



Enabling Grids for E-science

# GILDA Practicals

*Giuseppe Andronico*

*Roberto Barbera*

*Mike Mineter*

*EGEE Tutorial, Taipei, 22-23.08.2005*

[www.eu-egee.org](http://www.eu-egee.org)



Information Society



- **Proxy/Myproxy Management**
- **Workload Management**
  - **Job Description Language**
  - **Commands**
  - **Examples and Exercises**
- **Data Management (LFC and FireMan)**
  - **Commands**
  - **Examples and Exercises**
- **Explore the GILDA Testbed**
- **R-GMA**
  - **Commands and Exercises**

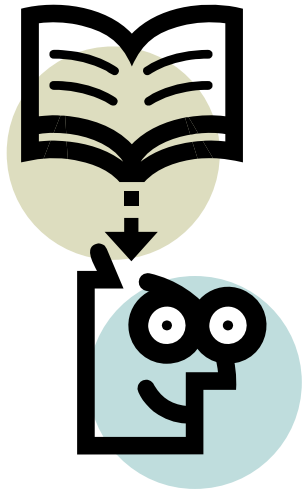
**Host: [glite-tutor.ct.infn.it](https://glite-tutor.ct.infn.it)**

**Username: taipeiXX (XX=01...60)**

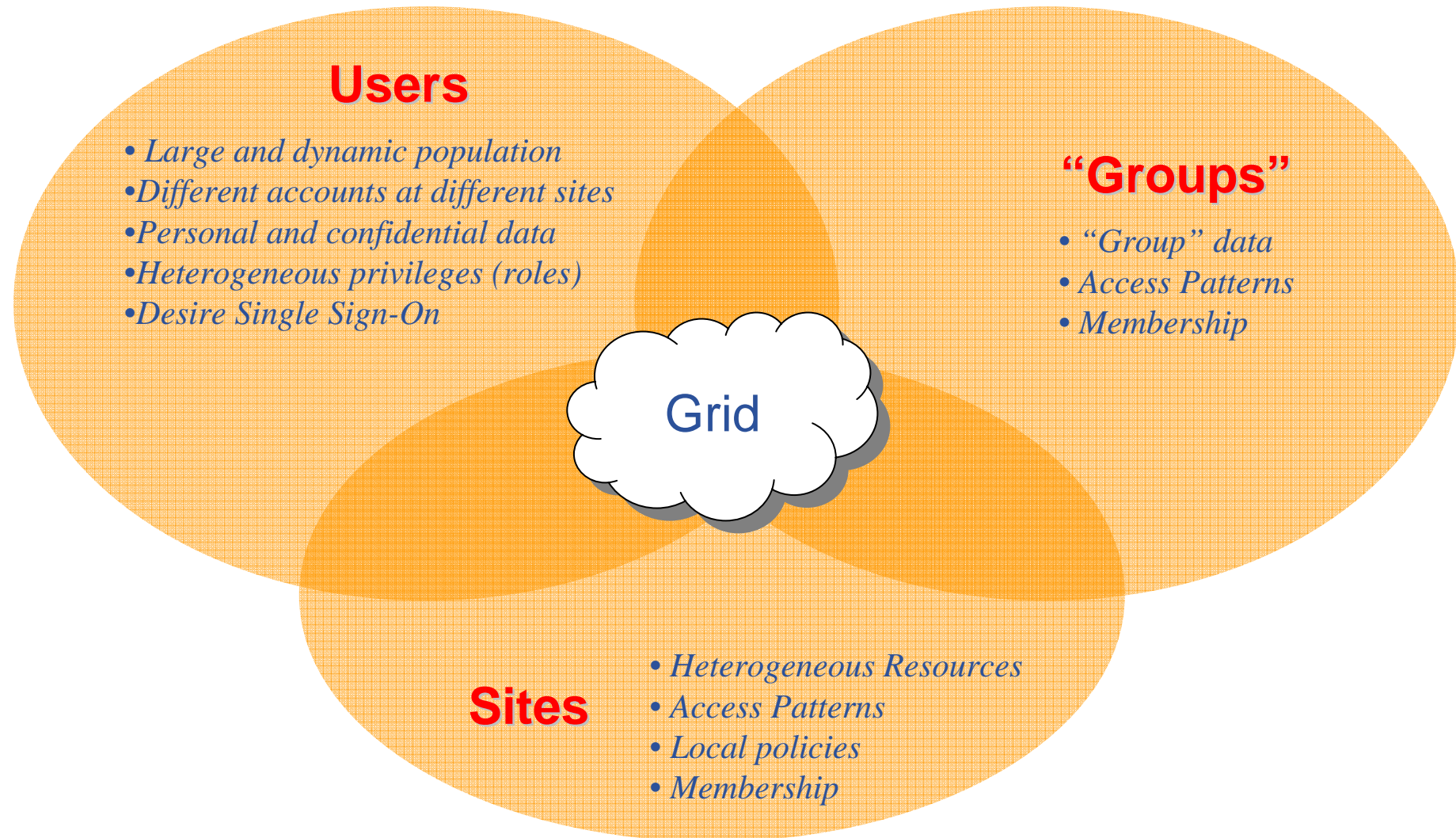
**Password: GridTAIXX (XX=01...60)**

**PassPhrase: TAIPEI**

**GENIUS: <https://glite-tutor.ct.infn.it>**



# Proxy/MyProxy Management



The goal of authorization and authentication of users and resources is done through digital certificates, in X.509 format

## Certification Authority (CA)

- Issue Digital Certificates for users and machines
- Check the identity and the personal data of the requestor
  - Registration Authorities (RAs) do the actual validation
- CA's periodically publish a list of compromised certificates
  - **Certificate Revocation Lists (CRL)**: contain all the revoked certificates yet to expire
- CA certificates are **self-signed**

For each player, a CA guarantees its authenticity with a certificate

- Digital certificates are split in public/private keys
- Public key is spread along the net, while the private stays encrypted on the disk
- Default location for public/private keys is `$HOME/.globus` (attention to file permissions)

```
ls -l $HOME/.globus
```

```
-rw-r--r--      1 local      local      1143 Jun 30
 16:01 usercert.pem
-r-----      1 local      local      963 Jun 30
 16:01 userkey.pem
```

To get information on your certificate, run

```
> openssl x509 -in .globus/usercert.pem -noout -text
```

Certificate:

Data:

Version: 3 (0x2)

Serial Number: 1783 (0x6f7)

Signature Algorithm: md5WithRSAEncryption

Issuer: C=IT, O=GILDA, CN=GILDA Certification

Authority

Validity

Not Before: Jun 30 07:14:13 2005 GMT

Not After : Jul 30 07:14:13 2005 GMT

Subject: C=IT, O=GILDA, OU=Personal Certificate,  
L=TAIPEI, CN=TAIPEI20/Email=roberto.barbera@ct.infn.it

.....



