ISTITUTO NAZIONALE DI FISICA NUCLEARE

Sezione di Pisa

INFN/TC-05/15
December 22, 2005

USING GNU AUTOTOOLS FOR THE GDMP PACKAGE

Flavia Donno[1], Maria Santa Mennea [2],

1) INFN Pisa, Italy
2) INFN & University Bari, Italy

Abstract

In this document we give a short introduction to the GNU Autotools functionality and we explain their use to manage the GDMP package, pointing out the problems encountered and outlining, where necessary, the need to review the structure of the package.

\

*Published by **SIS–Pubblicazioni***
Laboratori Nazionali di Frascati

1 The GNU Autotools: a short summary

Here we give a short summary of the way the GNU Autotools used to manage the GDMP package work. The GNU Autotools try to simplify the development of portable programs and the building of programs that are distributed as source code. In particular, we describe the tools ***autoconf***, ***automake*** and ***libtool***.

1.1 Autoconf

Autoconf is a tool for producing shell scripts that automatically configure software source code packages to adapt it to many kinds of UNIX-like systems.

The autoconf program produces a configuration shell script, named *configure*, which probes the installer platform for portability related information which is required to customize makefiles, configuration header files, and other application specific files. Then it proceeds to generate customized versions of these files from generic templates. This way, the user will not need to customize these files manually.

To produce a *configure* script, the package manager needs to create a file called *configure.in* which contains invocations of the Autoconf macros that test the system features the package needs or can use. A good start for writing *configure.in* file is given by the **autoscan** program [**3.1**].

1.2 Automake

Automake is a tool for generating *Makefiles* (descriptions of what to build) that conform to a number of standards. Automake substantially simplifies the process of describing the organization of a package and performs additional functions such as dependency tracking between source files.

The automake program produces makefile templates, *Makefile.in* to be used by Autoconf, from a very high level specification stored in a file called *Makefile.am* provided by the user. Each *Makefile.am* is written according to make syntax; Automake recognizes special macro and target names and generates code based on these.

When we deal with pluri-directories projects, such as GDMP, we use the `SUBDIRS` variable in the *Makefile.am* file to list the subdirectories that should be built.

1.3 Libtool

The Libtool tool takes care of all the peculiarities of creating, linking and loading shared and static libraries across a great number of platforms, providing a uniform command line interface to the developer. Using Libtool to manage project libraries, the package manager only need to concern him/herself with Libtool's interface: when someone else builds the project on a platform with different library architecture, Libtool invokes that platform's compiler and linker with the correct environment and command line switches. It installs libraries and library using binaries according to the conventions of the host platform, and follows that platform's rules for library versioning and library interdependencies. Although Libtool is usable by itself, either from the command line or from a non-make driven build system, it is also tightly integrated into Autoconf and Automake adding a few macros to *configure.in* and modifying *Makefile.am*.

Since GDMP is structured in subdirs, one per service, it seems a good idea to create a *convenience* library per service. Convenience libraries are libraries that are not installed in the install directory under lib. They can be combined in a big, exportable package library which can be installed in the lib directory. Package binaries can be built against the

exported package library and installed in the bin directory under the package install directory. Libtool generates a convenience library, one of the three library types that can be built with libtool. Convenience libraries are a portable way of creating such a partialy linked object: Libtool will handle all the low-level details in a way appropriate to the target host.

The key to creating Libtool convenience libraries with Automake is to use the `noinst_LTLIBRARIES` macro [**3.4**].

2 Old GDMP configuration

Before using the Autotools to manage GDMP, the package was structured as follows. Under the top directory, the source code and header files were organized in subdirectories, one per *service*:

Common	Main
ControlMsgs	Misc
DBManager	ReplicaCatalogue
DataMover	RequestManager
FlatFile	ScriptStaging
GDMPCatalog	Security
GDMPFile	StagingPlugins
GDMPHost	Threads
HRMIDL	

For a description of the content of the files in those directories refer to [5]. The ***Main*** directory contains the source code for the executables to be delivered with the package. Other directories in the package are:

bin	tmp
doc	utils
etc	var

In the ***bin*** directory, the package binaries are *installed* after build. In the ***bin*** directory the executable ***gdmp_setup*** is used to create the file ***gdmp_environment***, a shell script that sets up the environmental variables needed to the gdmp server and clients. Instead of using the executable ***gdmp_setup*** to generate the ***gdmp_environment*** file, the Autotools can help generating one automatically starting from a template for the specific gdmp installation.

The ***etc*** directory contains instead *data* files needed only to use the package. In particular in this directory you can find the server certificate, key and proxy.

The ***utils*** directory contains shell script (bash and perl) files useful for package operations. For instance the shell script ***gdmp_server_start*** is a useful template to create the script to be run by the inetd superdaemon in order to launch ***gdmp_server***. This script needs to be customized by the package installer. Also this can be a good example to use the autotools in order to generate an installation specific script, starting from this template. Other scripts in this directory can be used as is.

The ***var*** and ***tmp*** directories are only used during package operations, so just while using the package.

The ***doc*** directory contains the package documentation.

One comment about the structure above is that no distinction is made among directories needed during the *building* process and the directories needed by the *installation* process. Such a distinction is instead very important for package installers and for standard installation tools, such as RPM [6].

Another note is about the absence of defined user headers and library. For this reason the standard [7,8] directories ***lib*** and ***include*** are missing.

In the package top directory one can find the main *Makefile* for the package. Such *Makefile* defines interlibraries dependencies for the package (GDMP at the moment depends on ***Globus***, ***GSINCFTP*** and the ***Globus Replica Catalogue***), the compilers to be used, compilation flags, etc. It also includes a Globus specific **makefile_header** which hardcodes Globus specific flags and redefines internally variables that are used/defined by the Autotools (an example is given by the macros *includedir* and *LIBS*). The Globus makefile_header itself is, in fact, the result of the GNU Autotools used for managing the Globus package. The macro definitions in the Globus makefile_header can collide with the use of autotools for a package like GDMP if certain precautions are not taken. In the same make_header file, also compilation and loader flags are defined.

Other secondary files reside in the top directory of the GDMP package, such as README, a file reporting the CHANGES in the release, some source files for building internal executable tests. All those files should be distributed with the package.

A ***test*** directory containing a set of test executables to be invoked as a check for the various functionalities of the package is missing.

3 A first attempt to use GNU Autotools within GDMP

First of all, using the GNU AutoTools with GDMP, we restructure sligthly the package in order to introduce the concept of user header and library files, and to provide the possibility to specify an installation directory which must be different from the package build tree.

As described above, the Autoconf utility requires the package manager to provide two main files: *configure.in* and *Makefile.am*.

In order to generate *configure.in* file, one can take advantage of the **autoscan** tool that can generate a template as a good starting point.

3.1 Using autoscan

The autoscan program can help create a *configure.in* file. It examines source files in the directory tree. It searches the source files for common portability problems and creates a file *configure.scan* that is a preliminary *configure.in* for that package.

