



Enabling Grids for
E-science in Europe

www.eu-egee.org

Grid Data Management

Simone Campana
LCG Experiment Integration and Support
CERN IT



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

Overview

- **Introduction on Data Management (DM)**
 - **General Concepts**
 - **Some details on transport protocols**
 - **Data management operations**
 - **Files & replicas: Name Convention**
- **File catalogs**
 - **Cataloging requirements and catalogs in egee**
 - **RLS file catalog**
 - **LCG file catalog**
- **DM tools: overview**
- **Data Management CLI**
 - **lcg_utils**
- **Data Management API**
 - **lcg_utils**
- **Advanced concepts**
 - **Advanced utilities: CLI&APIs**
 - **OutputData JDL attribute**
- **Conclusions**



Overview

- **Introduction on Data Management (DM)**
 - **General Concepts**
 - **Some details on transport protocols**
 - **Data management operations**
 - **Files & replicas: Name Convention**
- **File catalogs**
 - **Cataloging requirements and catalogs in egee**
 - **RLS file catalog**
 - **LCG file catalog**
- **DM tools: overview**
- **Data Management CLI**
 - **lcg_utils**
- **Data Management API**
 - **lcg_utils**
- **Advanced concepts**
 - **Advanced utilities: CLI&APIs**
 - **OutputData JDL attribute**
- **Conclusions**



Data Management: general concepts

- What does “Data Management” mean?
 - Users and applications produce and require data
 - Data may be stored in Grid files
 - Granularity is at the “file” level (no data “structures”)
 - Users and applications need to handle files on the Grid
- Files are stored in appropriate permanent resources called “Storage Elements” (SE)
 - Present almost at every site together with computing resources
 - Described in details in next presentations
 - We will treat a storage element as a “black box” where we can store data
 - Appropriate data management utilities/services hide internal structure of SE
 - Appropriate data management utilities/services hide details on transfer protocols

Data Management: general concepts

- A Grid file is READ-ONLY (at least in egee)
 - It can not be modified
 - It can be deleted (so it can be replaced)
 - Files are heterogeneous (ascii, binary ...)
- Data Management does not include file ACCESS
 - File access will be covered in the Storage section
- High level Data Management tools (lcg_utils, see later) hide
 - transport layer details (protocols ...)
 - Storage location
- To use lower level tools (edg-gridftp, see later) you need
 - some knowledge of the transport layer
 - some knowledge of Storage Element implementation

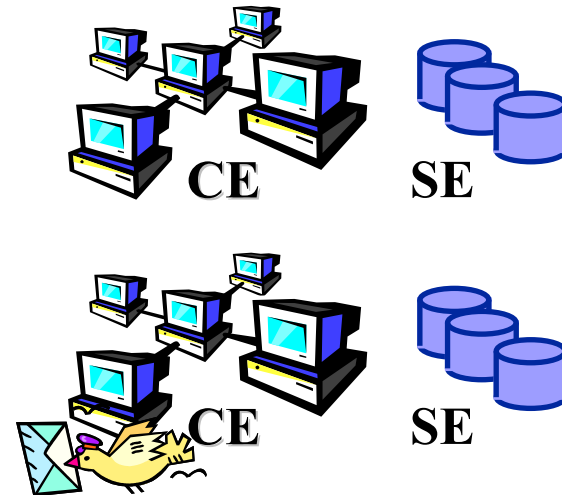
Some details on protocols

- Data channel protocol: mostly gridFTP (**gsiftp**)
 - secure and efficient data movement
 - extends the standard FTP protocol
 - Public-key-based **Grid Security Infrastructure** (GSI) support
 - Third-party control of data transfer
 - **Parallel** data transfer
- Other protocols are available, especially for File I/O
 - **rfio protocol**:
 - for CASTOR SE (and classic SE)
 - Not yet GSI enabled
 - **gsidcap protocol**:
 - for secure access to dCache SE
 - **file protocol**:
 - for local file access
- Other Control Channel Protocols (SRM, discussed in SE lecture ...)

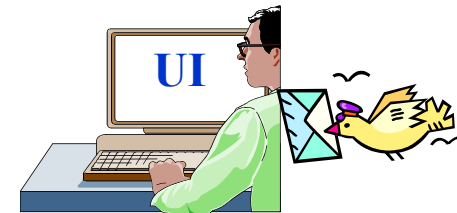
Data Management operations

Upload a file to the grid

- User need to store data in SE (from a UI)
- Application need to store data in SE (from a WN)
- User need to store the application (to be retrieved and run from WN)
 - For small files the InputSandbox can be used (see WMS lecture)