- **GSI extension to X.509 Identity Certificates**
  - signed by the normal end entity cert (or by another proxy)
- **Support some important features**
  - Delegation and Mutual authentication
- **Has a limited lifetime (minimized risk of “compromised credentials”)**
- **It is created by the grid-proxy-init command:**

**Grid Pass Phrase:  
TAIPEI**

```
> grid-proxy-init
```

```
Your identity: /C=IT/O=GILDA/OU=Personal
Certificate/L=TAIPEI/CN=TAIPEI20/Email=roberto.barbera@ct.infn.it
```

```
Enter GRID pass phrase for this identity:
```

```
Creating proxy
```

```
.....
..... Done
```

```
Your proxy is valid until: Mon Jul 18 07:14:28 2005
```

- **By `grid-proxy-info` you can inspect info about your proxy**

```
>grid-proxy-info -all
```

```
subject   : /C=IT/O=GILDA/OU=Personal
           Certificate/L=TAIPEI/CN=TAIPEI20/Email=roberto.barb
           era@ct.infn.it/CN=proxy
```

```
issuer    : /C=IT/O=GILDA/OU=Personal
           Certificate/L=TAIPEI/CN=TAIPEI20/Email=roberto.barb
           era@ct.infn.it
```

```
identity  : /C=IT/O=GILDA/OU=Personal
           Certificate/L=TAIPEI/CN=TAIPEI20/Email=roberto.barb
           era@ct.infn.it
```

```
type      : full legacy globus proxy
```

```
strength  : 512 bits
```

```
path      : /tmp/x509up_u500
```

```
timeleft  : 11:57:24
```

- **Proxy has limited lifetime (default is 12 h)**
  - Bad idea to have longer proxy
- **However, a grid task might need to use a proxy for a much longer time**
  - Grid jobs in HEP Data Challenges last up to 2 days
- **myproxy server:**
  - Allows to create and store a long term proxy certificate:
  - **-s <host\_name>** specifies the hostname of MyProxy server
  - **-l <user>** define user that will own **remote** credentials
  - **myproxy-init -s <host\_name> -l <user>**
  - **myproxy-info -s <host\_name> -l <user>**
    - Get information about stored long living proxy
  - **myproxy-get-delegation -s <host\_name> -l <user>**
    - Get a new proxy from MyProxy server
  - **myproxy-destroy -l <user> -s <host\_name>**
    - Destroy the credential into the server
  - Check out the **myproxy-xxx --help** option
- **A dedicated service on the RB can renew automatically the proxy**
  - contacts the myproxy server

# Store credentials on MyProxy Server

```
> grid-proxy-destroy remove local credentials
> myproxy-init -s grid001.ct.infn.it -l <UniqueUsername>
Your identity: /C=IT/O=GILDA/OU=Personal Certificate/L
=TAIPEI/CN=<UniqueUsername>/Email=
roberto.barbera@ct.infn.it
Enter GRID pass phrase for this identity:
Creating proxy .....Done
Proxy Verify OK
Your proxy is valid until: Sun Jul 24 18:53:44 2005
Enter MyProxy pass phrase:
Verifying password - Enter MyProxy pass phrase:
A proxy valid for 168 hours (7.0 days) for user
<UniqueUsername> now exists on grid001.ct.infn.it.
```

Now your credentials are stored on MyProxy server, and are available for delegation or renewal by RB.

**ATTENTION! <UniqueUsername> MUST BE your PERSONAL username**

```
> myproxy-get-delegation -s grid001.ct.infn.it -l
  <UniqueUser>
```

Enter MyProxy pass phrase:

```
A proxy has been received for user <UniqueUser> in
  /tmp/x509up_u500
```

```
> grid-proxy-info -all
```

```
subject  : /C=IT/O=GILDA/OU=Personal Certificate/L=TAIPEI/CN=
  <UniqueUser>/Email=roberto.barbera@ct.infn.it
```

```
/CN=proxy/CN=proxy/CN=proxy
```

```
issuer   : /C=IT/O=GILDA/OU=Personal Certificate/L=TAIPEI/CN=
  <UniqueUser>/Email=roberto.barbera@ct.infn.it
```

```
/CN=proxy/CN=proxy
```

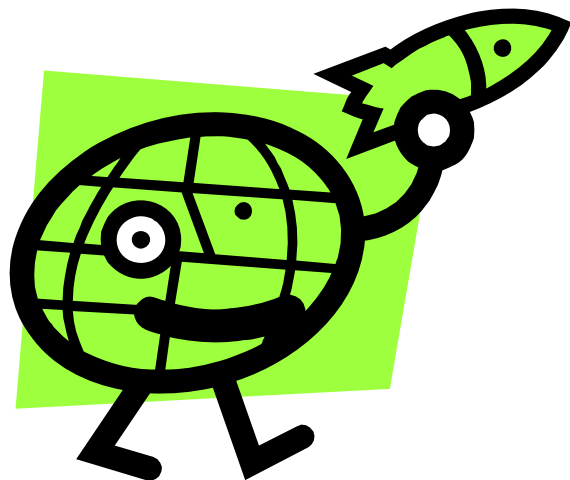
```
identity : /C=IT/O=GILDA/OU=Personal Certificate/L=TAIPEI/CN=
  <UniqueUser>/Email=roberto.barbera@ct.infn.it
```

```
type      : full legacy globus proxy
```

```
strength  : 512 bits
```

```
path      : /tmp/x509up_u500
```

```
timeleft  : 11:56:58
```



# Workload Management System

- The user interacts with Grid via a **Workload Management System (WMS)**
- The **Goal of WMS** is the distributed scheduling and resource management in a Grid environment.
- **What does it allow Grid users to do?**
  - To submit their jobs
  - To execute them on the “best resources”
    - The WMS tries to optimize the usage of resources
  - To get information about their status
  - To retrieve their output

- **Information to be specified when a job has to be submitted:**
  - Job characteristics
  - Job requirements and preferences on the computing resources
    - Also including software dependencies
  - Job data requirements
- **Information specified using a Job Description Language (JDL)**
  - Based upon Condor's *CLASSified ADvertisement language (ClassAd)*
    - Fully extensible language
    - A ClassAd
      - *Constructed with the classad construction operator []*
      - *It is a sequence of attributes separated by semi-colon (;).*
- **So, the JDL allows definition of a set of attribute, the WMS takes into account when making its scheduling decision**



- An attribute is a pair (key, value), where value can be a Boolean, an Integer, a list of strings, ....
  - <attribute> = <value>;
- In case of literal string for values:
  - if a string itself contains double quotes, they must be escaped with a backslash
    - `Arguments = " \"Hello\" 10";`
  - the character “” cannot be specified in the JDL
  - special characters such as &, |, >, < are only allowed
    - if specified inside a quoted string
    - if preceded by triple \
      - `Arguments = "-f file1\\\"&file2";`
- Comments must be preceded by a sharp character (#) or have to follow the C++ syntax
- The JDL is sensitive to blank characters and tabs
  - they should not follow the semicolon (;) at the end of a line

- The supported attributes are grouped in two categories:
  - Job Attributes
    - Define the job itself
  - Resources
    - Taken into account by the RB for carrying out the matchmaking algorithm (to choose the “best” resource where to submit the job)
    - *Computing Resource*
      - *Used to build expressions of Requirements and/or Rank attributes by the user*
      - *Have to be prefixed with “other.”*
    - *Data and Storage resources (see talk Job Services With Data Requirements)*
      - *Input data to process, SE where to store output data, protocols spoken by application when accessing SEs*