We need to manually examine *configure.scan* before renaming it to *configure.in* and apport some adjustments. The *configure.scan* that has been generated by running **autoscan** for the gdmp package (after deleting the *Makefile* from all directories) is the following:

```
dnl Process this file with autoconf to produce a configure script.
AC_INIT(Common/gdmp_common.C)
dnl Checks for header files.
AC_HEADER_DIRENT
AC_HEADER_STDC
```

```
AC_CHECK_HEADERS(fcntl.h malloc.h sys/file.h unistd.h)

dnl Checks for typedefs, structures, and compiler characteristics.
AC_C_CONST
AC_TYPE_SIZE_T

dnl Checks for library functions.
AC_FUNC_UTIME_NULL
AC_OUTPUT()
```

Running **autoscan** when the *Makefile* file is present in all package directories produces the following result:

```
dnl Process this file with autoconf to produce a configure script.
AC_INIT(Common/gdmp_common.C)

dnl Replace `main' with a function in -lStrn:
AC_CHECK_LIB(Strn, main)
dnl Replace `main' with a function in -lncftp:
AC_CHECK_LIB(ncftp, main)
dnl Replace `main' with a function in -lpthread:
AC_CHECK_LIB(pthread, main)
dnl Replace `main' with a function in -lsio:
AC_CHECK_LIB(sio, main)

dnl Checks for header files.
AC_HEADER_DIRENT
AC_HEADER_STDC
AC_CHECK_HEADERS(fcntl.h malloc.h sys/file.h unistd.h)

dnl Checks for typedefs, structures, and compiler characteristics.
AC_C_CONST
AC_TYPE_SIZE_T

dnl Checks for library functions.
AC_FUNC_UTIME_NULL
AC_CHECK_FUNCS(putenv strdup strtoul)

AC_OUTPUT( Threads/Makefile DataMover/Makefile ControlMsgs/Makefile
RequestManager/Makefile Security/Makefile DBManager/Makefile)
```

The presence of the original *Makefiles* in place when autoscan is run generates a *configure.scan* file with more macro definitions in it, which has been very useful in order to write a more complete *configure.in* file. The macro descriptions are explained in the next paragraph.

3.2 The configure.in file

The following file shows the final *configure.in* file for GDMP package.

```
dnl Process this file with autoconf to produce a configure script.
AC_INIT(Common/gdmp_common.C)

dnl set config options
AM_CONFIG_AUX_DIR(config)
AM_CONFIG_HEADER(config.h)
AM_INIT_AUTOMAKE(gdmp,1.2.2)

dnl Checks for programs.
AC_PROG_CXX

dnl Checks for use of libtool
AM_PROG_LIBTOOL
```

```
dnl Checks for libraries.

dnl Replace `main' with a function in -lpthread:
AC_CHECK_LIB(pthread, main)

dnl Checks for header files.
AC_HEADER_DIRENT
AC_HEADER_STDC
AC_CHECK_HEADERS(fcntl.h malloc.h sys/file.h unistd.h)

dnl Checks for typedefs, structures, and compiler characteristics.
AC_C_CONST
AC_TYPE_SIZE_T

dnl Checks for library functions.
AC_FUNC_UTIME_NULL
AC_CHECK_FUNCS(putenv strdup strtoul)

dnl Get globus install directory from command line option or environment
dnl variable GLOBUS_INSTALL_PATH.
AC_MSG_CHECKING([for globus-install directory])
AC_ARG_WITH(globus-install,[ --with-globus-install=<dir> Default is
\$GLOBUS_INSTALL_PATH],
     globus_install="$withval", globus_install="$GLOBUS_INSTALL_PATH")
if test -d "$globus_install" ; then
     AC_MSG_RESULT([found $globus_install])
     GLOBUS_INSTALL_PATH=$globus_install
else
     AC_MSG_ERROR([no such directory $globus_install])
Fi

dnl Determine globus flavor from command line option.
dnl If command line option is not given, look in
dnl $GLOBUS_INSTALL_PATH/development for a directory with threads (but not
dnl nothreads) in the name.
echo "Attempting to determine globus flavor..."
AC_ARG_WITH(globus-flavor,[ --with-globus-flavor=<dir> Default is
<globusinstall>/development/<directory with threads>],
globus_flavor=$withval, globus_flavor="")
if test -z "$globus_flavor" ; then
     globus_flavor_tmp=`ls -1 $GLOBUS_INSTALL_PATH/development 2>/dev/null |
     grep thread | grep -v nothread`
     globus_flavor=$GLOBUS_INSTALL_PATH/development/`echo
     "$globus_flavor_tmp" | head -n 1`
fi

echo "globus flavor is \"$globus_flavor\""
GLOBUS_FLAVOR=$globus_flavor

dnl We need globus flavor in order to get the globus makefile_header.
dnl Make sure we can find it.
AC_MSG_CHECKING([for makefile_header in globus-flavor])
if test -f "$GLOBUS_FLAVOR/etc/makefile_header" ; then
     AC_MSG_RESULT([yes])
else
     AC_MSG_ERROR([$GLOBUS_FLAVOR/etc/makefile_header not found])
Fi

dnl Get gsnicftp source directory from command line option or environment
dnl variable NCFTP_SOURCE_DIR
AC_MSG_CHECKING([for gsincftp source directory])
AC_ARG_WITH(gsincftp-src,[ --with-gsincftp-src=<dir> Default is
\$NCFTP_SRC_DIR],
ncftp_src_dir="$withval", ncftp_src_dir="$NCFTP_SRC_DIR")
if test -d "$ncftp_src_dir" ; then
     AC_MSG_RESULT([$ncftp_src_dir found])
```

```
     NCFTP_SRC_DIR=$ncftp_src_dir
else
     AC_MSG_ERROR([no such directory $ncftp_src_dir])
Fi

dnl Make sure we can find ncftp.h in NCFTP_SRC_DIR.
AC_MSG_CHECKING([for ncftp.h in gsincftp-src])
if test -f "$NCFTP_SRC_DIR/libncftp/ncftp.h" ; then
     AC_MSG_RESULT([yes])
else
     AC_MSG_ERROR([$NCFTP_SRC_DIR/libncftp/ncftp.h not found])
Fi

AC_MSG_CHECKING([for globus replica catalog source directory])
AC_ARG_WITH(globus-rep-cat-src,[ --with-globus-rep-cat-src=<dir> Default is
\$GLOBUS_REP_CAT_DIR],
globus_rep_cat_dir="$withval",
globus_rep_cat_dir="$GLOBUS_REP_CAT_DIR")
if test -d "$globus_rep_cat_dir" ; then
     AC_MSG_RESULT([$globus_rep_cat_dir found])
     GLOBUS_REP_CAT_DIR=$globus_rep_cat_dir
else
     AC_MSG_ERROR([no such directory $globus_rep_cat_dir])
fi

dnl Make sure we can find globus_replica_catalog.h in GLOBUS_REP_CAT_DIR.
AC_MSG_CHECKING([for ncftp.h in globus-rep-cat-src])
if test -f "$GLOBUS_REP_CAT_DIR/libraries/catalog/globus_replica_catalog.h"
; then
     AC_MSG_RESULT([yes])
else
     AC_MSG_ERROR([$GLOBUS_REP_CAT_DIR/libraries/catalog not found])
Fi

dnl Get staging option default is script
dnl variable STAGING_PLUGIN.
AC_MSG_CHECKING([for staging plugin option])
AC_ARG_WITH(staging-plugin,[ --with-staging-plugin=<option> Default is
script ],
staging_plugin="$withval", staging_plugin="script")
if test x$staging_plugin = xhrm || test x$staging_plugin = xscript ; then
     AC_MSG_RESULT([$staging_plugin selected])
     STAGING_PLUGIN=$staging_plugin
else
     AC_MSG_ERROR([invalid option $staging_plugin])
Fi

if test x$STAGING_PLUGIN = xhrm ; then
dnl Get orbacus source directory from command line option or environment
dnl variable ORBACUS_DIR.
AC_MSG_CHECKING([for orbacus directory])
AC_ARG_WITH(orbacus-install,[ --with-orbacus-install=<dir> Default is
\$ORBACUS_DIR],
orbacus_dir="$withval", orbacus_dir="$ORBACUS_DIR")
if test -d "$orbacus_dir" ; then
     AC_MSG_RESULT([$orbacus_dir found])
     ORBACUS_DIR=$orbacus_dir
else
     AC_MSG_ERROR([no such directory $orbacus_dir])
Fi
fi

AM_CONDITIONAL(STAGING_SCRIPT, test x$STAGING_PLUGIN = xscript)
AM_CONDITIONAL(STAGING_HRM, test x$STAGING_PLUGIN = xhrm)

dnl Just set GDMP_INSTALL_DIR
GDMP_INSTALL_DIR=`pwd`
```

```
dnl Define required variables.
dnl GLOBUS_ARCH given by $(host_alias)
AC_SUBST(GLOBUS_INSTALL_PATH)
AC_SUBST(GDMP_INSTALL_DIR)
AC_SUBST(NCFTP_SRC_DIR)
AC_SUBST(GLOBUS_REP_CAT_DIR)
AC_SUBST(GLOBUS_FLAVOR)
AC_SUBST(GDMP_INSTALL_DIR)
AC_SUBST(ORBACUS_DIR)
AC_SUBST(STAGING_PLUGIN)

AC_OUTPUT(Common/Makefile Threads/Makefile DataMover/Makefile
ControlMsgs/Makefile RequestManager/Makefile Security/Makefile
DBManager/Makefile FlatFile/Makefile HRMIDL/Makefile
ReplicaCatalogue/Makefile StagingPlugins/Makefile Misc/Makefile lib/Makefile
Main/Makefile Makefile etc/gdmp_environment etc/Makefile
utils/gdmp_server_start utils/Makefile)
```

