GILDA Practicals

Giuseppe Andronico
Roberto Barbera
Mike Mineter

EGEE Tutorial, SEOUL, 29-30.08.2005

Outline

- 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

User Interface Access

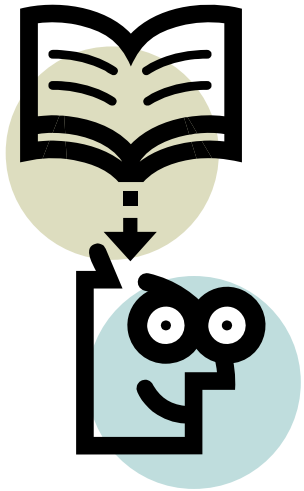
Host: glite-tutor.ct.infn.it

Username: seoulXX (XX=01…60)

Password: GridSEOXX (XX=01…60)

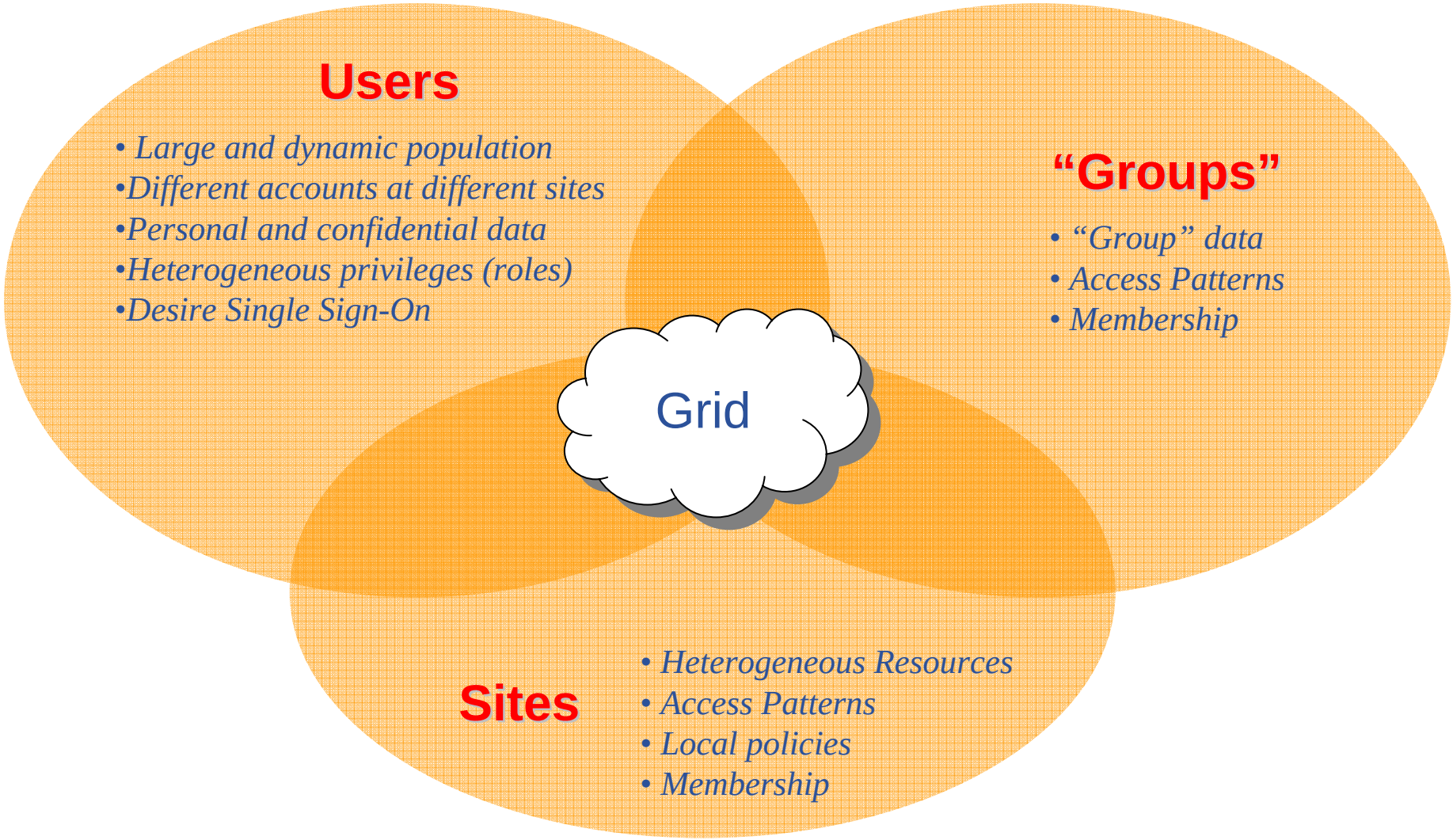
PassPhrase: SEOUL

GENIUS: https://glite-tutor.ct.infn.it



Proxy/MyProxy Management

GRID Security: the players



Digital certificates

The goal of authorization and autentication 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 autenticity with a certificate

Certificate Use

- Digital certificates are split in public/private keys
- Public key is spread along the net, while the private stays encripted 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
```

Verify your certificate

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=SEOUL, CN=SEOUL20/Email=roberto.barbera@ct.infn.it

......
```

X.509 proxy certificates

- **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:

SEOUL

```
> grid-proxy-init
Your identity: /C=IT/O=GILDA/OU=Personal
   Certificate/L=SEOUL/CN=SEOUL20/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
```

Inspecting your proxy

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

```
>grid-proxy-info -all
subject  : /C=IT/O=GILDA/OU=Personal
  Certificate/L=SEOUL/CN=SEOUL20/Email=roberto.barber
  a@ct.infn.it/CN=proxy
issuer   : /C=IT/O=GILDA/OU=Personal
  Certificate/L=SEOUL/CN=SEOUL20/Email=roberto.barber
  a@ct.infn.it
identity : /C=IT/O=GILDA/OU=Personal
  Certificate/L=SEOUL/CN=SEOUL20/Email=roberto.barber
  a@ct.infn.it