## JobType

*Normal* (simple, sequential job), *Interactive*, *MPICH*, *Checkpointable*

Or combination of them

## Executable (mandatory)

The command name

## Arguments (optional)

Job command line arguments

## StdInput, StdOutput, StdError (optional)

Standard input/output/error of the job

## InputSandbox (optional)

List of files on the UI local disk needed by the job for running

The listed files will automatically staged to the remote resource

## OutputSandbox (optional)

List of files, generated by the job, which have to be retrieved

## VirtualOrganisation (optional)

A different way to specify the VO of the user

- **glite-job-submit** performs the job submission to the WMS

Usage: **glite-job-submit** *[options]* <jdl file>



## Principal Options :

- vo** <vo name> : perform submission with a different VO than the UI default one
- output, -o** <output file> save jobld on a file, instead of STDIN
- resource, -r** <resource value>, specify the resource for execution (needs the GLUE Uniqueld of the queue, obtainable with **list-match**)
- debug** show function calls and parameters

- **glite-job-status <job id>**  
check job execution status
- **glite-job-output <job id>**  
If job status is done, allows output retrieve
- **glite-job-cancel <job id>**  
perform job deletion

All of these commands accepts (with the option **-i <file>**)  
input from a file.

```
glite-job-status -i myjobId
```

```
Type = "Job";  
JobType = "Normal";  
Executable = "/bin/bash";  
StdOutput = "std.out";  
StdError = "std.err";  
InputSandbox = {"yourscript.sh"};  
OutputSandbox = {"std.err", "std.out"};  
Arguments = "yourscript.sh";
```

```
>cd UIPnPcomb/example_JDL
```

Create a bash script which displays hostname and current date

Save the script as yourscrip.sh

## Simple job submission

```
cp hostname.jdl exercise1.jdl
```

Modify exercise1.jdl file

Instead of running hostname command, run a bash script you have just created (yourscrip.sh).

Submit the job, check its status and when done retrieve the output

- Requirements
  - Job requirements on the resources
  - Specified using GLUE attributes of resources published in the Information Service
  - Its value is a boolean expression
  - **Only one requirements can be specified**
    - if there are more than one, only the last one is taken into account
    - If you need several Requirements, combine them through logical operators (&&, ||, !, .....
  - If not specified, default value defined in UI configuration file is considered
    - Default: *other.GlueCEStateStatus == "Production"* (the resource has to be able to accept jobs and dispatch them on WNs)



```
#Insert a requirement to parse only the short queues.
Requirements = (other.GlueCEPolicyMaxWallClockTime >
    720);
```

```
#Insert a requirement to parse only the long queues.
Requirements = (other.GlueCEPolicyMaxWallClockTime >
    1440);
```

```
#Insert a requirement to parse only the infinite
    queues.
Requirements = (other.GlueCEPolicyMaxWallClockTime >
    2880);
```

```
#Insert a requirement to steer the execution on a
    particular CE Queue.
Requirements = other.GlueCEUniqueID ==
    "grid010.ct.infn.it:2119/jobmanager-lcgpbs-long";
```

- **glite-job-list-match** allows to check the suitable resources for execution
- No job submission is performed, just listmatch is performed
- Usage: **glite-job-list-match** *[options]* <jdl file>



- Principal Options :

- vo <vo name> : perform list-match with a different VO than the UI default one
- rank show resources in order of ranking
- output, -o <output file> redirect output on a file, instead of STDIN
- debug show function calls and parameters

- **Simple job using Requirements**
  - Modify exercise1.jdl file so that user with a even workstation number will submit their job on a “long” queue, and the other to an “infinite” one
  - Verify the list of CE suitable for this job execution
  - Submit the job, check its status and retrieve the output

```

Type = "Job";
JobType = "Normal";
Executable = "/bin/sh";
StdOutput = "povray_cubo.out";
StdError = "povray_cubo.err";
InputSandbox = {"start_povray_cubo.sh", "cubo.pov"};
OutputSandbox =
    {"povray_cubo.out", "povray_cubo.err", "cubo.png"};
RetryCount = 7;
Arguments = "start_povray_cubo.sh";
Requirements = Member("POVRAY-
    3.5", other.GlueHostApplicationSoftwareRunTimeEnviron
    ment);

```

```
#!/bin/bash
mv cubo.pov OBJECT.POV #rename input file
/usr/bin/povray /usr/share/povray-
  3.5/ini/res800.ini #run povray
mv OBJECT.png cubo.png #rename output file
```

- **cd UIPnPcomb/example\_JDL**
- **Modify povray\_cubo.jdl, specifying the resource for execution into the jdl file**
- **Check job status and when done retrieve the output**
- **Display .png file obtained as output, using ImageMagick**

- It is a mechanism by which a job can access at some information about itself...at execution time!
- The Resource Broker creates and attaches this file to the job when it is ready to be transferred to the resource that best matches the request.
- Two ways for parsing elements from .BrokerInfo file:
  - 1) Directly from the Worker Node at execution time;
  - 2) From User Interface, but only if you have inserted the name of “.BrokerInfo” file in the JDL’s OutputSandbox, and you have just retrieved job output, once that job has been Done;

**edg-brokerinfo** [options] function param



# Example of .BrokerInfo file

```
[
  ComputingElement =
  [
    CloseStorageElements =
    {
      [
        GlueSAStateAvailableSpace =
        14029724;
        GlueCESEBindCEAccesspoint
        = "/flatfiles/SE00";
        mount =
        GlueCESEBindCEAccessPoint;
        name = "grid003.cecalc.ula.ve";
        freespace =
        GlueSAStateAvailableSpace
      ]
    };
    name =
    "grid006.cecalc.ula.ve:2119/jobmanager-
    lcgpbs-infinite"
  ];
  InputFNs =
  {
  };
  StorageElements =
  {
  };
  VirtualOrganisation = "gilda" ]
```

```
edg-brokerinfo getCE
edg-brokerinfo
getDataAccessProtocol
edg-brokerinfo getInputData
edg-brokerinfo getSEs
edg-brokerinfo getCloseSEs
edg-brokerinfo getSEMMountPoint
<SE>
edg-brokerinfo getSEFreeSpace <SE>
edg-brokerinfo getSEProtocols <SE>
edg-brokerinfo getSEPort <SE>
<Protocol>
edg-brokerinfo getVirtualOrganization
edg-brokerinfo getAccessCost
```



## Exercise 1

- Create a file called **startScriptBrokerInfo.sh** with this content:

```
#!/bin/sh
```

```
MY_NAME="Your name"
```

```
WORKER_NODE_NAME=`hostname`
```

```
echo "Hello $MY_NAME, from $WORKER_NODE_NAME"
```

```
ls -a
```

```
echo "This job is running on this CE: "
```

```
/opt/edg/bin/edg-brokerinfo getCE
```

## Exercise 2

- Create a file called **scriptBrokerInfo.jdl** with this content:

```
[  
Executable = "startScriptBrokerInfo.sh";  
  StdOutput = "std.out";  
  StdError = "std.err";  
  VirtualOrganisation = "gilda";  
  InputSandbox = {"startScriptBrokerInfo.sh"};  
  OutputSandbox = {"std.out", "std.err", ".BrokerInfo"};  
  RetryCount = 7;  
]
```

1. Replace your name in the file script `startScriptBrokerInfo.sh`;
2. Submit / Query the status / Retrieve Output the JDL file  
`scriptBrokerInfo.jdl`;
3. In JobOutput folder, go into directory of the job that you have just retrieved and inspect the `.BrokerInfo` file.
4. Take practice with the `edg-brokerinfo` command and its functions.