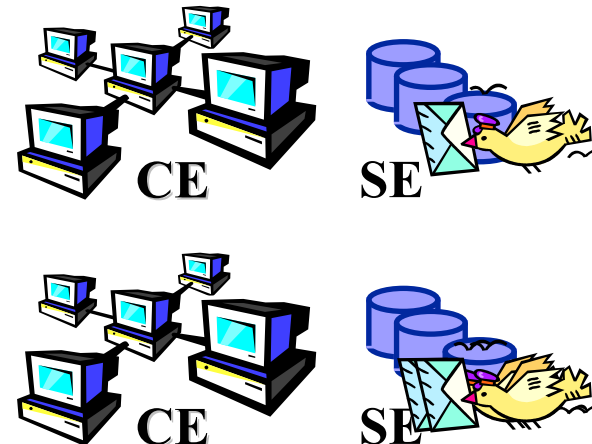
Several Grid Components



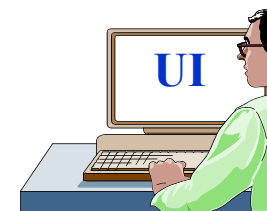
Data Management operations

Download files from the grid

- User needs to retrieve (onto the UI) data stored into SE
 - For small files produced in WN the OutputSandbox can be used (see WMS lecture)
- Application needs to copy data locally (into the WN) and use them
- The application itself must be downloaded onto the WN and run



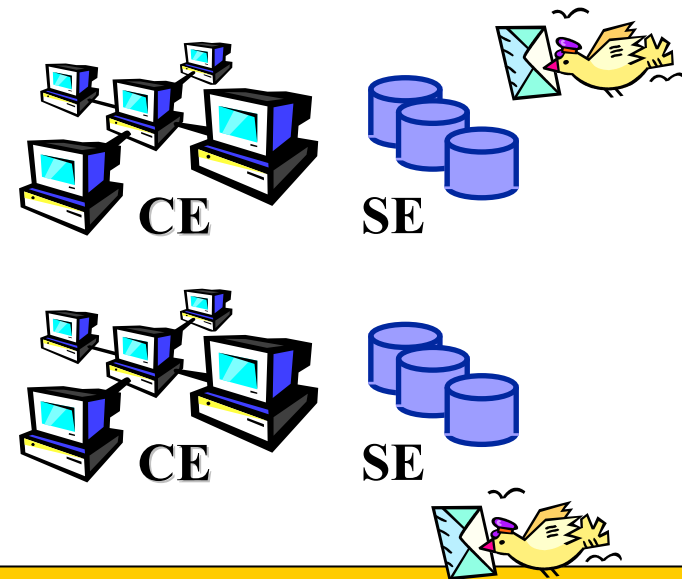
Several Grid Components



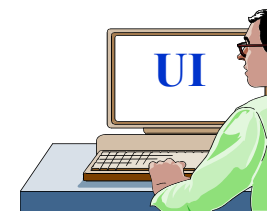
Data Management operations

Replicate a file across different SEs

- Load share balancing of computing resources
 - Often a job needs to run at a site where a copy of input data is present
 - See InputData JDL attribute in WMS lecture
- Performance improvement in data access
 - Several applications might need to access the same file concurrently
- Important for redundancy of key files (backup)



Several Grid Components



Data management operations

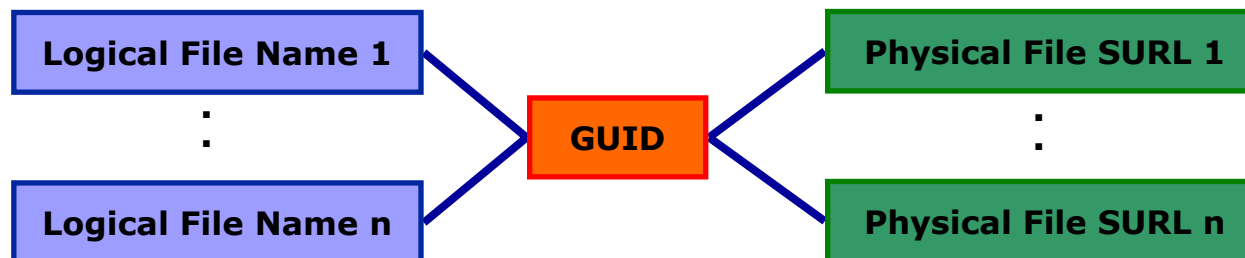
- Data Management means movement and replication of files across/on grid elements
- Grid DM tools/applications/services can be used for all kind of files

HOWEVER

- Data Management focuses on “large” files
 - large means greater than ~20MB
 - Typically on the order of few hundreds MB
- Tools/applications/services are optimized to deal with large files
- In many cases, small files can be efficiently treated using different procedures
 - Examples:
 - User can ship data to be used by the application on the WN (and possibly the application itself) using the InputSandbox (see WMS lecture)
 - User can retrieve (on the UI) data generated by a job (on the WN) using the OutputSandbox (see WMS lecture)

Files & replicas: Name Convention

- Globally Unique Identifier (**GUID**)
 - A non-human-readable unique identifier for a file, e.g.
“guid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6”
- Site URL (**SURL**) (or Physical/Site File Name (**PFN/SFN**))
 - The location of the actual file on a storage system, e.g.
“sfn://lxshare0209.cern.ch/data/alice/ntuples.dat”
- Logical File Name (**LFN**)
 - An alias created by a user to refer to some file, e.g.
“lfn:cms/20030203/run2/track1”
- Transport URL (**TURL**)
 - Temporary locator of a replica + access protocol: understood by a SE, e.g.
“gsiftp://lxshare0209.cern.ch//data/alice/ntuples.dat”



