DataTAG
Advance Reservation WP2
Database Management in GARA

Installation and Configuration Manual

Network reservation in Gara
With database
MySQL/MyODBC

Chiara Curti-Emanuela Bertelli

DATA TAG

Advanced Reservation WP2

Database Management in GARA

Chiara Curti-Emanuela Bertelli

11/6/2003

1.1 Introduction

This document would like to constitute a track to follow in order to use Gara with database.
It is supposed that Gara has been already installed and configured on a Linux machine.
The following paragraphs explain at first how to install and configure external software packages (MySQL and MyODBC) and then list the configuration steps necessary to use Gara with the MySQL database through MyODBC driver.
Originally Gara diffserver manager used a library called libglobus_slot_table.la which was linked by the manager at compile time. This library managed the file structures and provided interfaces of type “globus_slot_yyy” that were invoked from the different resource managers to read and write the reservations information on a specific file support.
In order to switch from the file management to the database, simply a new library has been implemented. This new library, called libgara_slot_db_manager.la, is linked at compile time only by network resource manager which did not require any changes in its code.

1.2 Packages and software versions

Here below a list of external packages and software versions is provided:

- Linux RedHat 7.3 operating system
- Globus 2.2
- Gara 1.3.0
  extended version of Gara (based on release 1.2.2) including: slot table management based on MySQL database for network reservations, compiled with gcc 3.2.2, Globus 2.2 provided by VDT 1.8
- MySQL 4.0.13
- MyODBC 3.51.06
- Libiodbc 3.51

1.3 How to retrieve external packages and Gara

1.3.1 MySQL 4.0.13

The MySQL database server is a very popular open source database. Its architecture makes it fast and easy to customize and to deploy. The separation of the core server from the storage engine makes it possible to run with strict transaction control or with ultra-fast

transactionless disk access. In our case the transaction control has been chosen because it proves to be safer than the transaction less case.
A transaction is a sequence of one or more SQL statements that form a logical unit of work. Each SQL statement in the transaction performs a part of a task and only when all statements in the transaction are executed successfully the task is completed.
For this reason is used the InnoDB transaction-safe storage engine which provides commit, rollback, crash recovery and locking capabilities.
The MySQL database server is available without a license fee under the GNU General Public License (GPL).

To download the package the link is:

http://datagrid.in2p3.fr/distribution/external/RPMS

The RPMs that have to be downloaded are:

MySQL-client-4.0.13-0
MySQL-server-4.0.13-0
MySQL-embedded-4.0.13-0
MySQL-shared-compat-4.0.13-0
MySQL-devel-4.0.13-0
MySQL-shared-4.0.13-0

1.3.2 MyODBC 3.51.06

MySQL Connector/ODBC (also known as MyODBC) allows you to connect to a MySQL database server using the ODBC database API. ODBC (Open Database Connectivity) provides a way to access a wide range of databases. It defines a set of function calls, error codes and data types that can be used to develop database independent applications.

To download the package the link is:

http://grid.infn.it/datatag/wp2/

Select TASK 2.3 NETWORK ADVANCE RESERVATION,
SOFTWARE AND DOCUMENTATION, DB GARA 1.3 Binaries,
gara/dist/RPMS → i386 → MyODBC-3.51.06-1.i386.rpm

1.3.3 Libiodbc 3.51

The ODBC Driver Manager is a library that manages communication between application and driver or drivers. It does:

- Resolves Data Source Names (DSN).
- Loading and unloading of the drivers.
- Processes ODBC function calls or passes them to the driver.