# Practicals on Data Management (LFC and lcg-utils)

- Set the following environment variables to specify the catalog type and its location:
  - `export LCG_CATALOG_TYPE=lfc`
  - `export LFC_HOST=lfc-gilda.ct.infn.it`
- Ensure you have created a proxy certificate and it is still valid. If not create it by:
  - `grid-proxy-init`
- Remember: The Passphrase is **TAIPEI**

## Listing the entries of a LFC directory

*lfc-ls [-cdiLIRTu] [--comment] path...*

where *path* specifies the LFC pathname (mandatory)

- Remember that **LFC** has a **directory tree structure**
- */grid/<VO\_name>/<you create it>*



- All members of a given VO have read-write permissions under their directory
- **-l** (it is a lowercase “L”) outputs long listing
- **-R** lists the contents of directories recursively (**don't use it so often!**)
- You can set **LFC\_HOME** to use relative paths  
*LFC\_HOME=/grid/gilda/myDir* → */grid/gilda/myDir/myFile*  
 becomes *myFile*

```
$ lfc-ls -l /grid/gilda
```

```
...
-rw-rw-r-- 1 4401 4400 0 Jun 21 09:40 tutor02-rel-pippo-pluto
-rw-rw-r-- 1 4401 4400 0 Jun 21 09:39 tutor14
-rw-rw-r-- 1 4401 4400 0 Jun 21 09:40 tutor16-mytxt
-rw-rw-r-- 1 4401 4400 0 Jun 21 09:32 unitprot-ibcp02
-rw-rw-r-- 1 4401 4400 0 Jun 21 09:36 uploadfile
-rw-rw-r-- 1 4401 4400 0 Jun 21 09:36 uploadfilefn
-rw-rw-r-- 1 4401 4400 0 Jun 21 09:38 user.example
-rw-rw-r-- 1 4401 4400 0 Jun 21 09:38 user.example2
-rw-rw-r-- 1 4401 4400 0 Jun 21 09:40 valencia15.ejemplo
-rw-rw-r-- 1 4401 4400 0 Jun 21 09:40 valencia15.example
...
```

```
$ export LFC_HOME=/grid/gilda/
```

```
$ lfc-ls -l user.example
```

```
-rw-rw-r-- 1 4401 4400 0 Jun 21 09:38 /grid/gilda/user.example
```

## Creating a symbolic link

***lfc-ln -s file linkname***

***lfc-ln -s directory linkname***

Create a link to the specified *file* or *directory* with *linkname*

– *Example:*

***\$ lfc-ln -s /grid/gilda/user.example /grid/gilda/taipei/linkToUser.ex***



Let's check the link using `lfc-ls` with long listing (`-l`)

***\$ lfc-ls -l /grid/gilda/taipei***

```
lrwxrwxrwx  1 4404    4400  0 Jul 17 12:06 linkToUser.ex ->
  /grid/gilda/user.example
```



## Creating directories in the LFC

***lfc-mkdir [-m mode] [-p] path...***

- Where *path* specifies the LFC pathname
- Remember that while registering a new file (using `lfc-cr`, for example) the corresponding destination directory must be created in the catalog before

- Examples:

***\$ lfc-mkdir /grid/gilda/Examples***

You can just check the directory with:

***\$ lfc-ls -l /grid/gilda***

## Adding/deleting metadata information

***lfc-setcomment path comment***

***lfc-delcomment path***

*lfc-setcomment* adds/replaces a *comment* associated with a file/directory in the LFC Catalog

*lfc-delcomment* deletes a comment previously added

- Example:

***lfc-setcomment /grid/gilda/user.example "Hello Taipei"***

- Check your job with..

***lfc-ls --comment /grid/gilda/user.example***

```
lfc-ls --comment /grid/gilda/user.example
/grid/gilda/user.example Hello Taipei
```

- Example:

```
lfc-delcomment /grid/gilda/user.example
```

- Check your job with..

```
lfc-ls -l --comment /grid/gilda/user.example
```

```
-rw-rw-r-- 1 4401 4400 0 Jun 21 09:38 /grid/gilda/user.example
```

## Exercise No.1:

- Log onto an UI and initialize your proxy credentials if not already done
- set up properly the environment variables to use lfc-gilda.ct.infn.it catalog
- have a look inside the catalog
- create a directory with your surname
- put inside the just created dir a link to an existing file
- add a comment to that file and verify it

## Summary of the LFC Catalog commands

<b>lfc-chmod</b>	<b>Change access mode of the LFC file/directory</b>
<b>lfc-chown</b>	<b>Change owner and group of the LFC file-directory</b>
<b>lfc-delcomment</b>	<b>Delete the comment associated with the file/directory</b>
<b>lfc-getacl</b>	<b>Get file/directory access control lists</b>
<b>lfc-ln</b>	<b>Make a symbolic link to a file/directory</b>
<b>lfc-ls</b>	<b>List file/directory entries in a directory</b>
<b>lfc-mkdir</b>	<b>Create a directory</b>
<b>lfc-rename</b>	<b>Rename a file/directory</b>
<b>lfc-rm</b>	<b>Remove a file/directory</b>
<b>lfc-setacl</b>	<b>Set file/directory access control lists</b>
<b>lfc-setcomment</b>	<b>Add/replace a comment</b>

- The LCG Data Management tools (usually called *lcg-utils*) allow users to copy files between UI, CE, WN and a SE, to register entries in the File Catalogs and replicate files between SEs.
- Check if LCG\_GFAL\_INFOSYS environment variable is correctly set to the local GILDA Information Index (BDII)
  - **export LCG\_GFAL\_INFOSYS=grid004.ct.infn.it:2170**

## Upload a file to a SE and register it into the catalog

- `lcg-cr -d dest_file | dest_host -l lfn [-g guid] [-l lfn]`  
`[-v | --verbose] --vo vo src_file`

where

- **dest\_host** is the fully qualified hostname of the destination SE
- **dest\_file** is a valid SURL (both `sfn://` or `srm://` format are valid)
- **guid** specifies the Grid Unique Identifier. If this option is not present, a GUID is generated internally
- **lfn** specifies the Logical File Name associated with the file
- **vo** specifies the Virtual Organization the user belongs to
- **src\_file** specifies the source file name: the protocol can be `file:///` or `gsiftp:///`

- To discover which SEs the user is allowed to use, remember you can use **lcg-infosites** command.

```
lcg-infosites --vo gilda se
```

The output is a list of SEs and related information on available/used space

- lcg-cr** usage example:

```
$ lcg-cr -v -d grid-se.bio.dist.unige.it -l lfn:/grid/gilda/taipei/note.txt -
  -vo gilda file:///home/local/note.txt
Using grid catalog type: lfc
Source URL: file:///home/local/note.txt
File size: 51
Destination specified: grid-se.bio.dist.unige.it
Destination URL for copy: gsiftp://grid-
  se.bio.dist.unige.it/flatfiles/SE00/gilda/generated/2005-07-17/file1f0e73d8-7e3f-
  47d1-bc95-c03c92aae569
# streams: 1
Alias registered in Catalog: lfn:/grid/gilda/taipei/note.txt
Transfer took 11320 ms
Destination URL registered in Catalog: sfn://grid-
  se.bio.dist.unige.it/flatfiles/SE00/gilda/generated/2005-07-17/file1f0e73d8-7e3f-
  47d1-bc95-c03c92aae569
guid:4c10a8e3-2244-4c38-bc98-ed98ae7cb94e
```