type     : full legacy globus proxy
strength : 512 bits
path     : /tmp/x509up_u500
timeleft : 11:57:24
```

Long term proxy

- **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

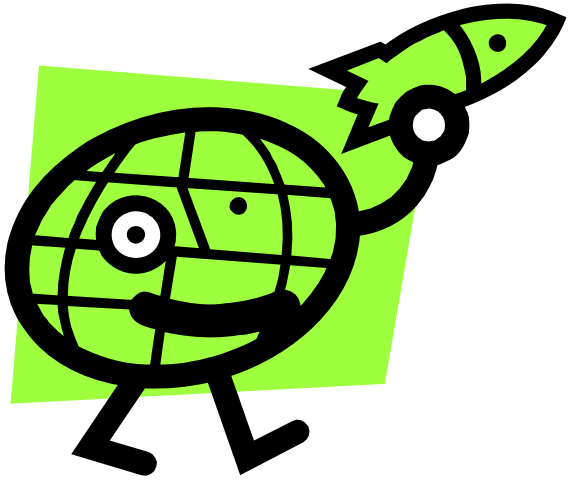
```
> grid-proxy-destroy remove local credentials
> myproxy-init -s grid001.ct.infn.it -l <UniqueUsername>
Your identity: /C=IT/O=GILDA/OU=Personal Certificate/L
=SEOUL/CN=<UniqueUsername>/Email=
roberto.barbera@ct.infn.it
Enter GRID pass phrase for this identity:
Creating proxy .....................Done
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

Get delegation

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

Workload Managements 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

JDL

- **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**

Job Preparation

- **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

Job Description Language

- **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*

JDL – Relevant Attributes

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

Job Submission

- **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 jobId on a file, instead of STDIN

--resource, -r <resource value>, specify the resource for execution (needs the GLUE UniqueId of the queue, obtainable with **list-match**)

--debug show function calls and parameters

Job life cycle check

- **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
```

JDL -- Example

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

Exercises

>cd UIPnPcomb/example_JDL

Create a bash script which displays hostname and current date

Save the script as yourscript.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 (yourscript.sh).