A commonly used driver manager is iODBC ODBC Driver Manager for Unix that can be found with the following link.
http://grid.infn.it/datatag/wp2/

1) Select TASK 2.3 NETWORK ADVANCE RESERVATION,
   SOFTWARE AND DOCUMENTATION, DB GARA 1.3 Binaries,
   gara/dist/SRPMS → libiodbc-3.51.0-1.src.rpm

2) Select TASK 2.3 NETWORK ADVANCE RESERVATION,
   SOFTWARE AND DOCUMENTATION, DB GARA 1.3 Binaries,
   gara/dist/SOURCES → libiodbc-3.51.0.tar.gz

3) Select TASK 2.3 NETWORK ADVANCE RESERVATION,
   SOFTWARE AND DOCUMENTATION, DB GARA 1.3 Binaries,
   gara/dist/RPMS → i386 → libiodbc-3.51.0-1.i386.rpm
   → libiodbc-admin-3.51.0-1.i386.rpm
   → libiodbc-devel-3.51.0-1.i386.rpm

1.3.4 Gara

To download Gara it is possible to follow two different ways. The first one concerns the access to the CVS repository. The second way provides the access to the INFN web page as described below.

CVS

It is possible to access Gara repository with the following link:

http://cvs.infn.it/cgi-bin/cvsweb.cgi/ARandCoA/

To download the code you should be allowed through username and password provided by Luca Dell'Agnello (luca.dellagnello@cnaf.infn.it).

If you have already access to the repository, it is necessary to set in a bash shell:

```
export CVSROOT=<your_username>@cvs.infn.it:/usr/local/CVS/ARandCoA
export CVS_RSH=ssh
```

After that it is necessary to use the following command in order to download Gara code from the repository:

```
> cvs co gara
```

Otherwise to download just the last modifications from the repository (to update Gara code already installed):

```
> cvs update
```

INFN web page

With the following link it is possible to reach the INFN web page related to the wp2 of Datatag project:

http://grid.infn.it/datatag/wp2/

In this page software and documentation can be found. It is provided the 1.3.0 version of Gara and some useful RPMs like the libraries MyODBC and Libiodbc.

Some documents are available at the same link: the Database specification and the Test specification. This documentation would provide information about the new version of Gara with the database and the tests that have been already run at CNAF.

Under gara/distr/RPMS/i386/ the following rpms can be downloaded:

db-gara-db-slot-manager-1.3.0-1.i386.rpm

db-gara-dsrt-cpp-admin-1.3.0-1.i386.rpm

db-gara-dsrt-cpp-api-1.3.0-1.i386.rpm

db-gara-dsrt-cpp-client-1.3.0-1.i386.rpm

db-gara-dsrt-java-api-1.3.0-1.i386.rpm

db-gara-dsrt-java-client-1.3.0-1.i386.rpm

db-gara-gara-1.3.0-1.i386.rpm

db-gara-gara-admin-1.3.0-1.i386.rpm

db-gara-gara-devel-1.3.0-1.i386.rpm

db-gara-logging-devel-1.3.0-1.i386.rpm

db-gara-lram-1.3.0-1.i386.rpm

db-gara-lram-devel-1.3.0-1.i386.rpm

db-gara-resource-manager-1.3.0-1.i386.rpm

db-gara-services-1.3.0-1.i386.rpm

Under gara/distr/SOURCES the following files can be downloaded:

db-gara-1.3.0-1.src.rpm → it contains the source files of Gara

db-gara-1.3.0.tar.gz

1.4 Installation

In following paragraphs are suggested the locations on Linux machine where the installation of external software packages can be done.

This type of installation is not mandatory in order to compile and run Gara with database but is the one that has been chosen at CNAF to test the new code.

To install the packages it is necessary to log on in the machine as root.

More details about MySQL and MyODBC installation can be found on MySQL web site (http://www.mysql.com).

The right installation order of the external packages is MySQL, MyODBC and at the end libiodbc.

1.4.1 MySQL

To install the downloaded MySQL RPMs run the command in a shell where the user is root:

```
> rpm -ivh MySQL-server-4.0.13-0.i386.rpm  MySQL-client-4.0.13-0.i386.rpm
           MySQL-devel-4.0.13-0.i386.rpm    MySQL-shared-4.0.13-0.i386.rpm
           MySQL-embedded-4.0.13-0.i386.rpm  MySQL-shared-compat-4.0.13-0.i386.rpm
```

In this way the default install path is /usr, the commands which launch the server and the client are under /usr/bin and the libraries are installed under /usr/lib. The libraries are:
-lmysqlclient
-lmysqlclient_r (thread safe version)

1.4.2 MyODBC

To install the downloaded MyODBC RPM run the command in a shell where the user is root:

```
> rpm -ivh MyODBC-3.51.06-1.i386.rpm
```

In this way the default install path is /usr/local and the libraries are installed under /usr/local/lib. They are:
- lmyodbc3
- lmyodbc3_r (thread safe version)

1.4.3 libiodbc

Move the file libiodbc-3.51.0.tar.gz in an available directory (for example your /home/username directory).