Overview

- **Introduction on Data Management (DM)**
 - **General Concepts**
 - **Some details on transport protocols**
 - **Data management operations**
 - **Files & replicas: Name Convention**
- **File catalogs**
 - **Cataloging requirements and catalogs in egee**
 - **RLS file catalog**
 - **LCG file catalog**
- **DM tools: overview**
- **Data Management CLI**
 - **lcg_utils**
- **Data Management API**
 - **lcg_utils**
- **Advanced concepts**
 - **Advanced utilities: CLI&APIs**
 - **OutputData JDL attribute**
- **Conclusions**



At this point you should ask:

- 1) How do I keep track of all my files on the Grid?
- 2) Even if I remember all the Ifns of my files, what about someone else files?
- 3) Anyway, how does the Grid keep track of associations Ifn/GUID/surl?

Well... we need a FILE CATALOGUE

Cataloging Requirements

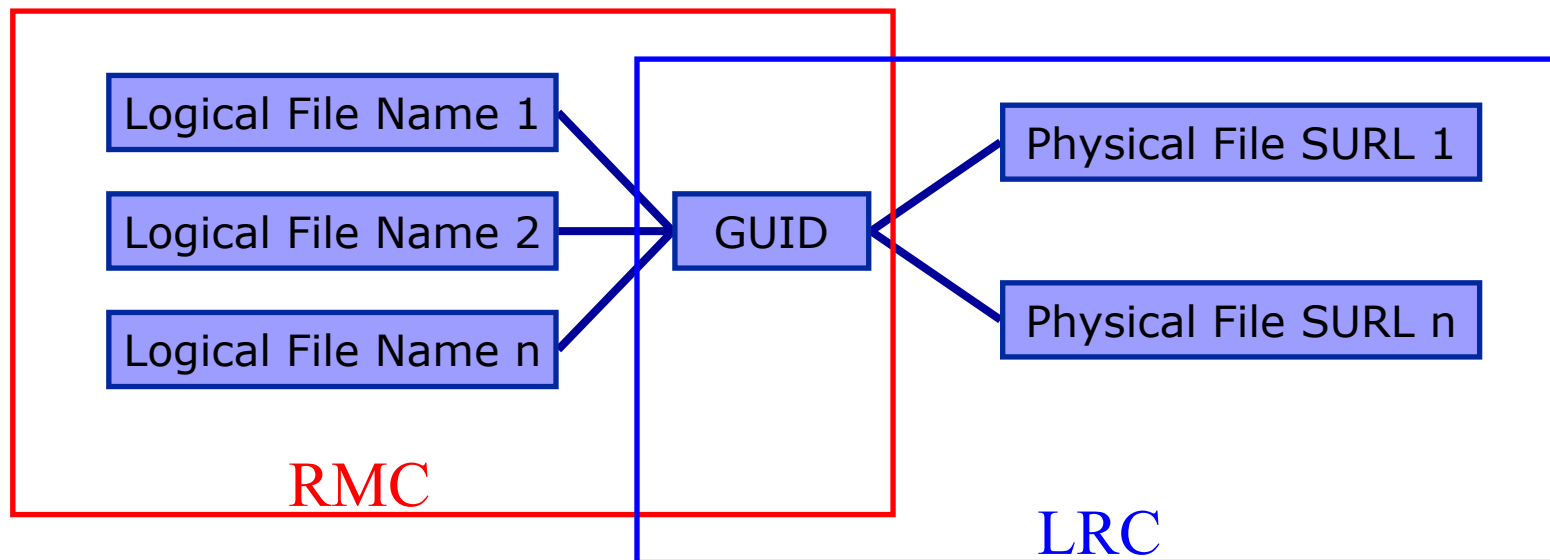
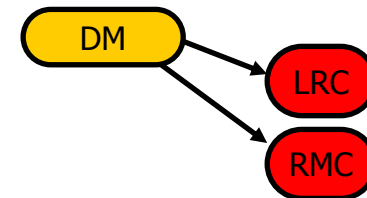
- Need to keep track of the location of copies (replicas) of Grid files
- Replicas might be described by **attributes**
 - Support for **METADATA**
 - Could be “system” metadata or “user” metadata
- Potentially, millions of files need to be registered and located
 - Requirement for **performance**
- Distributed architecture might be desirable
 - **scalability**
 - prevent single-point of failure
 - Site managers need to change autonomously file locations

File Catalogs in egee

- Access to the file catalog
 - The DM tools and APIs and the WMS interact with the catalog
 - Hide catalogue implementation details
 - Lower level tools allow direct catalogue access
- EDG's Replica Location Service (**RLS**)
 - Catalogs in use in LCG-2
 - Replica Metadata Catalog (**RMC**) + Local Replica Catalog (**LRC**)
 - Some performance problems detected during LCG Data Challenges
- New LCG File Catalog (**LCF**)
 - Already being certified; deployment in January 2005
 - Coexistence with RLS and migration tools provided
 - Better performance and scalability
 - Provides new features: security, hierarchical namespace, transactions...