Lines which start with the m4 builtin macro *dnl* are comments that don't appear in the generated *configure* script. The macros that begin with AC come with autoconf, and those that begin with AM usually come with automake.

These macros are written in m4 [9] and reside in /usr/share/aclocal, if you installed autoconf/automake under /usr.

These are the macros presents in the GDMP *configure.in* file:

- AC_INIT is always the first macro in *configure.in*. It expands to a lot of boilerplate code shared by all configure scripts; this code parses the command line arguments to *configure* script. The macro's one argument is a file that should be present in the source directory; this is used as a sanity check, to be sure *configure* has correctly located the source directory. In our case for example it is the first file in the GDMP Common dir.
- AC_CONFIG_AUX_DIR allows for an alternative directory to be specified for the location of auxiliary scripts such as *config.guess*, *config.sub*, etc. In GDMP we asked for the use of the *config* auxiliary dir.
- AM_CONFIG_HEADER specifies a header file to create; this will almost always be *config.h*. The created header file will contain C pre-processor symbols defined by configure. At a minimum, the symbols PACKAGE and VERSION will be defined, which makes it easy to put the name and version of a program in the code without hard-coding them.
- AM_INIT_AUTOMAKE initializes automake; the arguments to this macro are the name and version of the package being compiled. (These arguments become the values of PACKAGE and VERSION, defined in *config.h*).
- AC_PROG_CXX locates the C++ compiler.
- AM_PROG_LIBTOOL is used by automake to set up its use of libtool. This macro allows to set several variables, such as `host_alias` where the type of the operating system used is stored. For Linux RedHat 6.1 the value of such variable is *i686-pc-linux-gnu*. This variable can be used instead of the environmental variable GLOBUS_ARCH required by GDMP.
- AC_CHECK_LIB looks for the named function in the named library specified by its base name. If the argument function is "main" than the library is searched for and, by default, the C preprocessor macro HAVE_LIB_lib is set so that it could be directly used in the code, and the flag -llib is automatically added in LIBS. Here, only the check for the library pthread has been put (so automatically the C macro

HAVE_LIB_PTHREAD is defined and the flag - lpthread added to LIBS), but other libraries could be searched as well, given that they are located in a standard search path for libs. In order to locate a library wich is not in the standard library search path, one can use the method as in the following example:

```
for dir in /usr/local/gsincftp /usr/local/globus ; do
    if test –d "$dir" ; then
         LDFLAGS="$LDFLAGS –L$dir/lib"
         CPPFLAGS="$CPPFLAGS –I$dir/include"
         break
    fi
done
AC_CHECK_LIB(gsincftp,onegsincftpfunction,[LIBS="-lgsincftp $LIBS"])
AC_CHECK_LIB(globus, oneglobusfunction,[LIBS="-lglobus $LIBS"])
```

The method described above to look for libraries has not been used at the moment in GDMP.

- AC_HEADER_DIRENT searches a number of specific header files for a declaration of the C type DIR.
- AC_HEADER_STDC checks whether the present system has the standard ANSI standard C header files.
- AC_CHECK_HEADERS looks for a series of headers
- AC_C_CONST defines the C preprocessor macro const to the string const if the C compiler supports the const keyword
- AC_TYPE_SIZE_T looks for the type `size_t`. If not defined on the system, it defines it (as a macro) to be `unsigned'
- AC_FUNC_UTIME_NULL defines the C preprocessor macro HAVE_UTIME_NULL if a call to utime with a NULL utimbuf pointer sets the file's time stamp to the current time
- AC_CHECK_FUNCS looks for a series of functions
- AC_MSG_CHECKING notifies the user that *configure* is checking for any of the external packages (globus, gsincftp, globus_replica_catalog, orbacus) or for a particular feature like a staging plugin option. This macro must be followed by a call to AC_MSG_RESULT to print the result of the check and the newline.
- AC_ARG_WITH macro allows the maintainer to specify additional packages needed by GDMP (globus, gsincftp, globus_replica_catalog, orbacus). The user indicates this preference by invoking *configure* with an option such as `-- with-globus-install' for globus-install directory. If an optional argument is given, this value is available to the shell code in the shell variable `withval`.
- AC_MSG_RESULT notifies the user of the results of a check that is almost always the value of the cache variable for the check, typically `yes', `no', or a file name. This macro follows a call to AC_MSG_CHECKING, and the result-description should be the completion of the message printed by the call to AC_MSG_CHECKING.
- AC_MSG_ERROR macro prints an error message on the standard error output and exits *configure* with a nonzero status.
- AM_CONDITIONAL macro is used introduce a conditional. It takes two arguments. The first argument is the name of the conditional. The second argument is a shell condition, suitable for use in a shell ‘if’ statement. The condition is evaluated when *configure* is run .