In the selected location it is necessary to unzip and untar the file with the commands:

```
> gunzip libiodbc-3.51.0.tar.gz
> tar -xvf libiodbc-3.51.0.tar
```

In our case only the include files of iODBC driver manager are important for Gara database.

1.4.4 Gara rpms

Due to the fact that only the network resource manager works with the database for the moment, to run and test the reservation process with the information stored into the database the diffserver section has to be installed. The list of the necessary rpms is provided:

db-gara-db-slot-manager-1.3.0-1.i386.rpm

db-gara-dsrt-cpp-api-1.3.0-1.i386.rpm

db-gara-gara-1.3.0-1.i386.rpm

db-gara-gara-admin-1.3.0-1.i386.rpm

db-gara-gara-devel-1.3.0-1.i386.rpm

db-gara-logging-devel-1.3.0-1.i386.rpm

db-gara-lram-1.3.0-1.i386.rpm

db-gara-lram-devel-1.3.0-1.i386.rpm

db-gara-resource-manager-1.3.0-1.i386.rpm

db-gara-services-1.3.0-1.i386.rpm → this rpm contains the end2end and the simple-multi features.

1.5 Slot_db_manager module in Gara

To maintain compatibility with the slot_manager module which handles the file management, a new directory under Gara has been created. Its name is slot_db_manager and its location is under:

path_to_gara_location/gara

where path_to_gara_location is supposed to be the path to reach the directory where Gara has been installed.

Note that tmp is the directory where the updated version of Gara code is supposed to be located after the command 'cvs update' as described in paragraph 1.3.4.
The content of the directory path_to_gara_location /gara is:

```
AUTHORS       acinclude.m4   config.status   logging
COPYING       aclocal.m4     configure       lram
CVS           autogen.sh     configure.in    m4
INSTALL       clean          doc             resource_manager
Makefile      common         dsrt            simple_multi
Makefile.am   config         end2end         slot_db_manager
Makefile.in   config.cache   gara            slot_manager
NEWS          config.h       gara.spec       stamp-h
README        config.h.in    gara.spec.in    stamp-h.in
```

acconfig.h	config.log	libtool

With bold and underlining characters, three directories are shown: slot_manager which handles the storage on file support for CPU and disk reservations, slot_db_manager which contains the database handling for network reservation and the resource_manager directory that contains the code for all types or resource manager.

This module can be compiled and tested as stand alone module or can be compiled and linked together with the diffserver manager code.
In paragraphs 1.9 and 1.10 the way to test the library libgara_slot_db_manager.la is described. In 1.9 paragraph only the stand alone library is tested whereas in paragraph 1.10 is explained the way to test the diffserver manager of Gara which uses the new library with database handling.
In the slot_db_manager module three directories are located:

- etc
- libraries
- tests

In the paragraphs from 1.5.1 to 1.5.3 the new files contained in the directories mentioned above are shortly described.

1.5.1 etc directory

This directory contains the following files:

Makefile	dbgarastart.csh.in	Makefile.am	dbgarastart.sh
Makefile.in	dbgarastart.sh.in	odbc.ini.in	odbc.ini
creategaraschema.sql	tmp.sql	dbtest.sh.in	dbtest.sh

A short description of them is provided here below.

odbc.ini.in and odbc.ini: odbc.ini is the configuration file needed by ODBC driver. Without it it's not possible to use Gara MySQL database through ODBC APIs.
odbc.ini is the output of the odbc.ini.in file after the compiling phase.

