Enabling Grids for E-science in Europe

www.eu-egee.org

LCG-2 Middleware Internals and APIs
29th – 30th November 2004

Grid Data Management Hands-on

Antonio Delgado Peris
LCG Experiment Integration and Support
CERN IT

This presentation complements the APIs-Tutorial presentation

http://agenda.cern.ch/askArchive.php?base=agenda&categ=a044732&id=a044732s1t1/transparencies

EGEE is a project funded by the European Union under contract IST-2003-508833

Agenda

Enabling Grids for E-science in Europe

- Data Management command line tools
- lcg_utils API
- GFAL API
- Globus API

DM Command Line Tools

Enabling Grids for E-science in Europe

1. Check the syntax of the described commands in the manpages and LCG-2 User Guide.

   https://edms.cern.ch/file/454439//

2. Submit a job that creates a file and automatically brings it to the Grid (OutputData attribute).

3. Check the LFN, GUID and SURL of the file.

4. Copy the file to two SEs with lcg-rep and lcg-cp. Check that both files exist but only one is registered.

5. Register the file that was not already in the catalog.

6. Copy the file back to the UI and check it.

7. Delete and unregister all the files and replicas created.

DM Command Line Tools

Enabling Grids for E-science in Europe

General Remarks:

- These exercises will work both with the EDG and the new LFC file catalogs, but files registered in one catalog will not be visible in the other one.

- You may define the environmental variable $LCG_CATALOG_TYPE to be “edg” or “lfc” in order to use one or the other catalog (default is “edg”), and check that files are stored in one or the other catalog, but the functionality is the same.

- Since the LFC catalog is not published in the BDII, you need to define the environmental variable $LFC_HOST with the value of the server “lxb1191.cern.ch”, in order to use the catalog.

DM Command Line Tools

Enabling Grids for E-science in Europe

The automatic upload of data from a job with a default SE will only work properly if that CE has at least one SE defined as close in the information system (that is usually the case). Otherwise, use the StorageElement attribute

```
$ edg-job-sumbmit  -o jobid OutputData.jdl
$ edg-job-status   -i jobid
```

OutputData.jdl

```
Executable="OutputData.sh";
StdOutput="std.out";
StdError="std.err";
InputSandbox={"OutputData.sh"};
OutputSandbox={"std.out","std.err","dummy.dat"};

OutputData = { [
   OutputFile = "dummy.dat";
   StorageElement = "gilda-se-01.pd.infn.it";
   LogicalFileName = "lfn:AntonioDelgado_outputData"; ]
};

# Let the RB chose a CE near this SE
OutputSE = "gilda-se-01.pd.infn.it";
```

DM Command Line Tools

Enabling Grids for E-science in Europe

OutputData.sh

```
#!/bin/bash
# This is a simple BASH script that creates a dummy file

echo "Let us write into the file (dummy.dat)"
echo "This is the file for the OutputData JDL attribute" > dummy.dat
echo "Nothing really interesting to say, here..." >> dummy.dat
echo "See you!" >> dummy.dat

if [ -f dummy.dat ]; then
   echo "The file has been correctly created"
else
   echo "Something went wrong with the creation of the file" 1>&2
fi
```

```
$ edg-job-get-output  -i jobid

$ cat std.out
        Let us write into the file (dummy.dat)
        The file has been correctly created

$ cat DSUpload_Uc9HOrx8OGVa6JHAMhrTkw.out
        # <outputfile> <lfn|guid|Error>
        dummy.dat    lfn:AntonioDelgado_outputData
```

DM Command Line Tools

Enabling Grids for E-science in Europe

```
$ lcg-lg --vo gilda  lfn:AntonioDelgado_outputData_example
        guid:58d1e428-2f36-11d9-8a52-e231017ae683

$ lcg-lr --vo gilda  lfn:AntonioDelgado_outputData_example
        sfn://gilda-se-01.pd.infn.it/shared/gilda/generated/2004-11-05/file56a78a17-2f36-11d9-
                8a52-231017ae683
```

DM Command Line Tools

Enabling Grids for E-science in Europe

For the manual copy, the file has to be written in the SARoot directory (directory with permission for that VO). It appears in the information system ("flatfiles/SE00/gilda").