- AC_SUBST "exports" a variable into the files generated by configure. In order to buid GDMP four environmental variables need to be set, as described in section 2. They are `GLOBUS_INSTALL_PATH`, `GDMP_INSTALL_DIR`, `NCFTP_SRC_DIR`, `GLOBUS_REP_CAT_DIR`. Instead of defining these variables as global, one could define them in a build script or somewhere else at convenience.
- AC_OUTPUT lists the files to be created by the configure script. These will be created from a file with the same name, with *.in* appended. For example, the output file Common/Makefile is generated from Common/Makefile.in. A *Makefile* is generated per subdir to create object files and convenience libraries. The *Makefile* in Main invokes recursively all subdirs Makefiles and creates a global library to be used with GDMP package. The top level Makefile can be used to generate a package binary tarball. At the moment this feature has not yet been exploited. In the lib subdir the *Makefile* is used to create a user GDMP library. Automake generates the *Makefile.in* files, starting from the user provided *Makefile.am* files.

3.3 Creating GDMP makefile_header

In order to use a set of macro definitions needed to build the package, a *makefile_header* has been written starting from the *Makefile* of the original package. Here we list the *makefile_header* for GDMP:

```
MAIN_DIR = @GDMP_INSTALL_DIR@/Main
BIN_DIR = @GDMP_INSTALL_DIR@/bin
LIBGDMP_DIR = @GDMP_INSTALL_DIR@/lib
MISC_DIR = @GDMP_INSTALL_DIR@/Misc
COMMON_DIR = @GDMP_INSTALL_DIR@/Common
CONTROL_MSGS_DIR = @GDMP_INSTALL_DIR@/ControlMsgs
SECURITY_DIR = @GDMP_INSTALL_DIR@/Security
REQ_MAN_DIR = @GDMP_INSTALL_DIR@/RequestManager
DATA_MOVER_DIR = @GDMP_INSTALL_DIR@/DataMover
THREADS_DIR = @GDMP_INSTALL_DIR@/Threads
DB_MANAGER_DIR = @GDMP_INSTALL_DIR@/DBManager
REP_CATALOG_DIR = @GDMP_INSTALL_DIR@/ReplicaCatalogue
STAGINGPLUGINS_DIR = @GDMP_INSTALL_DIR@/StagingPlugins
GDMPHOST_DIR = @GDMP_INSTALL_DIR@/GDMPHost
GDMPFILE_DIR = @GDMP_INSTALL_DIR@/GDMPFile
GDMP_CATALOGUE_DIR = @GDMP_INSTALL_DIR@/GDMPCatalogue
if OBJECTIVITY_SELECTED
OBJECTIVITY_CFLAGS = -DOBJECTIVITY -I@OBJECTIVITY_DIR@/include
OBJECTIVITYLDFLAGS = -L@OBJECTIVITY_DIR@/lib -
L@OBJECTIVITY_DIR@/ToolKit/lib
OBJECTIVITYLIBS = -loo -looseccl -lpthread
Endif
if STAGING_SCRIPT
STAGING_CFLAGS = -DSCRIPT_STAGING=@STAGING_PLUGIN@
ORBACUS_CFLAGS =
ORBACUSLDFLAGS =
ORBACUSLIBS =
endif
if STAGING_HRM
HRMIDL_DIR = @GDMP_INSTALL_DIR@/HRMIDL
STAGING_CFLAGS = -DHRM_STAGING=@STAGING_PLUGIN@
ORBACUS_CFLAGS = -I@ORBACUS_DIR@/include
ORBACUSLDFLAGS = -L@ORBACUS_DIR@/lib
ORBACUSLIBS = -lOB -lJTC -lpthread -lCosNaming
Endif

DATAMOVER_CFLAGS = -I@NCFTP_SRC_DIR@/libncftp -I@NCFTP_SRC_DIR@/Strn \
                   -I@NCFTP_SRC_DIR@/sio
```

```
REP_CATALOG_CFLAGS = -I@GLOBUS_REP_CAT_DIR@/libraries/catalog $(LDAP_CFLAGS)

include $(GLOBUS_FLAVOR)/etc/makefile_header

INCLUDES = -I$(includedir) -I$(CONTROL_MSGS_DIR) -I$(COMMON_DIR) \
           -I$(SECURITY_DIR) -I$(REQ_MAN_DIR) -I$(DATA_MOVER_DIR)\
           -I$(THREADS_DIR) -I$(MISC_DIR) -I$(DB_MANAGER_DIR) -I$(HRMIDL_DIR)\
           -I$(STAGINGPLUGINS_DIR) -I$(GDMPHOST_DIR) $(DB_MANAGER_CFLAGS) \
           -I$(GDMPFILE_DIR) -I$(GDMPCATALOGUE_DIR) $(DATAMOVER_CFLAGS)
           $(ORBACUS_CFLAGS) $(STAGING_CFLAGS) $(OBJECTIVITY_CFLAGS) \
           -I$(REP_CATALOG_DIR) $(REP_CATALOG_CFLAGS) \
           $(GLOBUS_COMMON_CFLAGS) \
           $(GLOBUS_GSSAPI_CFLAGS) \
           $(GLOBUS_IO_CFLAGS) \
           $(GLOBUS_NEXUS_CFLAGS)\
           $(GLOBUS_IO_CFLAGS)

DATAMOVERLDFLAGS =   -L@NCFTP_SRC_DIR@/libncftp \
                     -L@NCFTP_SRC_DIR@/Strn \
                     -L@NCFTP_SRC_DIR@/sio

DATAMOVERLIBS = -lncftp -lStrn -lsio

REP_CATALOGLDFLAGS = -L@GLOBUS_REP_CAT_DIR@/libraries/catalog $(LDAP_LDFLAGS)

REP_CATALOGLIBS = -lglobus_replica_catalog $(LDAP_LIBS)

DATAMOVERLDFLAGS =   -L@NCFTP_SRC_DIR@/libncftp \
                     -L@NCFTP_SRC_DIR@/Strn \
                     -L@NCFTP_SRC_DIR@/sio

DATAMOVERLIBS = -lncftp -lStrn -lsio

MYLDFLAGS =    -L$(libdir) \
                $(DB_MANAGER_LDFLAGS) \
                $(DATAMOVERLDFLAGS) \
                $(REP_CATALOGLDFLAGS) \
                $(GLOBUS_COMMON_LDFLAGS) \
                $(GLOBUS_GSSAPI_LDFLAGS) \
                $(GLOBUS_IO_LDFLAGS) \
                $(GLOBUS_NEXUS_LDFLAGS) \
                $(ORBACUSLDFLAGS) \
                $(OBJECTIVITYLDFLAGS)

MYLIBS =  $(LIBS) \
           $(DB_MANAGER_LIBS) \
           $(DATAMOVERLIBS) \
           $(REP_CATALOGLIBS) \
           $(GLOBUS_COMMON_LIBS) \
           $(GLOBUS_GSSAPI_LIBS) \
           $(GLOBUS_IO_LIBS) \
           $(GLOBUS_NEXUS_LIBS)\
           $(ORBACUSLIBS) \
           $(OBJECTIVITYLIBS)

GDMPOBJS = $(LIBGDMP_DIR)/libgdmp.la
MYGDMPLDFLAGS = $(MYLDFLAGS) $(MYLIBS)
```

The variables which appear in between the @ symbol are substituted with their values by *configur*e. They are ordinary shell variables that are set in *configure*. To make configure substitute a particular variable into the generated output files, the macro AC_SUBST must be called (in *configure.in*) with that variable name as an argument.

The header file includes, as in the original *Makefile*, the Globus *makefile_header* with a lot hardcoded and platform dependent macros. This inclusion should be avoided in the future and proper variable definitions should be provided. The macro GDMPOBJS is used to link against the GDMP *user* library, instead of using the object files.