dbtest.sh.in and dbtest.sh: dbtest.sh.in is the configuration file necessary to create the dbtest.sh script which launches the test executable garaDBprocess. The dbtest.sh script is the output of dbtest.sh.in after the compiling phase.

dbgarastart.sh.in, dbgarastart.csh.in and dbgarastart.sh: dbgarastart.sh.in is the configuration file necessary to create the dbgarastart.sh script. It drops the old database if exists and creates the new one. It invokes creategaraschema.sql which creates the three tables RESOURCE, SLOT, DIFFSERV in the database. To perform these actions is also managed the file tmp.sql for temporarly use.
In the etc directory the file dbgarastart.csh.in is also present. It is a script for a cshell. The makefile now manages only the *.sh.in version of the script.

1.5.2 libraries directory

This directory contains the following files:

Makefile	Makefile.am	Makefile.in
slot_errors.h	slot_table.c	slot_table.h
slot_table.lo	slot_table.o	garadbutility.c
garadbutility.h	garadbutility.o	garadbutility.lo

slot_table.c and slot_table.h : slot_table.c is the main file of this directory. It contains the procedures of type globus_slot_yyy which constitute the interface between resource manager and slot manager.
In slot_manager module the interface is developed in order to store the information on a file, in the new slot_db_manager module the interface provides the calls to the database.
Slot_table.h is the header file which contains defines and functions prototypes.

slot_errors.h : this header file contains the enumerate which defines the errors of type: GLOBUS_SLOT_MANAGER_ERROR_something.
For database management the following error has been added: GLOBUS_SLOT_MANAGER_GENERIC_DB_ERROR.

garadbutility.c and garadbutility.h: the *.c file contains all the utilities needed to connect to the database, to disconnect, to get the environment variables, to print the information log message from database and to calculate the latetest time of a reservation as requested by some procedures of slot_table.c file.
garadbutility.h is the related header file.

1.5.3 tests directory

This directory contains the following files:

Makefile	Makefile.in	garaDBProcess.o
Makefile.am	garaDBProcess.c	garaDBprocess

A short description:

garaDBProcess.c : this file consists of the main program of test.
It is useful to test slot_db_manager as a stand alone library without the resource manager which invokes it.
The executable is garaDBprocess. In paragraph 1.9 is explained how to use it.

1.6 Gara and Instdb

At the moment the makefile and autogen system is built so as to create a new directory at the same level of gara directory. This directory is called instdb and there are put all the executable and final scripts after the compiling and the installing phase. Instdb contains the subdirectories listed below:

bin etc include lib sbin share

bin: it contains the executable for diffserver manager, diffserver manager test program and garaDBprocess executable. They are copied here after the installing command as described in paragraph 1.7.

etc: among the files that are stored here, there are some configuration files like diffserv_manager.conf that lists information needed by the diffserver manager (number of routers, name of logging file or the IP addresses served by the routers shown in setup_flow.cfg file).
The setup_flow.cfg file contains the configuration needed to setup the router/s as described in the "Administrators Guide to Gara" document that can be retrieve at: http://www-fp.mcs.anl.gov/qos/qos_papers.htm.
In this directory are copied automatically also the files odbc.ini, tmp.sql and creategaraschema.sql.

include: the following header files are copied here → garai_fprintf.h, globus_gara_common.h, logging.h, globus_gara_client.h globus_i_gram_version.h.

lib: here are present libraries of type libglobus_gara_client and liblogging.

sbin: in this directory the scripts dbgarastart, dbtest and setup_flow are copied during the installing phase.

share: it contains the doc directory with the information related to Gara code.

1.7 Changes on Configure and Makefiles

As mentioned above, it possible as first case to compile the slot db library as stand alone in order to test it with the garaDBprocess provided with Gara code.

To build the libgara_slot_db_manager.la library in the stand alone case use the following commands at the level of gara directory (path_to_gara_location /gara):

```
> ./autogen.sh
> ./configure  --enable-slot-db-manager
>  make
```

The second step is the test of diffserver manager that uses the database management.
With the following commands the diffserver manager links and uses the new db library instead of the old one:

```
> ./autogen.sh
> ./configure  --enable-diffserv --prefix=path_to_gara_location /instdb
> make
> make install
```

The “make install” is the command that perform the copy of executable files from the correspondent directories to their locations under instdb.