For lcg-cp's destination name and edg-griftp-exist, use "gsiftp" as protocol

```
$ lcg-rep --vo gilda  -d grid3.na.astro.it -P AntonioDelgado_outputData_1 \
 lfn:AntonioDelgado_outputData

$ lcg-cp  --vo gilda  lfn:AntonioDelgado_outputData \
   gsiftp://grid009.ct.infn.it/flatfiles/SE00/gilda/AntonioDelgado_outputData_2


$ edg-gridftp-exists gsiftp://grid3.na.astro.it/flatfiles/SE00/gilda/AntonioDelgado_outputData_1

$ edg-gridftp-exists gsiftp://grid009.ct.infn.it/flatfiles/SE00/gilda/AntonioDelgado_outputData_2


$ lcg-lg --vo gilda sfn://grid3.na.astro.it/flatfiles/SE00/gilda/AntonioDelgado_outputData_1
        guid:58d1e428-2f36-11d9-8a52-e231017ae683

$ lcg-lg --vo gilda sfn://grid009.ct.infn.it/flatfiles/SE00/gilda/AntonioDelgado_outputData_2
        lcg_lg: No such file or directory
```

DM Command Line Tools

Enabling Grids for E-science in Europe

We provide the GUID, as we know that the file is the same (they will share LFN also)

```
$ lcg-rf --vo gilda -g guid:58d1e428-2f36-11d9-8a52-e231017ae683 \
        sfn://grid009.ct.infn.it/flatfiles/SE00/gilda/AntonioDelgado_outputData_2
        guid:58d1e428-2f36-11d9-8a52-e231017ae683

$ lcg-lr --vo gilda lfn:AntonioDelgado_outputData
        sfn://gilda-se-01.pd.infn.it/shared/gilda/generated/2004-11-05/file56a78a17-2f36-11d9-
              8a52-e231017ae683
        sfn://grid009.ct.infn.it/flatfiles/SE00/gilda/AntonioDelgado_outputData_2
        sfn://grid3.na.astro.it/flatfiles/SE00/gilda/AntonioDelgado_outputData_1
```

DM Command Line Tools

Enabling Grids for E-science in Europe

$ lcg-cp --vo gilda lfn:AntonioDelgado_outputData file:`pwd`/retrieved

$ cat retrieved

This is the file for the OutputData JDL attribute
Nothing really interesting to say, here...
See you!

$ lcg-del --vo gilda -a lfn:AntonioDelgado_outputData
$ lcg-del --vo gilda -a guid:42405581-ec58-4c27-8b8a-9e0acf3143c3

$ lcg-lr --vo gilda lfn:AntonioDelgado_outputData

lcg_lr: No such file or directory

lcg_utils API

Enabling Grids for E-science in Europe

1. Check the syntax of the lcg_util API in the manpages and in $LCG_LOCATION/include/lcg_util.h .
2. Create an application that copies "/etc/services" and brings it to the Grid using lcg_cr method (we will call it "file1"). Submit it in a job and use the default SE.
3. Create an application that gets the LFNs, GUID, SURLs and TURLs of "file1".
4. Submit a job that retrieves this "file1" and compares it with local "/etc/services" (in the WN) to see if they are equal.

lcg_utils API

Enabling Grids for E-science in Europe

General Remarks:

- Look into the "lcg_util" folder of the provided tarball. Complete the programs of the "skeletons" folder to create yours (see "YOUR CODE" marks). In the "solutions" folder, you can find some working solutions.
- Include the provided "lcg_util.h" file in your working directory (it adds the prototype for the lcg_la function, which is missing in the installed "lcg_util.h".
- You may use the provided "Makefile" changing only the PROGRAM variable as necessary. It is important that you:
  - **Use the** g++-3.2.2 **compiler (and linker)**
  - **Link against** -lgfal -llcg_util **from** -L$(LCG_LOCATION)/lib
  - **Link against** -lglobus_gass_copy_<flavor> **from -**L$(GLOBUS_LOCATION)/lib
- Globus flavors:
  - **gcc32** → **Globus for gcc 3.2**
  - **gcc32pthr** → **Support for multi-threaded applications**
  - **gcc32dbg** → **Debuggable**
  - **gcc32dbgpthr** → **Multi-threaded debuggable**

lcg_utils API: Sample Makefile

```
######################## VARIABLES ############################
# Compiler
CC = /opt/gcc-3.2.2/bin/g++-3.2.2
CFLAGS = -g  -I /opt/lcg/include

# Linker
LD = /opt/gcc-3.2.2/bin/g++-3.2.2
LFLAGS = -L$(LCG_LOCATION)/lib/   -lgfal   -llcg_util \
              -L$(GLOBUS_LOCATION)/lib   -lglobus_gass_copy_gcc32

# Default program to compile (use "make PROGRAM=your_name" to override)
PROGRAM = copy_and_register_1
SOURCES = $(PROGRAM).cpp
OBJECTS = $(PROGRAM).o

######################## TARGETS ############################
# Default
all:  LINK  $(SOURCES)

# Linking...
LINK:  $(OBJECTS)  $(SOURCES)
       $(LD)  -o  $(PROGRAM)  $(LFLAGS)  $<

# Compiling...
%.o:  %.cpp
       $(CC)  –c  $(CFLAGS)  $<
```

