/gus.properties
$ export PERL5LIB=$GUS_HOME/lib/perl
$ export PATH=$GUS_HOME/bin:$PROJECT_HOME/install/bin:$PATH
```

Note

Throughout the rest of this document, `$GUS_HOME`, `$PROJECT_HOME`, and `$GUS_CONFIG_FILE` will frequently be used to refer to the locations specified above.

After your environment has been configured for GUS, you must configure the GUS software itself. First, copy the `$PROJECT_HOME/install/gus.config.sample` file to `$PROJECT_HOME/install/gus.config`. Once copied, you must open this file and provide the necessary values:

Example 2.2. Configuring `gus.config` file

```
dbVendor=Oracle_or_Postgres❶
dbiDsn=perl_dbi_string❷
jdbcDsn=jdbc_conection_string
databaseLogin=database_username❸
databasePassword=database_password
userName=gus_username❹
group=gus_group
project=primary_gus_project
perl=path_to_perl_binary❺
md5sum=path_to_md5_or_md5sum_binary❻
```

❶ The database system you are using. This must be either `Oracle` or `Postgres`.

❷ `dbiDsn` is the Perl DBI string for your database. For example, `dbi:Oracle:mygus`.

`jdbcDsn` is the JDBC string for your database. For example, `jdbc:oracle:thin:@myserver:1521:mygus`.

❸ `databaseLogin` and `databasePassword` are the username and password that the software will use to connect to your database.

❹ `username`, `group`, and `project` are used within the GUS system for auditing and permissions purposes. For now, use `dba` as the `userName` and `group` and `Database administration` as the project. See the GUS User's Guide for more information about these settings.

❺ `perl` is the full path to your Perl executable.

❻ `md5sum` is the full path to your `md5sum` or `md5` executable.

Caution

Since GUS 3.5 the `gus.properties`, `install.prop`, `schema.prop` and other various configuration files are automatically created using the values specified in the `gus.config` file. All of these files, except the gus.properties file, will be overwritten by the GUS installer whenever it us run. *The gus.properties file will never be overwritten by the GUS installer. Care should be taken to understand at all times what values are specified in the gus.properties file as it determines the database the GUS system will access.*

Installing GUS

You are now ready to install the GUS Schema and/or Application Framework. The Schema should only be installed once per GUS instance/database and most commonly will be shared by many users (each with their own installation of the Application Framework). To install *both* the Schema and Application Framework, use the following command:

```
$ build GUS install -append -installDBSchema
```

If the Schema has already been installed, you should instead use the following command:

```
$ build GUS install -append
```

The install process should take 5-30 minutes to complete, depending on the speed of your system, your RDBMS, and the specific tasks required for your environment. When complete, you will see the following on your screen:

```
[echo] Installation Complete
```

Note

You may at some point see `[concat] No existing files and no nested text, doing nothing` as part of the install process. This is a normal message and does not indicate that an error has occurred.

Post-Installation Setup

If you installed the GUS Schema in the previous step, you must now address the issues in this section. If you only installed the Application Framework, you may skip this section.

Database Privileges and Roles

The GUS Application Framework includes support for using basic access permissions with a unix-based model of users, groups, and "others" and through the use of "projects". These permissions are not meant as a robust solution to data security, and are easily circumvented though use of direct database access (i.e. `sqlplus` or `psql`) or local modifications of the GUS Application Framework. As such, it is highly recommended that you carefully evaluate and implement a database-level privileges system that makes sense to your configuration. Groups that need the highest levels of security, such as HIPAA compliance, should consider using Oracle with the Virtual Private Database option and/or other RDBMS-based technologies.

As an example, the Computational Biology and Informatics Laboratory (CBIL) uses a privilege system that is based on two roles, a read-only role, `GUS_R`, and a write role, `GUS_W`. `GUS_R` has been granted the `select` privilege on all tables and views in GUS, and `GUS_W` has been granted the `update, insert, delete` privileges on all tables and views and `select` on all sequences in GUS. Individual databases users are then granted the `GUS_R` role and, as appropriate, the `GUS_W` role. For finer control, this model may be extended to individual schemata or sets of tables.

Registering the GUS Application Framework

For auditing purposes, the GUS system requires that the Application Framework and individual plugins be registered within the GUS Schema. To register the Application Framework in the GUS Schema, use the following command:

```
$ ga +meta --commit
```

It is only necessary to run this command once. Upon successfully running the command, you will see a stream of XML displayed on your screen.

Plugins will need to be registered in a similar fashion prior to use. For more information on registering plugins, please refer to the GUS User's Guide.

Creating users, groups, and projects

You may have noticed the `userName`, `group`, and `project` options specified in the `gus.config` file above. These values are used for auditing changes to the database on an individual row level, as well as part of the permissions system built into the GUS Application Framework. Your instance of GUS should now be configured with the standard "DBA" entries. You may at this point wish to create more specific users, groups, and projects so that further work with GUS will be properly tracked.

For more information on creating users, groups, and projects within GUS, please refer to the GUS User's Guide.

Reinstalling the GUS Application Framework

From time to time it may be necessary to reinstall the GUS application framework, either as part of an upgrade, or to propagate changes you've made within the `$PROJECT_HOME` directory to the `$GUS_HOME` directory. If you wish to force all data model objects to be rebuilt (for example, due to a change within the schema), you should first run the command:

```
$ touch $PROJECT_HOME/Schema/gus_schema.xml
```

To reinstall the GUS application framework, simply run the build command:

```
$ build GUS install -append
```