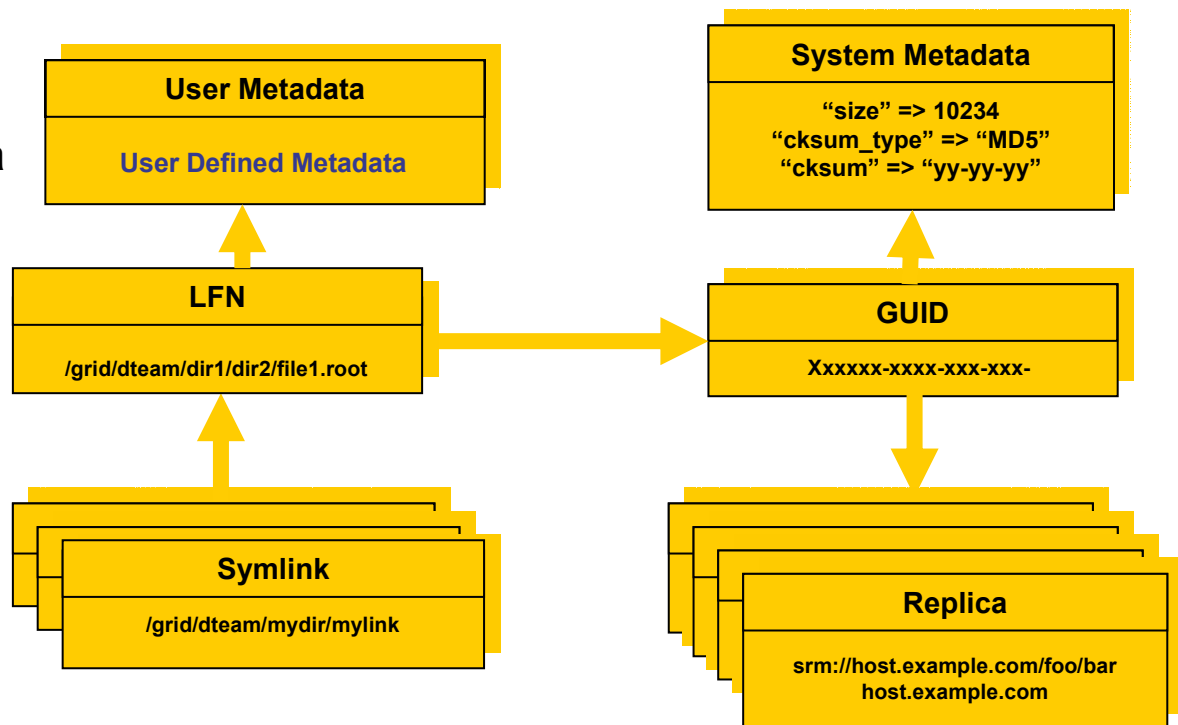
File Catalogs: The RLS

- RMC:
 - Stores LFN-GUID mappings
 - Accessible by edg-rmc CLI + API
- LRC:
 - Stores GUID-SURL mappings
 - Accessible by edg-lrc CLI + API



File Catalogs: The LFC

- One single catalog
- LFN acts as main key in the database. It has:
 - Symbolic links to it (additional LFNs)
 - Unique Identifier (GUID)
 - System metadata
 - Information on replicas
 - One field of user metadata



File Catalogs: The LFC (II)

- Fixes performance and scalability problems seen in EDG Catalogs
 - Cursors for large queries
 - Timeouts and retries from the client
- Provides more features than the EDG Catalogs
 - User exposed transaction API (+ auto rollback on failure of mutating method call)
 - Hierarchical namespace and namespace operations (for LFNs)
 - Integrated GSI Authentication + Authorization
 - Access Control Lists (Unix Permissions and POSIX ACLs)
 - Checksums
- Interaction with other components
 - Supports Oracle and MySQL database backends
 - Integration with GFAL and lcg_util APIs complete
 - New specific API provided
- New features will be added (requests welcome!)
 - ROOT Integration in progress
 - POOL Integration will be provided soon
 - VOMS will be integrated

Overview

- **Introduction on Data Management (DM)**
 - General Concepts
 - Some details on transport protocols
 - Data management operations
 - Files & replicas: Name Convention
- **File catalogs**
 - Cataloging requirements and catalogs in egee
 - RLS file catalog
 - LCG file catalog
- **DM tools: overview**
- **Data Management CLI**
 - lcg_utils
- **Data Management API**
 - lcg_utils
- **Advanced concepts**
 - Advanced utilities: CLI&APIs
 - OutputData JDL attribute
- **Conclusions**



Data Management tools

- Data Management tool (set of services)
 - to queue transfer
 - possible
 -
 - balance
 -
 - transfer (limited functionality in respect of the one described so far)
 -
 - validate transfer as processes
 - i.e. checking the
- Data Management tool
 - acts as the single interface to all data management services