3.4 Creating the Makefile.am files

The *Makefile.am* files have been created for each package subdirectory. The *Makefile.am* in the top source directory is generally very simple; here is the one for the GDMP package:

```
SUBDIRS= Common ControlMsgs Security RequestManager DataMover DBManager
Threads ReplicaCatalogue FlatFile StagingPlugins HRMIDL Misc lib etc utils Main
```

The SUBDIRS line instructs automake to recursively look for *Makefile.am* files in the given subdirectories in the order specified. So for istance, the ***lib*** directory is processed before ***Main*** so that the user library is built before the executable are linked against it.

The *Makefile.am* files for almost all subdirectories (except Main and lib) are similar. For example for the ***Common*** directory:

```
include $(top_srcdir)/makefile_header noinst_LTLIBRARIES = libCommon.la
libCommon_la_SOURCES = gdmp_common.C gdmp_common.h
```

With the `include` directive the produced GDMP *makefile_header* is implicitly included after being preprocessed by *configure*.

The `noinst_LTLIBRARIES` macro allows the package manager to create the Libtool convenience library. For the Libtool library named in this macro, Automake will create the Libtool convenience library which can subsequently be linked into other Libtool libraries. The prefix `noinst_` tells automake that the current library should not be installed, but should be built anyway. Although not required for compilation, *source.h* (`gdmp_common.h`) is listed in the `SOURCES` macro of *library.la* (`libCommon.la`) so that correct source dependencies are generated.

For the ***Main*** directory the *Makefile.am* is:

```
include $(top_srcdir)/makefile_header

bin_PROGRAMS = gdmp_server gdmp_replicate_file_get\ gdmp_publish_catalogue
gdmp_host_subscribe gdmp_setup\ gdmp_get_catalogue gdmp_filter_catalogue
gdmp_stage_complete\ gdmp_cleanup gdmp_ping …..
gdmp_server_SOURCES = gdmp_server.C
gdmp_server_LDFLAGS = $(GDMPOBJS) $(MYGDMPLDFLAGS)
gdmp_replicate_file_get_SOURCES = gdmp_replicate_file_get.C
gdmp_replicate_file_get_LDFLAGS = $(GDMPOBJS) $(MYGDMPLDFLAGS)
.........
```

The `bin_PROGRAMS` macro lists the programs to be created. *`program`*`_SOURCES` lists the source files to be compiled and linked to create the program. All files in this variable are automatically included in the distribution.

The *`program`*`_LDFLAGS` macro is used to redefine, if needed, the global LDFLAGS while building the specific program. In this case the global LDFLAGS is redefined using the `MYGDMPLDFLAGS` and `GDMPOBJS` variables that list the flags to be passed to the linker. The global LDFLAGS needed to be overwritten in the case of GDMP because of the hardcoded definition present in the Globus *makefile_header*. The variables `MYGDMPLDFLAGS` and `GDMPOBJS` are defined in GDMP *makefile_header*.

For the ***lib*** directory the *Makefile.am* is:

```
LINK =$(LIBTOOL) --mode=link $CC $(AM_CFLAGS) $(CFLAGS) -o $@
lib_LTLIBRARIES = libgdmp.la
lib_la_SOURCES=
lib_la_LDFLAGS=-no-undefined -version-info 1:0:1
if STAGING_HRM
libgdmp_la_LIBADD = ../Common/libCommon.la ../ControlMsgs/libControl.la \
../DBManager/libDBManager.la ../DataMover/libDataMover.la \
../Misc/libMisc.la ../RequestManager/libRequestManager.la\
../Security/libSecurity.la ../ReplicaCatalogue/libReplicaCatalogue.la
../FlatFile/libFlatFile.la ../Threads/libThreads.la\
libgdmp_la_LIBADD +=../HRMIDL/libHRMIDL.la\
../StagingPlugins/libStagingPlugins.la
endif
if STAGING_SCRIPT
libgdmp_la_LIBADD = ../Common/libCommon.la ../ControlMsgs/libControl.la \
../DBManager/libDBManager.la ../DataMover/libDataMover.la \
../Misc/libMisc.la ../RequestManager/libRequestManager.la\
../Security/libSecurity.la ../ReplicaCatalogue/libReplicaCatalogue.la
../FlatFile/libFlatFile.la ../Threads/libThreads.la\
libgdmp_la_LIBADD +=../StagingPlugins/libStagingPlugins.la
endif
```

The code tells automake that we want to build a library (libgdmp.la) for use within the build tree. Since there are not source files by default, we need to define the variable `LINK` at the top line, since libtool does not define it automatically. This seems to be a **bug** in the current version of libtool (1.3.4). To pass any additional flag to libtool during the build process, we need to use the LDFLAGS macro for libgdmp library, as follows:

```
lib_la_LDFLAGS=-no-undefined –version-info 1:0:0
```

We have used the `-no-undefined` option to require that *all* symbols are solved while creating shared libraries. With the second option, we pass version information to be used with shared libraries, for istance. Version info are specified using the syntax *current:revision:age*. *Current* is the number of the current interface exported by the library (you increase this number when the library user interface changes); *revision* is the implementation number of the current interface (you increase this number if the user interface does not change but the implementation does); *age* is the number of previous additional interfaces supported by library.

4 Building GDMP

To build and install the package one must run the *configure* script which tests system features. The actual build is performed using the *make* program. Before invoking these commands we must prepare files that will be process by the above commands. To do this a *bootstrap* procedure is executed.

4.1 The bootstrap file

A *bootstrap* file is a script which just runs the various commands required to bring the source tree into a state in which the end user can invoke the *configure* and *make* commands.
This is the bootstrap file:

```
#!/bin/sh
set -x
aclocal -I config
autoheader
libtoolize --automake
automake --foreign --add-missing --copy
autoconf
```

It must be invoked everytime any of the files *configure.in* or *Makefile.am* is changed, anywhere in the package tree.

Because *configure.in* contains macro invocations that are not known to Autoconf itself (like `AM_INIT_AUTOMAKE`), it's necessary to collect all the macro definitions for Autoconf to use when generating *configure*. This is done using the **aclocal** program, so called because it generates *aclocal.m4*. If we examine the contents of *aclocal.m4*, we find the definition of the `AM_INIT_AUTOMAKE` macro. The `–I config` option inform the aclocal program of the existence of the new directory *config* where many of the files added by runninng **autoconf** and **automake** (such as config.guess, config.sub, install-sh ..) are created.

Autoheader runs m4 over *configure.in*, but with key macros defined differently from when **autoconf** is executed, such that suitable cpp definition are output to *config.h.in*.

Libtoolize is used to add libtool support to package. In particolar, the option `–-automake` adds *config.guess*, *config.sub*, *ltconfig* and *ltmain.sh* files to distribution. This files are created under the config directory.

Automake command processes *Makefile.am* to produce a standards-compliant *Makefile.in*. It creates all standard targets, such as *install* and *clean*. It also creates more complex targets. For istance, simply typing *make dist* you can create a standard *.tar.gz* file containing the package source distribution. This feature has not yet been tested with the current provided version of *autotoolized* GDMP.

The option `--add-missing` copies some boilerplate file from Automake installation into the current directory. A number of utility scripts are also installed that are used by generated Makefile's, in particolar by the install target. One of the this script is for istance *mkinstalldirs*, to create installation directoris under the installation tree.