Linking implies the object compilation

"$<" matches first dependency

"%.o" matches any .o file and "%.cpp" will then match the corresponding source file

lcg_utils API

Enabling Grids for E-science in Europe

Remarks:

- You need to include the "lcg_util.h" library as an extern C.
- First create an application that just copies and registers your local "/etc/services". Compare your solution with "copy_and_register_1.cpp"
- Try again, and realize that if the LFN exists already, you get an error, but the file is copied and registered in the LRC anyway. Improve the previous application in order to deal with this. Compare with "copy_and_register_2.cpp"
- Now, create a JDL file and send your application with a job, so that it gets executed in a WN.
- Modify the application so that it reads the environmental variable "VO_GILDA_DEFAULT_SE" and uses its value as the destination SE. The variable should be defined everywhere, but to be sure, you may use "grid010.ct.infn.it". Compare with "copy_and_reg_3.cpp"

lcg_utils API

Enabling Grids for E-science in Europe

Remarks:

- You should use “lcg_la”, “lcg_lr” and “lcg_gt” and the skeleton “getNames.cpp”
- Hint: consider that if the function expects char ***lfns you probably should define char **lfns and pass &lfns as the argument.
- Take into account that you may get more than one SURL, and that you should get one TURL for each of these SURLs.
- As protocol for the TURLs, we will request “gsiftp”, since we are sure that it is defined for every SE.

lcg_utils API

Enabling Grids for E-science in Europe

Remarks:

- Use the skeleton “compare.cpp”
- You need to include “fstream” for local file access and “unistd.h” for the unlink function to delete the temporary file after use.
- Use the provided function for comparison of the files:
  bool compareFiles(const char * pFile1, const char * pFile2)
- Hint: First try copying the file to the “/tmp” directory of the WN.
- Afterwards you may try copying it to the current directory.
  Hint: Get that directory from the $PWD environmental variable.

GFAL API

Enabling Grids for E-science in Europe

1. Check the syntax of the GFAL API in the manpages and in $LCG_LOCATION/include/gfal_api.h .

2. Submit a job that opens the previously created “file1” and reads only the first 20 lines. Retrieve them with the standard output of the job.

3. Submit a job that creates a file, writes the date in it every second for 10 seconds, and registers the file.

4. Compare access time for files in CASTOR that have been pre-staged and files only in tape. Create an application that asks for the stage of a file. Enhance the application so that it waits for the staging to complete, and then copies the file to the local filesystem.

GFAL API

Enabling Grids for E-science in Europe

General Remarks:

- Use skeletons and solutions of the "gfal" folder.
- Remember to use double slash in the SURLs to be used with RFIO under GFAL.
- Most programs will not work directly from the UI. You have to send them in a job to be executed in a WN. They will not be able to access SEs in different sites either.

GFAL API

Enabling Grids for E-science in Europe

Remarks:

- You need to include the "gfal.h" library as an "extern C".
- You should open the file and read characters until the 20th line is reached.
- Hint: read one char a time and compare it to the newline character.

GFAL API

Enabling Grids for E-science in Europe

Remarks:

- First, create an application that writes the file. Use the skeleton of "gfal_time_write.cpp" for the timing tasks (need to include "time.h" and "unistd.h"). As destination filename, a SURL has to be used.

- Second, create an application that registers the created file (providing its SURL), using register_pfn and create_alias (since register_alias is used for extra LFNs). Use functions from "uuis/uuid.h" for the manul GUID generation, as shown in the skeleton of "gfal_time_reg.cpp".

  Hint: You may use gfal_stat, to check the file existence and its size.

- Now, create an application that performs this registration using lcg_utils. See that this other way is simpler. "gfal_time_reg_lcg.cpp"

- Integrate both creation/writing and registration in a single program. See "gfal_time_complete.cpp".

GFAL API

Enabling Grids for E-science in Europe

This exercise will only be demonstrated !

Remarks:

- Some files "test_xx" are stored in "/castor/cern.ch/grid/dteam/testAntoSRM", and are accessible through the SE "castorsrm.cern.ch".
- Some of the files are not staged in the disk pool (check with the command stageqry). Try to access the files with lcg-cp and see that the command takes really long (if it completes).
- Check "gfal_stage.cpp", which stages a file with srm_get(), so that it can be accessed with lcg-cp later on.
- Check "gfal_stage_check.cpp", which asks for the staging of a file and waits for the process to be completed, checking periodically with srm_getstatus().
- Check "gfal_stage_copy.cpp", which waits for the staging of a file, and afterwards copies the file to the local filesystem with lcg-cp.