Data management tools

- **Replica manager**: lcg-* commands + lcg_* API
 - Provide (all) the functionality needed by the egee user
 - Combine file transfer and cataloging as an **atomic transaction**
 - Insure consistent operations on catalogues and storage systems
 - Offers high level layer over technology specific implementations
 - Based on the Grid File Access Library (**GFAL**) API
 - Discussed in SE section
- **edg-gridftp** tools: CLI
 - Complete the lcg_utils with GridFTP operations
 - Lower level layer w.r.t. Replica Manager
 - Only for gridFTP protocol
 - Functionality available in GFAL
 - May be implemented as lcg-* commands

DM CLIs & APIs: Old EDG tools

- Old versions of EDG CLIs and APIs still available
- File & replica management
 - `edg-rm`
 - Implemented (mostly) in java
- Catalog interaction (only for EDG catalogs)
 - `edg-lrc`
 - `edg-rmc`
 - Java and C++ APIs
- Use discouraged
 - Worse performance (slower)
 - New features added only to `lcg_utils`
 - Less general than GFAL and `lcg_utils`

Overview

- **Introduction on Data Management (DM)**
 - **General Concepts**
 - **Some details on transport protocols**
 - **Data management operations**
 - **Files & replicas: Name Convention**
- **File catalogs**
 - **Cataloging requirements and catalogs in egee**
 - **RLS file catalog**
 - **LCG file catalog**
- **DM tools: overview**
- **Data Management CLI**
 - **lcg_utils**
- **Data Management API**
 - **lcg_utils**
- **Advanced concepts**
 - **Advanced utilities: CLI&APIs**
 - **OutputData JDL attribute**
- **Conclusions**



Gathering informations: *lcg-infosites*

- Not really a Data Management tool
 - Wrapper around Information System Client
- Very useful to discover resources
 - Storage Elements
 - Catalog end points
 - (...)
- Usage: `lcg-infosites --vo voname option [--is BDII] [--help]`
 - Possible options: `se`, `ce`, `closeSE`, `lrc`, `rmc`, `all`
 - `--vo` field is mandatory
 - `--is` : allows to specify the BDII to query
 - If flag not used, the BDII defined into `LCG_GFAL_INFOSYS` environmental variable is used
 - Try the `--help` flag for a list of possible options

lcg_utils: Replica mgm. commands

lcg-cp	Copies a Grid file to a local destination
lcg-cr	Copies a file to a SE and registers the file in the LRC
lcg-del	Deletes one file (either one replica or all replicas)
lcg-rep	Copies a file from SE to SE and registers it in the LRC
lcg-se	set file status to “Done” in a specified request

lcg_utils: Catalog interaction cmd's

lcg-aa	Adds an alias in RMC for a given GUID
lcg-gt	Gets the TURL for a given SURL and transfer protocol
lcg-la	Lists the aliases for a given LFN, GUID or SURL
lcg-lg	Gets the GUID for a given LFN or SURL
lcg-lr	Lists the replicas for a given LFN, GUID or SURL
lcg-ra	Removes an alias in RMC for a given GUID
lcg-rf	Registers a SE file in the LRC (optionally in the RMC)
lcg-uf	Unregisters a file residing on an SE from the LRC

Gathering informations: *lcg-infosites*

```
[scampana@grid019:~]$ lcg-infosites --vo gilda se
```