Autoconf processes *configure.in* to produce a *configure* script. *Configure* is a portable shell script which examines the build environment to determine which libraries are available, which features the platform has, where libraries and headers are located, and so on. Based on this information, it modifies compiler flags, generates makefiles, and/or outputs the file config.h with appropriate pre-processor symbols defined.

If there are errors when running bootstrap, one needs to check the correctness of the *Makefile.am* or *configure.in* files.

4.2 Invoking configure

The *configure* script tests system features. The command is invoked as follows:

```
$./configure --prefix=<GDMP installation tree>
```

where the `<GDMP installation tree>` is the installation directory.

The script takes a large number of command line options. One of the most frequently used is the `prefix`. If generated *Makefiles* choose to observe the argument passed with this option, it is possible to entirely relocate the architectureindependent portion of a package when it is installed. The prefix directory must be different from the

build directory (`$HOME/gdmp`). After invoking **configure** command in the top directory several files are generated:

- *config.cache* where the result of system tests to determine the availbility of features are stored;
- *config.status* that may be used to recreate the current configuration;
- *config.log* file to look when *configure* goes hay-wire or a test produces a nonsense result.
- *Makefile,* one in the package directory and one for each subdirs.

4.3 Invoking make

Now that the build tree is configured, it is possible to go on and build the package and install it into the installation tree. Now **make** is invoked to process the *Makefile* file in the package directory. Several target can be specified on the make command line:

- *all* the default target which compiles the program;
- *clean* to delete the results of compilation (object file and executables);
- *install* to create installation directories if needed and copy the software into them;
- *dist* to build a tarball (.tar.gz) for distribution.

During building, in the various *service* subdirs where a convenience library is built, the temporary hidden subdirectories *.deps* and *.libs* are created. In the *.deps* directory you can find files which describe the source code dependencies from the header files.

In the .libs directory, convenience libraries and temporary files to build them are put in place.

After all convenience libraries are built, the top level Makefile specifies to process the lib subdirectory. Here, all convenience libraries are "*linked*" together in a bigger library, the package library libgdmp.a (or its shared version).

Then, the Makefile for the *Main* subdirectory is invoked to build the executables agains the library just built.

4.4 Invoking make install

When running *make install* the directories ***lib, bin, include, utils, tmp, var, etc*** will be created the in the installation directory specified via the `--prefix` option with configure. In ***lib*** the gdmp library will be put, in ***bin*** the executables, in ***include*** the header files, in ***etc*** the .cert .key .proxy files, in ***utils*** the services files. ***var*** and ***tmp*** are currently empty.

To do this the *Makefile.am* in the top dir must contain few other lines. Here is the final *Makefile.am*:

```
SUBDIRS= Common ControlMsgs Security RequestManager DataMover Threads
DBManager ReplicaCatalogue FlatFile StagingPlugins HRMIDL Misc lib etc utils
Main

configincdir = $(prefix)/include/gdmp
configinc_HEADERS = config.h

TMPDIRS = var tmp

install-data-local:
     for f in $(TMPDIRS); do \
          echo "Creating $(prefix)/$$f dir"; \
          $(mkinstalldirs) $(prefix)/$$f; \
     done
```

Automake enables the user to extend the list of directories to create new ones. The line

```
configincdir = $(prefix)/include/gdmp
```

instructs Automake to automatically create the directory specified as the value of the variable `configincdir` under the installation dir and

```
configinc_HEADERS = config.h
```

puts a *config.h* header file in this new directory. The macro used to define the new directory must end with 'dir'. Automake provides the predefined variable *pkgincludedir* to accommodate package include files [its value is $(includedir)/$(PACKAGE)]. In our case this predefined variable could not be used because the GLOBUS automake_header redefines the value of the variable includedir to point to the GLOBUS installed include directory. So we have to explicitely define its value to be `$(prefix)/include/gdmp`. During the install processs the `install-datalocal` target is invoked, if defined by the package maintainer. This target has been used in our case to create the ***var*** and ***tmp*** directories used by gdmp. The default ownership and mode of this directories can also be changed here.

Now the SUBDIRS variable contains the ***etc*** and ***utils*** dirs as well. So also in these directories there is a *Makefile.am*. The *Makefile.am* under ***utils*** is :

```
gdmputilsdir = $(prefix)/utils
gdmputils_DATA = gdmp_server_start.in \
                 gdmp_stage_from_mss\
                 get_progress_report

install-data-hook: gdmp_server_start
     if test -d $(prefix)/utils; then $(mkinstalldirs) $(prefix)/utils;fi
     rm $(prefix)/utils/gdmp_server_start.in
     cp gdmp_server_start $(prefix)/utils
```

The installation of miscellaneous data files is supported by automake using the DATA family of variables. Automake installs its auxiliary data files according to the line:

```
gdmputils_DATA = gdmp_server_start.in \
                 gdmp_stage_from_mss \
                 get_progress_report
```

The data will be installed in the directory `$(prefix)/utils` .

4.5 Invoking make dist

When running *make dist* a tar distribution is created in the top directory. The generated tar file is named package-version.tar.gz, and will unpack into a directory named package-version. For the most part, the files to distribute are automatically found by Automake: all source files are automatically included in a distribution, as well as all `Makefile.am''s and Makefile.in's. Automake will recursively include all subdirectories in the distribution because the SUBDIRS macro has been defined. All files which aren't included automatically (by default) in the distribution are added using the macro EXTRA_DIST in Makefile.am. So in the main Makefile.am we add the line

```
EXTRA_DIST = bootstrap
```

to add the bootstrap file.

In Makefile.am one needs also to define a dist-hook rule which Automake will arrange to run when the copying work for this directory is finished. We use this rule to do several things to the distribution directory: remove files that erroneously end up in the distribution or are not very important; copy files that are not distributed.

```
dist-hook:
     mkdir $(distdir)/bin $(distdir)/tmp $(distdir)/var
     for f in $(DIST_SUBDIRS); do \
          rm $(distdir)/$$f/Makefile.in;\
     done
     cp $(srcdir)/utils/gdmp_server_start\
          $(srcdir)/utils/get_progress_report $(distdir)/utils
     cp $(srcdir)/etc/gdmp_server.cert $(srcdir)/etc/gdmp_server.key \
          $(srcdir)/etc/gdmp_server.proxy $(distdir)/etc
     -rm $(distdir)/Makefile.in $(distdir)/aclocal.m4 \
          $(distdir)/config.h $(distdir)/config.h.in \
          $(distdir)/configure \
          $(distdir)/stamp-h.in \
          $(distdir)/etc/gdmp_environment.in \
          $(distdir)/utils/gdmp_server_start.in
```

5 Adding a new module to GDMP

We give here a short summary of the steps needed in order to add one or more modules to the GDMP package and correctly configure this new module with the GNU Autotools.