## Adding an alias for a given GUID

**lcg-aa --vo vo guid lfn**

where

- **vo** specifies the Virtual Organization the user belongs to
- **guid** specifies the Grid Unique Identifier of the file you want to add the alias to
- **lfn** specifies the new alias

- *Example:*

```
$ lcg-aa --vo gilda guid:4c10a8e3-2244-4c38-bc98-ed98ae7cb94e
lfn:/grid/gilda/taipei/aliasToNote.txt
```

- To check if the previous command was successful, you can use **lcg-la** command to **list the aliases for a given LFN, GUID or SURL**

```
$ lcg-la --vo gilda
lfn:/grid/gilda/taipei/aliasToNote.txt
```

```
lfn:/grid/gilda/taipei/note.txt
```

```
lfn:/grid/gilda/taipei/aliasToNote.txt
```

## Exercise No.2:

- verify that your **LCG\_GFAL\_INFOSYS** is correctly set up
- create a dummy file
- check the available storage elements
- copy and register the previous created file into your previously created dir
- add an alias to the just uploaded file
- check if the alias was assigned correctly

## Copying a file from one SE to another one and register it in the Catalog

```
lcg-rep -d dest_file | dest_host [-v | --verbose] --vo vo src_file
```

where

- **dest\_host** is the fully qualified hostname of the destination SE
- **dest\_file** is a valid SURL (both sfn:// or srm:// are valid)
- **vo** specifies the Virtual Organization the user belongs to
- **src\_file** specifies the source file name: the protocol can be LFN, GUID or SURL. An SURL scheme can be sfn: for a classical SE or srm:

```
$ lcg-rep -v -d grid009.ct.infn.it --vo gilda lfn:/grid/gilda/taipei/note.txt
```

```
Using grid catalog type: lfc
```

```
Source URL: lfn:/grid/gilda/taipei/note.txt
```

```
File size: 51
```

```
Destination specified: grid009.ct.infn.it
```

```
Source URL for copy: gsiftp://grid-se.bio.dist.unige.it/flatfiles/SE00/gilda/generated/2005-07-17/file1f0e73d8-7e3f-47d1-bc95-c03c92aae569
```

```
Destination URL for copy: gsiftp://grid009.ct.infn.it/flatfiles/SE00/gilda/generated/2005-07-17/file4f3b4cb2-b5fe-467e-9a3e-1ef602465a17
```

```
# streams: 1
```

```
Transfer took 2410 ms
```

```
Destination URL registered in LRC: sfn://grid009.ct.infn.it/flatfiles/SE00/gilda/generated/2005-07-17/file4f3b4cb2-b5fe-467e-9a3e-1ef602465a17
```

## Listing of replicas for a given LFN, GUID or SURL

**lcg-lr --vo vo file**

where

- **vo** specifies the Virtual Organization the user belongs to
- **file** specifies the Logical File Name, the Grid Unique Identifier or the Site URL. An SURL scheme can be sfn: for a classical SE or srm:

- **Example:**

**\$ lcg-lr --vo gilda lfn:/grid/gilda/taipei/note.txt**

```
sfn://grid-se.bio.dist.unige.it/flatfiles/SE00/gilda/generated/2005-07-17/file1f0e73d8-7e3f-47d1-bc95-c03c92aae569
```

```
sfn://grid009.ct.infn.it/flatfiles/SE00/gilda/generated/2005-07-17/file4f3b4cb2-b5fe-467e-9a3e-1ef602465a17
```

or we got the same output using its GUID

**\$ lcg-lr --vo gilda guid:4c10a8e3-2244-4c38-bc98-ed98ae7cb94e**

## Deleting replicas

- `lcg-del [ -a ] [ -s se ] [ -v | --verbose ] --vo vo file`

where

- **a** is used to delete all replicas of the given file
- **se** specifies the SE from which you want to remove the replica
- **vo** specifies the Virtual Organization the user belongs to
- **file** specifies the Logical File Name, the Grid Unique Identifier or the Site URL. An SURL scheme can be sfn: for a classical SE or srm:.

## Example:

- delete one replica
 

```
$ lcg-del --vo gilda -s grid009.ct.infn.it
  lfn:/grid/gilda/taipei/note.txt
```
- delete all the replicas
 

```
$ lcg-del -a --vo gilda lfn:/grid/gilda/taipei/note.txt
```
- let's check if the previous command was successful
 

```
$ lcg-lr --vo gilda lfn:/grid/gilda/taipei/note.txt
  lcg_lr: No such file or directory
```
- or by `lfs-ls /grid/gilda/taipei` (you will not see anymore note.txt and its alias)

## Downloading a Grid file in a SE to a local destination

```
lcg-cp [ -v | --verbose ] --vo vo src_file dest_file
```

where

- **vo** specifies the Virtual Organization the user belongs to
- **src\_file** specifies the source file name: the protocol can be LFN, GUID, SURL or local file. An SURL scheme can be sfn: for a classical SE or srm:
- **dest\_file** specifies the destination. The protocol can be file:/// or gsiftp://

### Example:

```
$ lcg-cp --vo gilda lfn:/grid/gilda/taipei/note2.txt
file:/home/local/note2.txt
```

```
Source URL: lfn:/grid/gilda/taipei/note2.txt
```

```
File size: 51
```

```
Source URL for copy:
```

```
gsiftp://gilda-se-01.pd.infn.it/shared/gilda/generated/2005-07-
17/file06c3b28c-465f-489c-be3c-b68728e1ca16
```

```
Destination URL: file:/home/local/note2.txt
```

```
# streams: 1
```

```
Transfer took 1060 ms
```

## Exercise No.3:

- Create two replicas of the file you previously uploaded (you could also use the alias to point it out)
- Check if the operation was successful
- Download the file back in your UI
- Delete just one replica and verify that
- Delete all the replicas and verify that
- Verify if the entry is still into the catalog

- **GOAL:**

Submit a job that does data management: it will retrieve a file previously registered into the catalog.

- **Steps to follow up:**

- Create a new file in your UI and put some data into it
- Choose a SE to upload the file to (hint: use **lcg-infosites**) and use the appropriate command to accomplish at this operation (**lcg-cr -v -vo gilda -l lfn:/grid/gilda/taipei/<choose an lfn> -d <an SE host> file:`pwd`/<your new file>**)
- create a script.sh file with the following content:

```
#!/bin/sh
/bin/hostname
#Change the LFN_NAME to download from the Catalog.
echo "Start to download.."
lcg-cp --vo gilda lfn:/grid/gilda/taipei/<lfn you choose> file:`pwd`/output.dat
echo "Done.."
```



- **Create the JobWithData.jdl:**

```
Type = "job";
JobType = "Normal";

Executable = "/bin/sh";
Arguments = "script.sh";

VirtualOrganisation = "gilda";

StdOutput = "std.out";
StdError = "std.err";

InputSandbox = {"script.sh"};
OutputSandbox = {"std.out", "std.err", "output.dat"};
```

- **Submit it to the grid**
- **Retrieve the output and verify the content of output.dat**