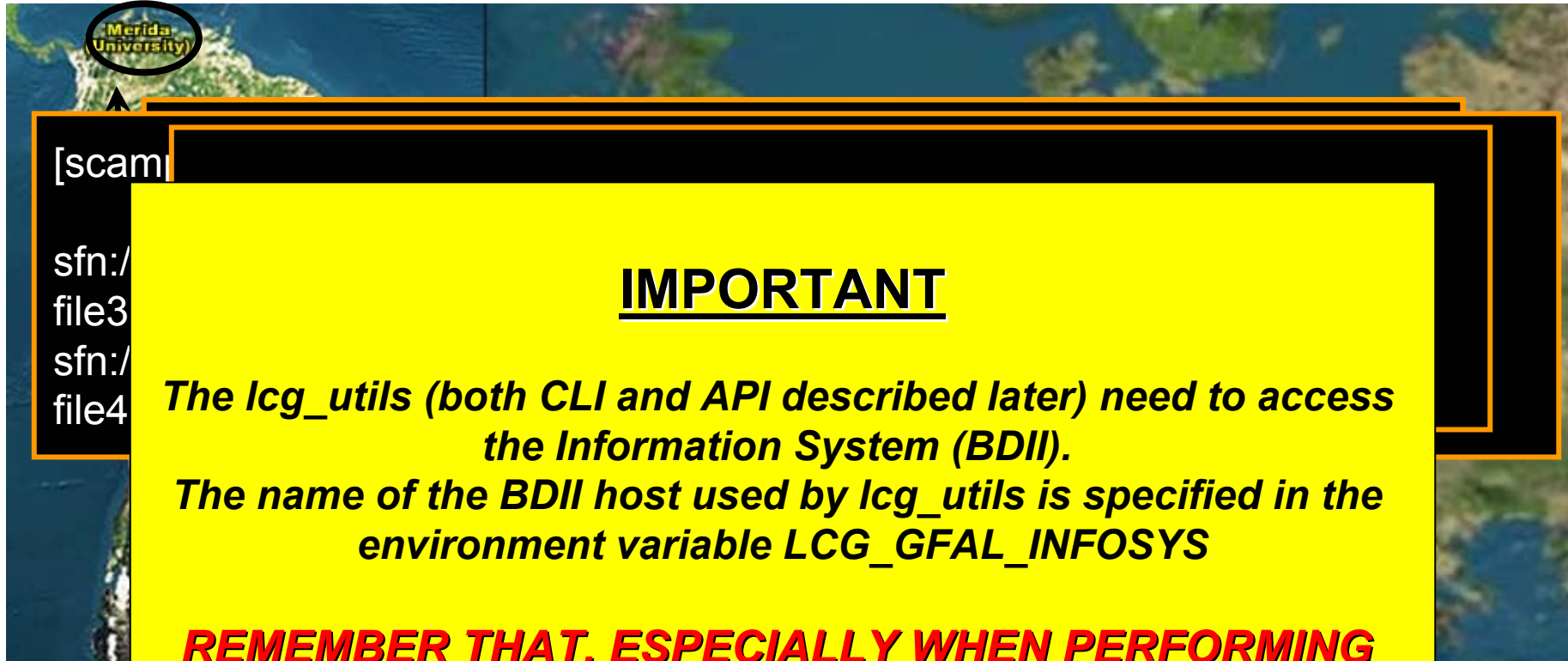
```
*****
```

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

```
*****
```

Avail Space (Kb)	Used Space (Kb)	SEs
1570665704	576686868	grid3.na.astro.it
225661244	1906716	grid009.ct.infn.it
523094840	457000	grid003.cecalc.ula.ve
1570665704	576686868	testbed005.cnaf.infn.it
15853516	1879992	gilda-se01.pd.infn.it

lcg_utils CLI : usage example



[scam]

sfn:/
file3
sfn:/
file4

IMPORTANT

*The lcg_utils (both CLI and API described later) need to access the Information System (BDII).
The name of the BDII host used by lcg_utils is specified in the environment variable LCG_GFAL_INFOSYS*

REMEMBER THAT, ESPECIALLY WHEN PERFORMING DATA MANAGEMENT OPERATIONS FROM THE WN

Welcome to the first Latin American Grid Workshop in Catania
The data is now being transferred to Merida now!!!

Overview

- **Introduction on Data Management (DM)**
 - **General Concepts**
 - **Some details on transport protocols**
 - **Data management operations**
 - **Files & replicas: Name Convention**
- **File catalogs**
 - **Cataloging requirements and catalogs in egee**
 - **RLS file catalog**
 - **LCG file catalog**
- **DM tools: overview**
- **Data Management CLI**
 - **lcg_utils**
- **Data Management API**
 - **lcg_utils**
- **Advanced concepts**
 - **Advanced utilities: CLI&APIs**
 - **OutputData JDL attribute**
- **Conclusions**



lcg_utils API

- lcg_utils API:
 - High-level data management C API
 - Same functionality as lcg_util command line tools
- Single shared library
 - `liblcg_util.so`
- Single header file
 - `lcg_util.h`
(+ linking against `libglobus_gass_copy_gcc32.so`)

lcg_utils: Replica management

```
int lcg_cp (char *src_file, char *dest_file, char *vo, int nbstreams, char  
* conf_file, int insecure, int insecure);
```

```
int lcg_cr (char *src_file, char *dest_file, char *guid, char *lfn, char  
*vo, char *relative_path, int nbstreams, char *conf_file, int insecure,  
int verbose, char *actual_guid);
```

```
int lcg_del (char *file, int aflag, char *se, char *vo, char *conf_file, int  
insecure, int verbose);
```

```
int lcg_rep (char *src_file, char *dest_file, char *vo, char  
*relative_path, int nbstreams, char *conf_file, int insecure, int  
verbose);
```

```
int lcg_sd (char *surl, int regid, int fileid, char *token, int oflag);
```

lcg_utils: Catalog interaction

```
int lcg_aa (char *lfn, char *guid, char *vo, char *insecure, int verbose);
```

```
int lcg_gt (char *surl, char *protocol, char **turl, int *regid, int *fileid,  
char **token);
```

```
int lcg_la (char *file, char *vo, char *conf_file, int insecure, char ***lfns);
```

```
int lcg_lg (char *lfn_or_surl, char *vo, char *conf_file, int insecure, char  
*guid);
```

```
int lcg_lr (char *file, char *vo, char *conf_file, int insecure, char ***pfns);
```

```
int lcg_ra (char *lfn, char *guid, char *vo, char *conf_file, int insecure);
```

```
int lcg_rf (char *surl, char *guid, char *lfn, char *vo, char *conf_file, int  
insecure, int verbose, char *actual_guid);
```

```
int lcg_uf (char *surl, char *guid, char *vo, char *conf_file, int insecure);
```