Globus API

Enabling Grids for E-science in Europe

1. Check the implementation of “existsFile”, which is equivalent to “edg-gridftp-ls”.

2. Create a command “listDir <url>” that lists the entries in a given directory of an Storage Element; i.e. equivalent to “edg-gridftp-ls”.

(See the DM-HandsOn presentation)

Globus API

Enabling Grids for E-science in Europe

General Remarks:

- Use skeletons and solutions of the "globus" folder.
- Use the provided Makefile (with globus headers included)
- Check the API documentation in:
  http://www-unix.globus.org/api/c/globus_ftp_client/html
- Review first the "existsFile" example.

Globus API

Enabling Grids for E-science in Europe

Remarks:

- Two functions are necessary:
  - **globus_ftp_client_verbose_list**: starts the listing transfer and blocks until all the data has been retrieved and the callbacks called
  - **globus_ftp_client_register_read**: retrieves actual data in blocks of defined size (indicating EOF when the end is reached)

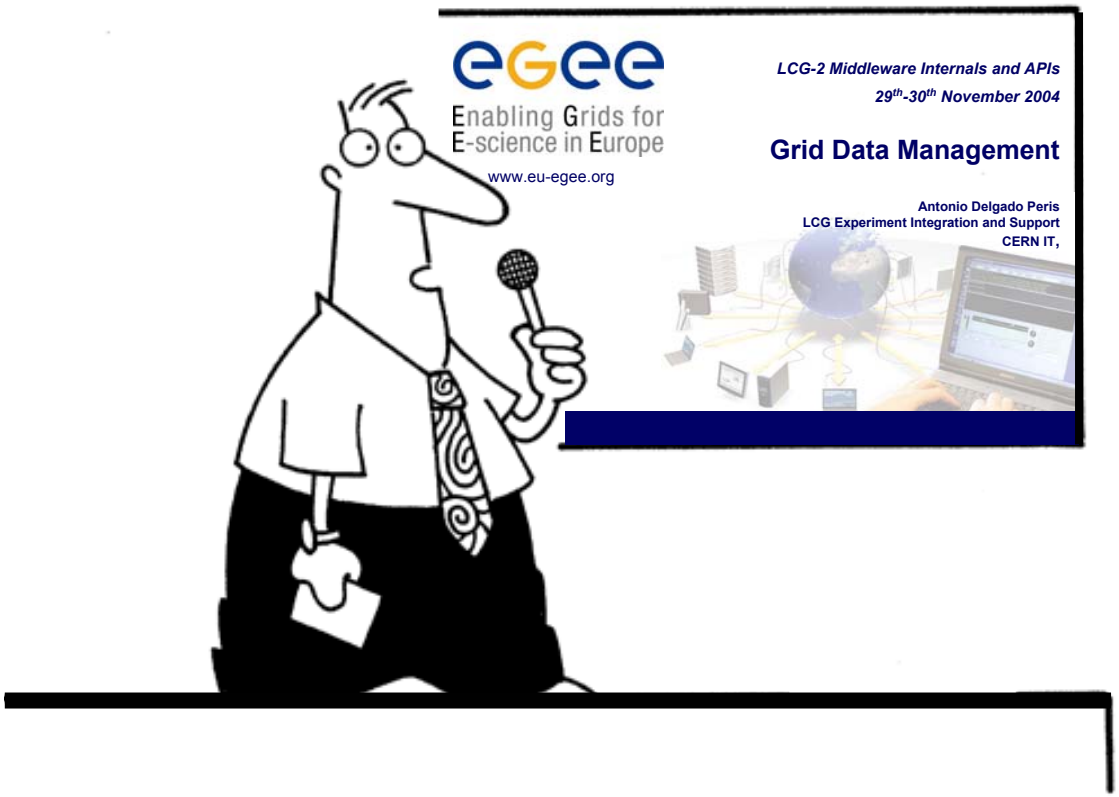
- Need two different callback functions

```
listCallback ( void* user_cb_arg,          globus_ftp_client_handle_t *handle,
               globus_object_t *error);

readCallback ( void* user_cb_arg,          globus_ftp_client_handle_t  *handle,
               globus_object_t *error,     globus_byte_t *buffer,
               globus_size_t length,       globus_off_t offset,
               globus_bool_t eof);
```

- Hint: You probably should also enhance the CallbackData class with some simple “signal()” and “wait()” function members

The End


Enabling Grids for
E-science in Europe
www.eu-egee.org
LCG-2 Middleware Internals and APIs
29th-30th November 2004
Grid Data Management
Antonio Delgado Peris
LCG Experiment Integration and Support
CERN IT,