## Replica Management

<b>lcg-cp</b>	<b>Copies a grid file to a local destination</b>
<b>lcg-cr</b>	<b>Copies a file to a SE and registers the file in the catalog</b>
<b>lcg-del</b>	<b>Delete one file</b>
<b>lcg-rep</b>	<b>Replication between SEs and registration of the replica</b>
<b>lcg-gt</b>	<b>Gets the TURL for a given SURL and transfer protocol</b>
<b>lcg-sd</b>	<b>Sets file status to “Done” for a given SURL in a SRM request</b>

## File Catalog Interaction

<b>lcg-aa</b>	<b>Add an alias in LFC for a given GUID</b>
<b>lcg-ra</b>	<b>Remove an alias in LFC for a given GUID</b>
<b>lcg-rf</b>	<b>Registers in LFC a file placed in a SE</b>
<b>lcg-uf</b>	<b>Unregisters in LFC a file placed in a SE</b>
<b>lcg-la</b>	<b>Lists the alias for a given SURL, GUID or LFN</b>
<b>lcg-lg</b>	<b>Get the GUID for a given LFN or SURL</b>
<b>lcg-lr</b>	<b>Lists the replicas for a given GUID, SURL or LFN</b>

- Information on the file catalogs
  - LFC, gfal, lcg-utils:
    - “Evolution of LCG-2 Data Management (J-P Baud, J. Casey)”
      - <http://indico.cern.ch/contributionDisplay.py?contribId=278&sessionId=7&confId=0>
  - LFC installation, administration, migration from RLS:
    - Wiki entries indicated through the presentation:
      - [http://goc.grid.sinica.edu.tw/gocwiki/How\\_to\\_set\\_up\\_an\\_LFC\\_service](http://goc.grid.sinica.edu.tw/gocwiki/How_to_set_up_an_LFC_service)
      - [http://goc.grid.sinica.edu.tw/gocwiki/How\\_to\\_migrate\\_the\\_RLS\\_entries\\_into\\_the\\_LCG\\_File\\_Catalog\\_%28LFC%29](http://goc.grid.sinica.edu.tw/gocwiki/How_to_migrate_the_RLS_entries_into_the_LCG_File_Catalog_%28LFC%29)
  - LFC contacts:
    - [Jean-Philippe.Baud@cern.ch](mailto:Jean-Philippe.Baud@cern.ch)
    - [Sophie.Lemaitre@cern.ch](mailto:Sophie.Lemaitre@cern.ch)

# Practicals on Data Management (FiReMan)

> `glite-catalog-ls <file/directory to list>`

## Main options :

- l long output (with permissions)
- s URL, specify the service endpoint (i.e. the catalog to use)
- c Display the creation time instead of the modification time
- g Also display GUIDs

Check all the options with `-h`

ACL : - for regular files, **d** for directory, **l** for symbolic links and **v** for virtual directories. **p** indicates the permission to change attribute, while **d** rights to delete the entry. Successive 12 bits indicates, for user (u), group (g), other (o), permission to **r**ead, **w**rite, **l**ist contents or **e**xecute the content. Last two are reserved for metadata use, and so are currently unused. They will show the right to **g**et or **s**et the metadata.

> `glite-catalog-ls -l /test1103`

```
-pdrwl-gs--r-l-g-----      30   2005-07-18
11:01:55 /test1103
```

# Upload and register a file : glite-put

```
glite-put <localfilename> <remotefilename> [-m <mode>]
[-c <config>]
```

- glite-put copies a local file to a gLite I/O server, updating also the File Catalog.
- The file is copied in the IO server pointed by the UI's IO client
- The catalog updated is the one associated with that IO server

```
> glite-put file2register lfn:///test1156
[glite_put] Total 0.00 MB |=====| 100.00
% [0.0 Mb/s]
```

Transfer Completed:

```
LFN                : /test1156
GUID               : 0002bb56-7ce9-12db-bf0e-
c0a70228beef
SURL               : srm://glite-
se.ct.infn.it:8443/srm/managerv1?SFN=/pnfs/ct.infn.i
t/data/gilda/test1156
Data Written [bytes] : 30
Eff.Transfer Rate [Mb/s] : 0.000005
```

```
glite-catalog-stat [options] URI
```

**glite-catalog-stat** gives all the informations on a catalog entries, including file/directory permissions, GUID, owner/group, SURL location...

**It's very useful to verify correct setting of permissions and ownerships**

```
glite-get <remotefilename> <localfilename> [-c
<config>]
```

- glite-get download locally a file from the IO server pointed by the UI's client

```
glite-get lfn:///emacs localcopy
```

```
[glite_get] Total 0.00 MB          |=====|
 100.00 % [0.0 Mb/s]
```

Transfer Completed:

```
LFN                : /emacs
GUID                : 0032f276-8402-12db-9124-
c0a70219beef
SURL                :
srm://lxcde08.pd.infn.it:8443/srm/managerv1?SFN=/pnf
s/pd.infn.it/data/gilda/emacs
Data Written [bytes] : 237
Eff.Transfer Rate [Mb/s] : 0.000067
```



```
glite-rm <remotefilename> [-c <config>]
```

**glite-rm** removes a file from the IO server pointed by the UI, updating also the file catalog

```
glite-rm lfn:///emacs
```

```
Unlink Completed:
```

```
File           : lfn:///emacs
```

```
Time [s]       : 4.255000
```

- **Create a text file containing your name, surname and your birthplace**
- **Copy and register the file assigning as remote file name your username**
- **Verify correct file creation**
- **Download the file you've just created, changing its local file name**
- **Delete the file from the catalog**
- **Verify the correct file deletion**



# Explore the GILDA Testbed

- Once an user is logged into an User Interface and he/she has properly configured his/her grid environment (RB, BDII, File Catalog, MyProxy Server, ...), he/she is ready to take advantage of the Grid Power for his/her own application.
- But what are the resources available to accomplish his/her tasks?
- The answer to this question comes through the interactions with the **Information System (IS)**.
- The BDII (or IS) is a component that collects all the dynamic information coming from resources (CEs, SEs, ...) that can be accessed and employed by authorized users.

- Only the resources published on it really exist (it means available for users)
- Provides information to the Resource Broker during the matchmaking process\*
- Needed by the data management tools (LFC)
- It is accessed using **LDAP (Lightweight Directory Access Protocol)** queries
- All the stored information follows a **Glue Schema**.

Note(\*): A process on the RB queries BDII to retrieve all the resources matching certain job requirements.

`lcg-infosites --vo <your_vo> feature --is <your_bdii>`

- It's mandatory to include the **vo** and the **feature**
- The **--is** option specifies the **BDII** you want to query.

If not supplied, the **BDII** defined into the environment variable **LCG\_GFAL\_INFO\$Y\$** will be used.