Available APIs

```
#include <iostream>
#include <stdlib.h>
#include <string.h>
#include <string>
#include <stdio.h>
#include <errno.h>

// lcg_util is a C library. Since we write C++ code here, we need to
// use extern C
//
extern "C"
{
#include <lcg_util.h>
}
using namespace std;
/*****/
/* The following example code shows you how you can use the lcg_util API for */
/* replica management. We expect that you modify parts of this code in */
/* to make it work in your environment. This is particularly indicated */
/* by ACTION, i.e. your action is required. */
/*****/
int main ()
{
cout << "Data Management API Example " << endl;
char *vo = "cms"; // ACTION: fill in your correct VO here: gilda !
cout << "-----" << endl;
```

C APIs

Available APIs

```
// Copy a local file to the Storage Element and register it in RLS  
//  
char *localFile = "file:/tmp/test-file"; // ACTION: create a testfile  
char *destSE = "lxb0707.cern.ch"; // ACTION: fill in a specific SE char  
  
*actualGuid = (char*) malloc(50);  
int verbose = 2; // we use verbosity level 2  
int nbstreams = 8; // we use 8 parallel streams to transfer a file  
  
lcg_cr(localFile, destSE, NULL,  
       NULL, vo, NULL, nbstreams,  
       NULL, 0, verbose, actualGuid);  
  
if (errno)  
{  
    perror("Error in copyAndRegister:");  
    return -1;  
} else {  
    cout << "We registered the file with GUID: " << actualGuid << endl;  
}  
cout << "-----" << endl;
```

Copy and Register



Available APIs

```
// Call the listReplicas (lcg_lr) method and print the returned URLs  
//  
// The actualGuid does not contain the prefix "guid:". We add it here and  
// then use the new guid as a parameter to list replicas  
//  
std::string guid = "guid:";  
guid.insert(5,actualGuid);  
char ***pfns = (char***) malloc(200);  
  
lcg_lr((char*) guid.c_str(), vo, NULL, 0, pfns);  
  
if(errno)  
{  
    perror("Error in listReplicas:");  
    free(pfns);  
    return -1;  
} else {  
    cout << "PFN = " << **pfns << endl;  
}  
  
free(pfns);  
  
cout << "-----" << endl;
```

List Replicas

Available APIs

```
// Delete the replica again  
//  
int rc = lcg_del((char*) guid.c_str(), 1, destSE, vo, NULL, 0, verbose);  
  
if(rc != 0)  
{  
    perror("Error in delete:");  
    return -1;  
} else {  
    cout << "Delete OK" << endl;  
}  
  
return 0;  
  
}
```

Delete Replica



Available APIs

```
CC = g++  
GLOBUS_FLAVOR = gcc32
```

```
all: data-management
```

```
data-management: data-management.o  
    $(CC) -o data-management \  
    -L${GLOBUS_LOCATION}/lib -lglobus_gass_copy_${GLOBUS_FLAVOR} \  
    -L${LCG_LOCATION}/lib -llcg_util -lgfal \  
    data-management.o
```

```
data-management.o: data-management.cpp  
    $(CC) -I ${LCG_LOCATION}/include -c data-management.cpp
```

```
clean:  
    rm -rf data-management data-management.o
```

Makefile used



Overview

- **Introduction on Data Management (DM)**
 - **General Concepts**
 - **Some details on transport protocols**
 - **Data management operations**
 - **Files & replicas: Name Convention**
- **File catalogs**
 - **Cataloging requirements and catalogs in egee**
 - **RLS file catalog**
 - **LCG file catalog**
- **DM tools: overview**
- **Data Management CLI**
 - **lcg_utils**
- **Data Management API**
 - **lcg_utils**
- **Advanced concepts**
 - **Advanced utilities: CLI&APIs**
 - **OutputData JDL attribute**
- **Conclusions**



Advanced utilities: `edg-gridftp`

Used for low level management of file/directories in SEs

<code>edg-gridftp-exists</code> TURL	Checks if file/dir exists on a SE
<code>edg-gridftp-ls</code> TURL	Lists a directory on a SE
<code>globus-url-copy</code> srcTURL dstTURL	Copies files between SEs
<code>edg-gridftp-mkdir</code> TURL	Creates a directory on a SE
<code>edg-gridftp-rename</code> srcTURL dstTURL	Renames a file on a SE
<code>edg-gridftp-rm</code> TURL	Removes a file from a SE
<code>edg-gridftp-rmdir</code> TURL	Removes a directory on a SE