Regarding the makefile system, the following variables have been added after the insertion of the database library.
From path_to_gara_location/gara/resource_manager:

```
MYSQL_INSTALL_PATH = /usr
MYSQL_LIBS =  -lmysqlclient -lz                    ( --> under /usr/lib)
MYSQL_THR_LIBS =  -lmysqlclient_r -lz -->          ( --> under /usr/lib)
PMYSQL = /usr/bin/mysql                            (path of mysql command)
MYODBC_INSTALL_PATH = /usr/local
MYODBC_LIBS = -L/usr/local/lib -lmyodbc3 -lz                 ( --> under /usr/local/lib)
MYODBC_THR_LIBS = -L/usr/local/lib -lmyodbc3_r -lz           ( --> under /usr/local/lib)
LIBIODBC_CFLAGS = -Ipath_to_libiodbc/libiodbc-3.51.0/include
                                                             (include files of iodbc)
LIBIODBC_INSTALL_PATH = path_to_libiodbc/libiodbc-3.51.0
```

Note: path_to_libiodbc is the path to the directory where is supposed to be located the libiodbc-3.51.0 package after the download.

Before compiling Gara with the new options of configure it is useful to check the variables listed above. Actually they depend on the default location where the external packages have been installed.
For example, if MySQL is not under /usr or MyODBC is not under /usr/local, it is important to changes the makefiles before compiling. Regarding the IODBC include files, it is necessary to specify the exact directory where they are located otherwise the ./configure command fails.
The variables list above is shown only to indicate the way to change the variables having as example the installation done in the tests’machine. The variables will be different depending on the particular performed installation procedure.
Anyway, to update the variable list in the makefile system, it’s necessary to modify only the variables contained in *.m4 files of gara/m4 directory.

Example 1:
MySQL has been installed under /usr/local instead of /usr.

- Go to the path_to_gara_location/gara/m4 directory and modify the mysql.m4 file:

- Change the line
  with_mysql_prefix=${MYSQL_INSTALL_PATH:-/usr}
  and put the correct installation path /usr/local instead of /usr
- The other variables of type MYSQL_something have not to be modified because they are built starting from "with_mysql_prefix" above.

Example 2:
MyODBC has been installed under /usr instead of /usr/local.

- Go to the path_to_gara_location/gara/m4 directory and modify the myodbc.m4 file:
- Change the line
  with_myodbc_prefix=${ MYODBC_INSTALL_PATH:-/usr/local}
  and put the correct installation path /usr instead of /usr/local

The other variables of type MYODBC_something have not to be modified because they are built starting from "with_myodbc_prefix" above.

1.8 Configurations

Before testing the Gara network reservation with the database management, the following configurations have to be applied.
The environment variables listed below have to be set in the same command shell where the tests described in the paragraphs 1.9 and 1.10 run.

1) export DBUSER=root
   → the connection to the database is performed by the superuser 'root'

2) export DBHOST=datatag4
   → put here the name of the machine where the MySQL server runs

3) export DBNAME=gara
   → 'gara' is the database name

4) export LD_LIBRARY_PATH=/usr/lib:/usr/local/lib:$LD_LIBRARY_PATH
   → insert here the path of MySQL libraries (/usr/lib) and of MyODBC libraries (/usr/local/lib)

5) export ODBCINI=path_to_gara_location /instdb/etc/odbc.ini
   → modify the path to the odbc.ini file in the install directory (it will depend on the location where gara code has been downloaded with the 'cvs update' command).

6) export ODBCSYSINI=path_to_gara_location /instdb/etc
   → similar to 5). In performed tests this variable was not necessary but it was suggested by the installation guide retrieved during the MyODBC installation.