1. Create the directory module gdmp/<module>.
2. Change gdmp/config.in file to add in the configure output list (AC_OUTPUT) in the right place (i.e. before lib and Main), the string <Module>/Makefile.
3. If there are executables to be built for this module, than the correspondent source files and headers should go into the gdmp/Main directory. In this case, the gdmp/Main/Makefile.am file needs to be updated with the the name of the executables in the bin_PROGRAMS line (look at what is done for the others) . Also the corrispondent lines, relatives to: <program_name>_SOURCES and <program_name>_LDFLAGS need to exist. You can copy one of the existing examples.
4. Copy into the module directory one of the Makefile.am that you find for instance in gdmp/Common or gdmp/Threads. Lets supposed that you copy the gdmp/Common/Makefile.am.
5. Change the word "Commonincdir" with <Module>incdir.
6. Change the word Commoninc_HEADERS with <Module>inc_HEADERS and make it equal to whatever header files you have for this module. If you do not have header files you just delete those two lines.
7. Specify after the noinst_LTLIBRARIES the library name for this module, it should be lib<Module>.la
8. Change the line "libCommon_la_SOURCES =" with lib<Module>_la_SOURCES and after that specify the sources and headers for this particular module.
9. In gdmp/lib, change Makefile.am to add in libgdmp_la_LIBADD the string ../<Module>/lib<Module>.la
10. In the top Makefile.am under gdmp, on the SUBDIRS line, add the name of the new module.

6 A short summary of commands

In order to build GDMP with the new tools, you need to execute from the gdmp root directory:

- ./bootstrap (the first time or everytime you change the configure.in or Makefile.am)
- ./configure –prefix=<install directory>
- make clean (to clean up what previously done in the build directory)
- make
- make install

7 The todo list

Here we give a short list of things that still need an implementation or need improvement.

- The new package version of GDMP needs a good revision to eliminate the dependency on the hardcoded GLOBUS *makefile_header*.
- It is foreseen to include the GDMP RPM spec file among the files preprocessed by configure to generate installation specific implementation. This work is on going.
- The installation step should also include the production of a script to correctly install a *gdmp_server* on a machine, defining its entry in the superdaemon configuration file or have the machine invoke a startup file at reboot. Also, the customization of the *gdmp_environment* and *gdmp_server_start* files can be improved.
- GDMP contains same architecture dependent code (see the usage of the getsockopt). Some macro definitions should be added in order to check the correct signature of system calls and use them correctly in the code.
- What has been described is just a first attempt to use the GNU tools to package GDMP in a more portable way. With experience many things and choices can surely be optimized.

8 Appendix

Configure.in file

```
dnl Process this file with autoconf to produce a configure script.
AC_INIT(Common/gdmp_common.C)
dnl Set config options
AC_CONFIG_AUX_DIR(config)
AM_CONFIG_HEADER(config.h)
AM_INIT_AUTOMAKE(gdmp,1.2.2)
dnl Checks for programs.
AC_PROG_CXX
dnl Checks for use of libtool
AM_PROG_LIBTOOL
dnl Checks for libraries.
dnl Replace `main' with a function in -lpthread:
AC_CHECK_LIB(pthread, main)
dnl Checks for header files.
AC_HEADER_DIRENT
AC_HEADER_STDC
AC_CHECK_HEADERS(fcntl.h malloc.h sys/file.h unistd.h)
```

```
dnl Checks for typedefs, structures, and compiler characteristics.
AC_C_CONST
AC_TYPE_SIZE_T
dnl Checks for library functions.
AC_FUNC_UTIME_NULL
AC_CHECK_FUNCS(putenv strdup strtoul)
dnl Get globus install directory from command line option or environment
dnl variable GLOBUS_INSTALL_PATH.
AC_MSG_CHECKING([for globus-install directory])
AC_ARG_WITH(globus-install,[ --with-globus-install=<dir> Default is
\$GLOBUS_INSTALL_PATH],
globus_install="$withval", globus_install="$GLOBUS_INSTALL_PATH")
if test -d "$globus_install" ; then
AC_MSG_RESULT([found $globus_install])
GLOBUS_INSTALL_PATH=$globus_install
else
AC_MSG_ERROR([no such directory $globus_install])
Fi
dnl Determine globus flavor from command line option.
dnl If command line option is not given, look in
dnl $GLOBUS_INSTALL_PATH/development for a directory with threads (but not
dnl nothreads) in the name.
echo "Attempting to determine globus flavor..."
AC_ARG_WITH(globus-flavor,[ --with-globus-flavor=<dir> Default is <globusinstall>/
development/<directory with threads>],
globus_flavor=$withval, globus_flavor="")
if test -z "$globus_flavor" ; then
globus_flavor_tmp=`ls -1 $GLOBUS_INSTALL_PATH/development 2>/dev/null | grep thread |
grep -v nothread`
globus_flavor=$GLOBUS_INSTALL_PATH/development/`echo "$globus_flavor_tmp" | head -n 1`
fi
echo "globus flavor is \"$globus_flavor\""
GLOBUS_FLAVOR=$globus_flavor
dnl We need globus flavor in order to get the globus makefile_header.
dnl Make sure we can find it.
AC_MSG_CHECKING([for makefile_header in globus-flavor])
if test -f "$GLOBUS_FLAVOR/etc/makefile_header" ; then
AC_MSG_RESULT([yes])
else
AC_MSG_ERROR([$GLOBUS_FLAVOR/etc/makefile_header not found])
Fi
dnl Get gsnicftp source directory from command line option or environment
dnl variable NCFTP_SOURCE_DIR
AC_MSG_CHECKING([for gsincftp source directory])
AC_ARG_WITH(gsincftp-src,[ --with-gsincftp-src=<dir> Default is \$NCFTP_SRC_DIR],
ncftp_src_dir="$withval", ncftp_src_dir="$NCFTP_SRC_DIR")
if test -d "$ncftp_src_dir" ; then
AC_MSG_RESULT([$ncftp_src_dir found])
NCFTP_SRC_DIR=$ncftp_src_dir
else
AC_MSG_ERROR([no such directory $ncftp_src_dir])
Fi
dnl Make sure we can find ncftp.h in NCFTP_SRC_DIR.
AC_MSG_CHECKING([for ncftp.h in gsincftp-src])
if test -f "$NCFTP_SRC_DIR/libncftp/ncftp.h" ; then
AC_MSG_RESULT([yes])
else
AC_MSG_ERROR([$NCFTP_SRC_DIR/libncftp/ncftp.h not found])
fi
AC_MSG_CHECKING([for globus replica catalog source directory])
```

```
AC_ARG_WITH(globus-rep-cat-src,[ --with-globus-rep-cat-src=<dir> Default is
\$GLOBUS_REP_CAT_DIR],
globus_rep_cat_dir="$withval", globus_rep_cat_dir="$GLOBUS_REP_CAT_DIR")
if test -d "$globus_rep_cat_dir" ; then
AC_MSG_RESULT([$globus_rep_cat_dir found])
GLOBUS_REP_CAT_DIR=$globus_rep_cat_dir
else
AC_MSG_ERROR([no such directory $globus_rep_cat_dir])
Fi dnl Make sure we can find globus_replica_catalog.h in GLOBUS_REP_CAT_DIR.
AC_MSG_CHECKING([for ncftp.h in globus-rep-cat-src])
if test -f "$GLOBUS_REP_CAT_DIR/libraries/catalog/globus_replica_catalog.h" ; then
AC_MSG_RESULT([yes])
else
AC_MSG_ERROR([$GLOBUS_REP_CAT_DIR/libraries/catalog not found])
fi
dnl Get staging option default is script
dnl variable STAGING_PLUGIN.
AC_MSG_CHECKING([for staging plugin option])
AC_ARG_WITH(staging-plugin,[ --with-staging-plugin=<option> Default is script ],
staging_plugin="$withval", staging_plugin="script")
if test x$staging_plugin = xhrm || test x$staging_plugin = xscript ; then
AC_MSG_RESULT([$staging_plugin selected])
STAGING_PLUGIN=$staging_plugin
else
AC_MSG_ERROR([invalid option $staging_plugin])
fi
if test x$STAGING_PLUGIN = xhrm ; then
dnl Get orbacus source directory from command line option or environment
dnl variable ORBACUS_DIR.
AC_MSG_CHECKING([for orbacus directory])
AC_ARG_WITH(orbacus-install,[ --with-orbacus-install=<dir> Default is \$ORBACUS_DIR],
orbacus_dir="$withval", orbacus_dir="$ORBACUS_DIR")
if test -d "$orbacus_dir" ; then
AC_MSG_RESULT([$orbacus_dir found])
ORBACUS_DIR=$orbacus_dir
else
AC_MSG_ERROR([no such directory $orbacus_dir])
fi
fi
AM_CONDITIONAL(STAGING_SCRIPT, test x$STAGING_PLUGIN = xscript)
AM_CONDITIONAL(STAGING_HRM, test x$STAGING_PLUGIN = xhrm)
dnl Just set GDMP_INSTALL_DIR
GDMP_INSTALL_DIR=`pwd`
dnl Define required variables.
AC_SUBST(GLOBUS_INSTALL_PATH)
AC_SUBST(GDMP_INSTALL_DIR)
AC_SUBST(NCFTP_SRC_DIR)
AC_SUBST(GLOBUS_REP_CAT_DIR)
AC_SUBST(GLOBUS_FLAVOR)
AC_SUBST(GDMP_INSTALL_DIR)
AC_SUBST(ORBACUS_DIR)
AC_SUBST(STAGING_PLUGIN)
AC_OUTPUT(Common/Makefile Threads/Makefile DataMover/Makefile ControlMsgs/Makefile
RequestManager/Makefile Security/Makefile DBManager/Makefile Flatfile/Makefile
HRMIDL/Makefile StagingPlugins/Makefile ReplicaCatalogue/Makefile Misc/Makefile lib/Makefile
Main/Makefile Makefile etc/gdmp_environment etc/Makefile utils/gdmp_server_start utils/Makefile)
```