edg-gridftp example

Create and delete a directory in a GILDA Storage Element

```
lxb0709.cern.ch - PuTTY
[scampana@grid019:~]$ edg-gridftp-ls --verbose gsiftp://grid3.na.astro.it/flatfiles/SE00/gilda/test
total 0
[scampana@grid019:~]$
[scampana@grid019:~]$
[scampana@grid019:~]$ edg-gridftp-mkdir gsiftp://grid3.na.astro.it/flatfiles/SE00/gilda/test/scampana
[scampana@grid019:~]$
[scampana@grid019:~]$
[scampana@grid019:~]$ edg-gridftp-ls --verbose gsiftp://grid3.na.astro.it/flatfiles/SE00/gilda/test
total 4
drwxrwxr-x    2 gilda006 gilda          4096 Oct 31 19:55 scampana
[scampana@grid019:~]$
[scampana@grid019:~]$
[scampana@grid019:~]$ edg-gridftp-rmdir gsiftp://grid3.na.astro.it/flatfiles/SE00/gilda/test/scampana
[scampana@grid019:~]$
[scampana@grid019:~]$
[scampana@grid019:~]$ edg-gridftp-ls --verbose gsiftp://grid3.na.astro.it/flatfiles/SE00/gilda/test
total 0
[scampana@grid019:~]$
```


Other Advanced CLI&API

- **globus-url-copy srcTURL destTURL**
 - low level file transfer
- Interaction with RLS components
 - edg-lrc command (actions on LRC)
 - edg-rmc command (actions on RMC)
 - C++ and Java API for all catalog operations
 - <http://edg-wp2.web.cern.ch/edg-wp2/replication/docu/r2.1/edg-lrc-devguide.pdf>
 - <http://edg-wp2.web.cern.ch/edg-wp2/replication/docu/r2.1/edg-rmc-devguide.pdf>
- *Using low level CLI and API is STRONGLY discouraged*
 - Risk: loose consistency between SEs and catalogues
 - **REMEMBER:** a file is in Grid if it is **BOTH:**
 - stored in a Storage Element
 - registered in the file catalog

OutputData JDL attribute

- Same as lcg-cr command
- OutputData JDL attribute specifies files to be copied and registered into the Grid
 - The filename (OutputData) is compulsory
 - If no LFN specified (LogicalFileName), none is set!
 - If no SE specified (StorageElement), the default SE is chosen (\$VO_<VO>_DEFAULT_SE)
- At the end of the job the files are moved from WN and registered

```
OutputData = { [  
  OutputFile = "toto.out" ;  
  StorageElement = "adc0021.cern.ch" ;  
  LogicalFileName = "lfn:theBestTotoEver" ;],  
  [  
  OutputFile = "toto2.out" ;  
  StorageElement = "adc0021.cern.ch" ;  
  LogicalFileName = "lfn:theBestTotoEver2" ; ]  
};
```

Overview

- **Introduction on Data Management (DM)**
 - **General Concepts**
 - **Some details on transport protocols**
 - **Data management operations**
 - **Files & replicas: Name Convention**
- **File catalogs**
 - **Cataloging requirements and catalogs in egee**
 - **RLS file catalog**
 - **LCG file catalog**
- **DM tools: overview**
- **Data Management CLI**
 - **lcg_utils**
- **Data Management API**
 - **lcg_utils**
- **Advanced concepts**
 - **Advanced utilities: CLI&APIs**
 - **OutputData JDL attribute**
- **Conclusions**



Summary

- We provided a description to the egee Data Management Middleware Components and Tools
- We described how to use the available CLIs
- Use-case scenarios of Data Movement on Grid
- We presented the available APIs
- An example usage of lcg_util library is shown

Bibliography

- General egee information
 - EGEE Homepage
<http://public.eu-egee.org/>
 - EGEE's NA3: User Training and Induction
<http://www.egee.nesc.ac.uk/>
 - LCG Homepage
<http://lcg.web.cern.ch/LCG/>
 - LCG-2 User Guide
<https://edms.cern.ch/file/454439//LCG-2-UserGuide.html>
 - GILDA
<http://gilda.ct.infn.it/>
 - GENIUS (GILDA web portal)
<http://grid-tutor.ct.infn.it/>

Bibliography

- Information on Data Management middleware
 - LCG-2 User Guide (chapters 3rd and 6th)
<https://edms.cern.ch/file/454439//LCG-2-UserGuide.html>
 - Evolution of LCG-2 Data Management. J-P Baud, James Casey.
<http://indico.cern.ch/contributionDisplay.py?contribId=278&sessionId=7&confId=0>
 - Globus 2.4
<http://www.globus.org/gt2.4/>
 - GridFTP
<http://www.globus.org/datagrid/gridftp.html>

- Information on egee tools and APIs
 - Manpages (in UI)
 - lcg_utils: lcg-* (commands), lcg_* (C functions)
 - Header files (in \$LCG_LOCATION/include)
 - lcg_util.h
 - CVS developement (sources for commands)
<http://isscvcs.cern.ch:8180/cgi-bin/cvsweb.cgi/?hidenonreadable=1&f=u&logsort=date&sortby=file&hideattic=1&cvsroot=lcgware&path=>
- Information on other tools and APIs
 - EDG CLIs and APIs
<http://edg-wp2.web.cern.ch/edg-wp2/replication/documentation.html>
 - Globus
<http://www-unix.globus.org/api/c/> , ...globus_ftp_client/html , ...globus_ftp_control/html