Moreover it is also necessary the modification of the odbc.ini.in file located under the path_to_gara_location/gara/slot_db_manager/etc directory. For example some changes are mandatory if the database is called in a different way instead of 'gara' or if the MyODBC libraries are not installed under / MYODBC_INSTALL_PATH/lib.

The changes have to be compatible with the environment variables set in the shell where the tests run (see the list above).
Before the modifications become valid, compiling and installing steps have to be re-applied as described at the beginning of the paragraph 1.7.

Here below an example of a simple odbc.ini.in file is shown.

```
;
;  odbc.ini configuration for MyODBC and MyODBC 3.51 Drivers
;

[ODBC Data Sources]

myodbc3     = MySQL ODBC 3.51 Driver DSN

[myodbc3]
Driver      = @MYODBC_INSTALL_PATH@/lib/libmyodbc3.so
Description  = MySQL ODBC 3.51 Driver DSN
SERVER      = @HOSTNAME@
PORT        =
USER        = root
Password    =
Database    = gara
OPTION      = 3
SOCKET      =
```

1.9 Stand alone test phase

In order to verify whether the database has been correctly inserted inside the Globus environment and the Gara code, it is important to perform a couple of tests.
This is not a test specifications document so the tests which follow have just the aim to verify the success of the installation and configuration procedure described as far as here.
The following steps show a simple way to test the stand alone database module:

1) To start the MySQL server give the command on the shell where the environment variables have been already set:

   > mysqld -u root → 'mysqld' is under /usr/sbin

   Verify that the mysqld daemon is alive through the ps –ef command.
2) Under path_to_gara_location/instdb/sbin run the script dbgarastart in order to create the database;
3) Run the MySQL client with the following command:

   > mysql –u root –h datatag4 → with –h specify the hostname where the server runs.

   > use gara; → specify the name of the database to use.

> show tables; → verify that the tables RESOURCE, SLOT and DIFFSERV have been created

> show columns from RESOURCE; → to verify the fields of this table. Do the same for the other two tables

> select * from RESOURCE; → verify that this table is empty. Do the same for the other two tables

4) Launch the executable path_to_gara_location/instdb/bin/ garaDBprocess.
At first connect to the DB by typing 1 on the command line when the test program has started.
Then the test program asks the user to choose the following action to perform.
It writes on the shell "Choose what to do:".
By typing a number, the correspondent procedure implemented in path_to_gara_location/gara/slot_db_manager/libraries/slot_table.c is tested.
Before running the garaDBprocess test program, see the correspondent garaDBprocess.c file under path_to_gara_location/gara/slot_db_manager/tests in order to see the values that have to be stored into the database.
After each action, check on the database client shell if the expected behaviour is performed (for example after a globus_slot_add action a new slot has to be present in table SLOT and after a globus_slot_bind the table DIFFSERV must contain a new row).
At the end disconnect from database by typing the number 0 which corresponds to the disconnect procedure.

A different way to test the stand alone library is to follow the previous steps from 1) to 3) and then launch the dbtest.sh script under path_to_gara_location/instdb/sbin. In that case the configuration variables described in paragraph 1.8 are set directly by the script and finally the garaDBprocess is launched. If this second way to test is chosen, it is necessary to modify the variables contained in dbtest.sh.ini file under path_to_gara_location/gara /slot_db_manager/etc, then it is necessary to compile and install with the command described in paragraph 1.7 before launching the dbtest.sh script.

1.10 Diffserver test phase

To test if the diffserver manager of Gara can correctly connect to the database and can manage it after the installation procedure described in this manual, it is better to follow the test procedures as shown on the "Administrators Guide to Gara" document (starting from "Configuring Cisco routers" to "Testing the Resource Manager").
The diffserver manager test program can be found under path_to_gara_location/instdb/bin directory.
The steps from 1) to 3) described in 1.9 paragraph of this manual must be applied also for this test.

During the test can happen that the following error causes a failure:
"command not found". Probably the setup_flow script is not found.
As a work around set the PATH environment variable in the same shell where the tests run.
As example:
> export PATH=path_to_gara_location/instdb/sbin:/$PATH

This variable will be updated with the correct path to retrieve the setup_flow script.