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

Job Requirements

- 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)

JDL – Requirements

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

Job Submission

- **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

Exercises

- **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

JDL -- Requirements

```
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);
```

Start_povray_cubo.sh

```
#!/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
```

.BrokerInfo file

- 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 transfered 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 getSEMountPoint <SE>

edg-brokerinfo getSEFreeSpace <SE>

edg-brokerinfo getSEProtocols <SE>

edg-brokerinfo getSEPort <SE> <Protocol>

edg-brokerinfo getVirtualOrganization

edg-brokerinfo getAccessCost

Exercises

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
```

Exercises

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;
 ]
```

Exercises

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.

Explore the GILDA Testbed

How to discover resources

- 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.

The BDII

- 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

```
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_INFOSYS** will be used.

Features and descriptions:

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

lcg-infosites

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
```

lcg-infosites (II)

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
```

lcg-infosites (III)

The names of the CEs where the user's VO is allowed to run together with their corresponding closest SEs 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
......
```

lcg-infosites test

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
```

lcg-info intro

- **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.

lcg-info options

--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.

lcg-info examples (I)

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

lcg-info --list-attrs

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

lcg-info examples (II)

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

lcg-info examples (III)

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

lcg-info examples (IV)

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

R-GMA Command Line Tool (1)

- 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>
```

Entering Command

- 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

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 (1)

- 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.

Querying Data (2)

- 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
```

Exercises

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

Maximum AGE of tuples

- 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

Query Timeout

- 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

Producer & Inserting Data

- **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

Exercises

1. **Insert and Select using Primary Producer to support Continuos + 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
```

Secondary Producer

- 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
```

Exercises

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
```

Practicals on Data Managemnt (LFC and lcg-utils)

Set up your environment

- **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 SEOUL

LFC Catalog commands

Listing the entries of a LFC directory

lfc-ls [-cdiLlRTu] [--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 examples

```
$ 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 uploadfilelfn
-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
```

LFC Catalog commands

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/seoul/linkToUser.ex

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

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

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

LFC Catalog commands

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 lcg-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

LFC Catalog commands

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 Seoul”

- Check your job with..

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

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

LFC Catalog commands

- 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
```

Hands-on Session

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

LFC Catalog commands

Summary of the LFC Catalog commands

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

lcg-utils

- 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**

lcg-utils: lcg-cr

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:///*

lcg-utils: lcg-cr

- **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/seoul/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/SEOUL/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
```

lcg-utils: lcg-aa and lcg-la

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/SEOUL/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/seoul/aliasToNote.txt
lfn:/grid/gilda/seoul/note.txt
lfn:/grid/gilda/seoul/aliasToNote.txt
```

Hands-on session

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

lcg-utils commands for replicas (I)

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/seoul/note.txt
Using grid catalog type: lfc
Source URL: lfn:/grid/gilda/seoul/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
```

lcg-utils commands for replicas (II)

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/SEOUL/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

lcg-utils commands for replicas (III)

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/seoul/note.txt
```

- delete all the replicas

```
$ lcg-del -a --vo gilda lfn:/grid/gilda/seoul/note.txt
```

- let's check if the previous command was successful

```
$ lcg-lr --vo gilda lfn:/grid/gilda/seoul/note.txt
```

lcg_lr: No such file or directory

- or by `lfs-ls /grid/gilda/seoul` (you will not see anymore note.txt and its alias)

lcg-utils: lcg-cp

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/seoul/note2.txt
  file:/home/local/note2.txt
Source URL: lfn:/grid/gilda/SEOUL/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
```

Hands-on session

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

Final exercise

- **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/seoul/<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/SEOUL/<lfn you choose> file:`pwd`/output.dat
echo "Done.."
```

Final exercise (II)

- **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**

Summary of lcg-utils commands

Replica Management

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

File Catalog Interaction

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

Bibliography

- 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_migrate_the_RLS_entries_into_the_LCG_File_Catalog_%28LFC%29*
  - LFC contacts:
    - Jean-Philippe.Baud@cern.ch
    - Sophie.Lemaitre@cern.ch

Practicals on Data Managemnt (FiReMan)

Browsing the catalog

```
> 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 read, write, list contents or execute the content. Last two are reserved for metadata use, and so are currently unused. They will show the right to get or set 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
```

Check FC entries : glite-catalog-stat

```
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

Download a file : glite-get

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
```

Remove a file

```
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
```

Exercise

- 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

THE END