*Features and descriptions:*

<b>closeSE</b>	Names of the CEs where the user's VO is allowed to run together with their corresponding closest SEs
<b>ce</b>	Number of CPUs, running and waiting jobs and names of the CEs
<b>se</b>	SEs names together with the available and used space
<b>lfc</b>	Name of the lfc Catalog for the user's VO
<b>all</b>	It groups all the features just described
<b>help</b>	Prints all the available options and exit

**To retrieve the status of all the available CEs serving the GILDA VO:**

**lcg-infosites --vo gilda ce --is grid004.ct.infn.it:2170**

\*\*\*\*\*

These are the related data for gilda: (in terms of queues and CPUs)

\*\*\*\*\*

#CPU	Free	Total Jobs	Running	Waiting	ComputingElement
36	36	0	0	0	grid010.ct.infn.it:2119/jobmanager-lcgpbs-long
0	0	0	0	0	ced-ce0.datagrid.cnr.it:2119/jobmanager-lcgpbs-long
36	36	0	0	0	grid010.ct.infn.it:2119/jobmanager-lcgpbs-short
0	0	0	0	0	ced-ce0.datagrid.cnr.it:2119/jobmanager-lcgpbs-short
20	20	0	0	0	gilda-ce-01.pd.infn.it:2119/jobmanager-lcgpbs-long
20	20	0	0	0	gilda-ce-01.pd.infn.it:2119/jobmanager-lcgpbs-short
36	36	0	0	0	grid010.ct.infn.it:2119/jobmanager-lcgpbs-infinite
0	0	0	0	0	ced-ce0.datagrid.cnr.it:2119/jobmanager-lcgpbs-infinite
6	6	0	0	0	grid-ce.bio.dist.unige.it:2119/jobmanager-lcgpbs-long
20	20	1	1	0	gilda-ce-01.pd.infn.it:2119/jobmanager-lcgpbs-infinite
6	6	0	0	0	grid-ce.bio.dist.unige.it:2119/jobmanager-lcgpbs-short
44	39	1	0	1	skurut1.cesnet.cz:2119/jobmanager-lcgpbs-gilda
6	6	0	0	0	grid-ce.bio.dist.unige.it:2119/jobmanager-lcgpbs-infinite

## Information on SEs available for the GILDA VO

```
lcg-infosites --vo gilda se --is grid004.ct.infn.it:2170
```

```
*****
These are the related data for gilda: (in terms of SE)
*****
```

Avail Space(Kb)	Used Space(Kb)	Type	SEs
208464576	78813032	disk	grid009.ct.infn.it
14762864	3278524	disk	ced-se0.datagrid.cnr.it
70479800	3666968	disk	gilda-se-01.pd.infn.it
388044168	419624	disk	grid-se.bio.dist.unige.it
14425144	3263156	disk	grid004.grid.elettra.trieste.it



**The names of the CE; where the user's VO is allowed to run together with their corresponding closest SE; are provided.**

• **lcg-infosites --vo gilda closeSE --is grid004.ct.infn.it:2170**

Name of the CE: `grid010.ct.infn.it:2119/jobmanager-lcgpbs-long`

Name of the close SE: `grid009.ct.infn.it`

Name of the CE: `ced-ce0.datagrid.cnr.it:2119/jobmanager-lcgpbs-long`

Name of the close SE: `ced-se0.datagrid.cnr.it`

Name of the CE: `grid010.ct.infn.it:2119/jobmanager-lcgpbs-short`

Name of the close SE: `grid009.ct.infn.it`

Name of the CE: `ced-ce0.datagrid.cnr.it:2119/jobmanager-lcgpbs-short`

Name of the close SE: `ced-se0.datagrid.cnr.it`

Name of the CE: `gilda-ce-01.pd.infn.it:2119/jobmanager-lcgpbs-long`

Name of the close SE: `gilda-se-01.pd.infn.it`

Name of the CE: `gilda-ce-01.pd.infn.it:2119/jobmanager-lcgpbs-short`

Name of the close SE: `gilda-se-01.pd.infn.it`

.....

Test some lcg-infosites features:

```
%lcg-infosites --vo gilda ce
```

```
%lcg-infosites --vo gilda se
```

```
%lcg-infosites --vo gilda lfc
```

```
%lcg-infosites --vo gilda all
```

- This program allows the user to query, in a deeper way, the BDII. It requires the environmental variable **LCG\_GFAL\_INFOSYS** to be set to the value of the BDII to be queried, e.g. **grid004.ct.infn.it:2170**
- Prints the list of the CEs or SEs satisfying a given **query** along with a list of specified attributes.
- The query syntax is like this:
  - attr1 op1 valueN, ... attrN opN valueN**
  - where *attrN* is an attribute name
  - op is =, >= or <=, and the cuts are ANDed.
  - A combination of attribute is supported.
  - The cuts are comma-separated and spaces are not allowed.

- list-attrs** Prints a list of attributes that can be queried.
- list-ce** Lists the CEs which satisfy a query, or all the CEs if no query is given.
- list-se** Lists the SEs which satisfy a query, or all the SEs if no query is given.
- query** Restricts the output to the CEs (SEs) which satisfy the given query.
- bdii** Allows to specify a BDII in the form <hostname>:<port>. If not given, the value of the environmental variable LCG\_GFAL\_INFOSYS is used. If that is not defined, the command returns an error.
- sed** Print the output in a "sed-friendly" format.
- attrs** Specifies the attributes whose values should be printed.
- vo** Restricts the output to CEs or SEs where the given VO is authorized. Mandatory when VO-dependent attributes are queried upon.

To get the list of all the “queryable” attributes:

## lcg-info --list-attrs

Attribute name	Glue object class	Glue attribute name
<b>MaxTime</b>	GlueCE	GlueCEPolicyMaxWallClockTime
<b>CEStatus</b>	GlueCE	GlueCEStateStatus
<b>TotalJobs</b>	GlueCE	GlueCEStateTotalJobs
<b>CEVOs</b>	GlueCE	GlueCEAccessControlBaseRule
<b>TotalCPUs</b>	GlueCE	GlueCEInfoTotalCPUs
<b>FreeCPUs</b>	GlueCE	GlueCEStateFreeCPUs
<b>CE</b>	GlueCE	GlueCEUniqueID
<b>WaitingJobs</b>	GlueCE	GlueCEStateWaitingJobs
<b>RunningJobs</b>	GlueCE	GlueCEStateRunningJobs
<b>CloseCE</b>	GlueCESEBindGroup	GlueCESEBindGroupCEUniqueID
<b>CloseSE</b>	GlueCESEBindGroup	GlueCESEBindGroupSEUniqueID
<b>SEVOs</b>	GlueSA	GlueSAAccessControlBaseRule
<b>UsedSpace</b>	GlueSA	GlueSAStateUsedSpace
<b>AvailableSpace</b>	GlueSA	GlueSAStateAvailableSpace
<b>Type</b>	GlueSE	GlueSEType
<b>SE</b>	GlueSE	GlueSEUniqueID
<b>Protocol</b>	GlueSEAccessProtocol	GlueSEAccessProtocolType
<b>ArchType</b>	GlueSL	GlueSLArchitectureType
<b>Processor</b>	GlueSubCluster	GlueHostProcessorModel
<b>OS</b>	GlueSubCluster	GlueHostOperatingSystemName
<b>Cluster</b>	GlueSubCluster	GlueSubClusterUniqueID
<b>Tag</b>	<b>GlueSubCluster</b>	<b>GlueHostApplicationSoftwareRunTimeEnvironment</b>
<b>Memory</b>	GlueSubCluster	GlueHostMainMemoryRAMSize
<b>OSVersion</b>	GlueSubCluster	GlueHostOperatingSystemRelease

To find out which kind of software is supported for your jobs by the CEs : `lcg-info --list-ce --attrs Tag`