Makefile.am file

```
SUBDIRS= Common ControlMsgs Security RequestManager DataMover Threads DBManager
```

```
ReplicaCatalogue FlatFile StagingPlugins HRMIDL Misc lib etc utils Main
configincdir = $(prefix)/include/gdmp
configinc_HEADERS = config.h
TMPDIRS = var tmp
install-data-local:
for f in $(TMPDIRS); do \
echo "Creating $(prefix)/$$f dir"; \
$(mkinstalldirs) $(prefix)/$$f; \
done
```

Common/Makefile.am file

```
include $(top_srcdir)/makefile_header
Commonincdir = $(prefix)/include/gdmp
Commoninc_HEADERS = gdmp_common.h
noinst_LTLIBRARIES = libCommon.la
libCommon_la_SOURCES = gdmp_common.C gdmp_common.h
```

Main/Makefile.am file

```
include $(top_srcdir)/makefile_header
bin_PROGRAMS = gdmp_server stage gdmp_replicate_get gdmp_publish_catalogue
gdmp_host_subscribe gdmp_setup gdmp_get_catalogue gdmp_filter_catalogue gdmp_stage_complete
gdmp_cleanup gdmp_ping gdmp_catalog_cleanup
gdmp_server_SOURCES = gdmp_server.C
gdmp_server_LDFLAGS = $(GDMPOBJS) $(MYGDMPLDFLAGS)
gdmp_replicate_get_SOURCES = gdmp_replicate_get.C
gdmp_replicate_get_LDFLAGS = $(GDMPOBJS) $(MYGDMPLDFLAGS)
gdmp_publish_catalogue_SOURCES = gdmp_publish_catalogue.C
gdmp_publish_catalogue_LDFLAGS = $(GDMPOBJS) $(MYGDMPLDFLAGS)
gdmp_host_subscribe_SOURCES = gdmp_host_subscribe.C
gdmp_host_subscribe_LDFLAGS = $(GDMPOBJS) $(MYGDMPLDFLAGS)
gdmp_setup_SOURCES = gdmp_setup.C
gdmp_setup_LDFLAGS = $(GDMPOBJS) $(MYGDMPLDFLAGS)
gdmp_get_catalogue_SOURCES = gdmp_get_catalogue.C
gdmp_get_catalogue_LDFLAGS = $(GDMPOBJS) $(MYGDMPLDFLAGS)
gdmp_filter_catalogue_SOURCES = gdmp_filter_catalogue.C
gdmp_filter_catalogue_LDFLAGS = $(GDMPOBJS) $(MYGDMPLDFLAGS)
gdmp_stage_complete_SOURCES = gdmp_stage_complete.C
gdmp_stage_complete_LDFLAGS = $(GDMPOBJS) $(MYGDMPLDFLAGS)
gdmp_cleanup_SOURCES = gdmp_cleanup.C
gdmp_cleanup_LDFLAGS = $(GDMPOBJS) $(MYGDMPLDFLAGS)
gdmp_catalog_cleanup_SOURCES = gdmp_catalog_cleanup.C
gdmp_catalog_cleanup_LDFLAGS = $(GDMPOBJS) $(MYGDMPLDFLAGS)
gdmp_ping_SOURCES = gdmp_ping.C
gdmp_ping_LDFLAGS = $(GDMPOBJS) $(MYGDMPLDFLAGS)
stage_SOURCES = stage.C
stage_LDFLAGS = $(GDMPOBJS) $(MYGDMPLDFLAGS)
```

utils/Makefile.am file

```
gdmputilsdir = $(prefix)/utils
gdmputils_DATA = gdmp_server_start.in gdmp_stage_from_mss get_progress_report
install-data-hook: gdmp_server_start
if test -d $(prefix)/utils; then $(mkinstalldirs) $(prefix)/utils; fi
rm $(prefix)/utils/gdmp_server_start.in
cp gdmp_server_start $(prefix)/utils
```

etc/Makefile.am file

```
gdmpdir = $(sysconfdir)
gdmp_DATA = gdmp_environment.in gdmp_server.cert gdmp_server.key gdmp_server.proxy
install-data-hook: gdmp_environment
rm $(sysconfdir)/gdmp_environment.in
cp gdmp_environment $(sysconfdir)
```

9 Reference

(1) G.V.Vaughan, B.Ellison, T.Tromey, I.L.Taylor, GNU Autoconf, Automake, and Libtool, New Riders, 2001

(2) http://www.gnu.org/manual/autoconf/html_mono/autoconf.html, Autoconf manual

(3) http://www.gnu.org/manual/automake/html_mono/automake.html, Automake manual

(4) http://www.gnu.org/software/libtool/manual.html, libtool manual

(5) H. Stockinger, “GDMP 1.2.1 Code Documentation”, http://cmsdoc.cern.ch/cms/grid/code-docu/gdmp_1.2.1_code_documentation.ps

(6) “RPM HOWTO” - http://rpm.redhat.com/RPM-HOWTO

(7) “GNU Coding Standards” - http://www.gnu.org/prep/standards_toc.html

(8) “Filesystem Hierarchy Standard” - http://www.pathname.com/fhs/

(9) ”GNU m4” – http://www.gnu.org/software/m4