CE: `grid010.ct.infn.it:2119/jobmanager-lcgpbs-infinite`

```
- Tag      LCG-2
           LCG-2_1_0
           LCG-2_1_1
           LCG-2_2_0
           LCG-2_3_0
           LCG-2_3_1
           LCG-2_4_0
           R-GMA
           AFS
           CMS-1.1.0
           ATLAS-6.0.4
           GATE-1.0.0-3
           LHCb-1.1.1
           IDL-5.4
           CMSIM-125
           ALICE-4.01.00
           ALIEN-1.32.14
           POVRAY-3.5
           DEMTOOLS-1.0
           CMKIN-VALID
           CMKIN-1.1.0
           CMSIM-VALID
           CSOUND-4.13
           MPICH
           VIRGO-1.0
           CMS-OSCAR-2.4.5
           LHCb_dbase_common-v3r1
           GEANT4-6
           VLC-0.7.2
           EGEODE-1.0
           RASTER3D
           SCILAB-2.6
           G95-3.5.0
           MAGIC-6.19
           CODESA3D-1.0
```

To get the list of all the CEs which satisfy the following attributes...

```
lcg-info --list-ce --attrs 'FreeCPUs,TotalJobs,CloseSE'
```

- CE: grid010.ct.infn.it:2119/jobmanager-lcgpbs-long

```
- FreeCPUs      36
- CloseSE       grid009.ct.infn.it
- TotalJobs     0
```

- CE: ced-ce0.datagrid.cnr.it:2119/jobmanager-lcgpbs-long

```
- FreeCPUs      6
- CloseSE       ced-se0.datagrid.cnr.it
- TotalJobs     0
```

- CE: grid010.ct.infn.it:2119/jobmanager-lcgpbs-short

```
- FreeCPUs      36
- CloseSE       grid009.ct.infn.it
- TotalJobs     0
```

- CE: ced-ce0.datagrid.cnr.it:2119/jobmanager-lcgpbs-short

```
- FreeCPUs      6
- CloseSE       ced-se0.datagrid.cnr.it
- TotalJobs     0
```

To get the list of all the CEs which satisfy the following query

```
lcg-info --list-ce --attrs FreeCPUs --query 'FreeCPUs >= 30'
```

- CE: grid010.ct.infn.it:2119/jobmanager-lcgpbs-long
  - FreeCPUs 36
- CE: grid010.ct.infn.it:2119/jobmanager-lcgpbs-short
  - FreeCPUs 36
- CE: grid010.ct.infn.it:2119/jobmanager-lcgpbs-infinite
  - FreeCPUs 36
- CE: skurut1.cesnet.cz:2119/jobmanager-lcgpbs-long
  - FreeCPUs 40
- CE: skurut1.cesnet.cz:2119/jobmanager-lcgpbs-gilda
  - FreeCPUs 40
- CE: skurut1.cesnet.cz:2119/jobmanager-lcgpbs-short
  - FreeCPUs 40
- CE: skurut1.cesnet.cz:2119/jobmanager-lcgpbs-infinite
  - FreeCPUs 40



# R-GMA Practical

- To Start the R-GMA command line tool run the following command:

**rgma**

- On startup you should receive the following message:

```
Welcome to the R-GMA virtual database for Virtual Organisations.  
You are connected to the R-GMA registry service at
```

```
http://<registry-host>:8080/R-GMA/RegistryServlet
```

```
Type "help" for a list of commands.
```

```
rgma>
```

- **Commands** are entered by typing at the **rgma>** prompt and hitting 'enter' to execute the command.
- A **history** of the commands executed can be accessed using the Up and Down arrow keys.
- To **search a command from history** use CTRL-R and type the first few letters of the command to recall.
- **Command autocompletion** is supported (use Tab when you have partly entered a command).

## General Commands

- **help**

Display general help information.

- **help <command>**

Display help for a specific command.

- **help examples**

Display list of example commands.

- **exit or quit**

Exit from R-GMA command line interface.

- **Show tables**

Display the name of all tables existing in the Schema

- **Describe <tablename>**

Show all information about the structure of a table

- Querying data uses the standard SQL SELECT statement, e.g.:

```
rgma> SELECT * FROM ServiceStatus
```

The behaviour of SELECT varies according to the type of query being executed. In R-GMA there are three basic types of query:

- **LATEST Queries** only the most recent tuple for each primary key
- **HISTORY Queries** all historical tuples for each primary key
- **CONTINUOUS Queries** returns tuples continuously as they are inserted.

- The type of query can be changed using the SET QUERY command as follow:

```
rgma> SET QUERY LATEST
```

or

```
rgma> SET QUERY CONTINUOUS
```

- The current query type can be displayed using  

```
rgma> SHOW QUERY
```

## 1. Display all the table of the Schema

```
rgma>show tables
```

## 2. Display information about Site table

```
rgma>describe Site
```

## 3. Basic select query on the table named Site

```
rgma>set query latest
```

```
rgma>show query
```

```
rgma>select Name,Latitude,Longitude from Site
```

- The maximum age of tuples to return can also be controlled. To limit the age of latest or historical tuples use the **SET MAXAGE** command. The following are equivalent:

```
rgma> SET MAXAGE 2 minutes
```

```
rgma> SET MAXAGE 120
```

- The current maximum tuple age can be displayed using **rgma> SHOW MAXAGE**
- To disable the maximum age, set it to none:  

```
rgma> SET MAXAGE none
```



- **The final property affecting queries is timeout.**
  - For a latest or history query the timeout exists to prevent a problem (e.g. network failure) from stopping the query from completing.
  - For a continuous query, timeout indicates how long the query will continue to return new tuples. Default timeout is 1 minute and it can be changed using

**rgma>SET TIMEOUT 3 minutes or SET TIMEOUT 180**

- **The current timeout can be displayed using**  
**rgma>SHOW TIMEOUT**

- The SQL INSERT statement may be used to add data to the system:

```
rgma> INSERT INTO Table VALUES ('a', 'b', 'c', 'd')
```

- In R-GMA, data is inserted into the system using a **Producer** component which handles the INSERT statement.
- Using the command line tool you may work with one producer at a time.
- The current producer type can be displayed using:  

```
rgma>show producer
```
- The producer type can be set using:  

```
rgma>set producer latest
```

## 1. Insert and Select using Primary Producer to support Continuous + History query

```
rgma>set producer continuous
```

```
rgma>insert into UserTable values('cod','string',1.4,66)
```

```
rgma>set query continuous
```

```
rgma>set maxage 1 minutes
```

```
rgma>set timeout 5 seconds
```

```
rgma>select * from UserTable
```

- To instruct the secondary producer to consume from table MyTable:

```
rgma> SECONDARYPRODUCER MyTable
```

- Like the producer, the secondary producer may be configured to answer latest and/or history queries:

```
rgma> SET SECONDARYPRODUCER latest
```

- (By default the secondary producer can answer both latest and history queries. )
- The current secondary producer type can be displayed using:

```
rgma> SHOW SECONDARYPRODUCER
```

## 2. Insert and Select using the Secondary Producer to support the latest query.

```
rgma>set secondaryproducer latest  
rgma>secondaryproducer UserTable  
rgma>show producers of UserTable  
rgma>set producer continuous  
rgma>insert into UserTable values ('cod','string',5.2,44)  
rgma>set query latest  
rgma>select * from UserTable
```

THE END