Enabling Grids for E-science in Europe

www.eu-egee.org

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

Grid Data Management

Antonio Delgado Peris
LCG Experiment Integration and Support
CERN IT

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

Agenda

- Introduction to Data Management (DM) in LCG
  - Files and replicas in LCG
  - Storage in LCG
  - File Catalogs in LCG
- DM CLIs & APIs overview
- DM command line tools
  - lcg_utils + edg-gridftp commands
  - OutputData JDL attribute
- DM APIs
  - lcg_utils API
  - GFAL API
  - Globus API

Agenda

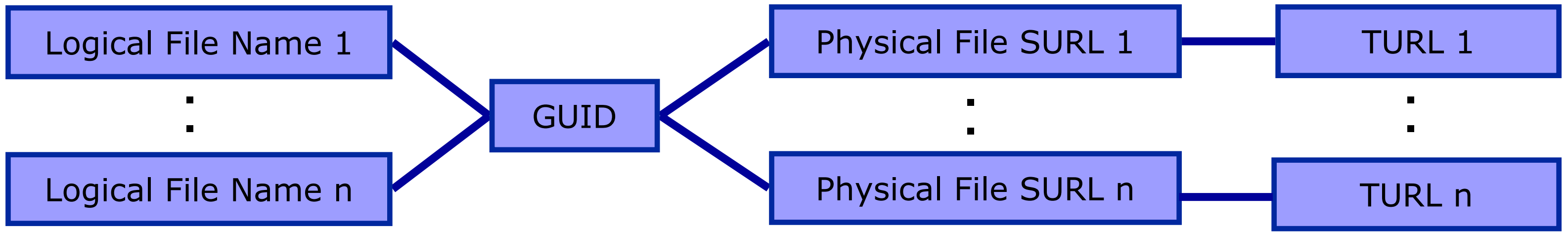
- **Introduction to Data Management (DM) in LCG**
  - **Files and replicas in LCG**
  - **Storage in LCG**
  - **File Catalogs in LCG**
- DM CLIs & APIs overview
- DM command line tools
  - lcg_utils + edg-gridftp commands
  - OutputData JDL attribute
- DM APIs
  - lcg_utils API
  - GFAL API
  - Globus API

Introduction: Files & Replicas

- User and programs produce and require data
- Data may be stored in Grid datasets (files)
  - Located in Storage Elements (SEs)
  - Several replicas of one file in different sites
  - Accessible by Grid users and applications from "anywhere"
  - Locatable by the WMS (data requirements in JDL)
- Also…
  - Resource Broker can send (small amounts of) data to/from jobs: Input and Output Sandbox **(see WMS presentation)**
  - Data may be copied from/to local filesystems (WNs, UIs) to the Grid

Files & replicas: Name Conventions

- Logical File Name (LFN)
  - An alias created by a user to refer to some item of data, e.g. "lfn:cms/20030203/run2/track1"
- Globally Unique Identifier (GUID)
  - A non-human-readable unique identifier for an item of data, e.g. "guid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6"
- Site URL (SURL) (or Physical File Name (PFN) or Site FN)
  - The location of an actual piece of data on a storage system, e.g.
    "srm://pcrd24.cern.ch/flatfiles/cms/output10_1" (SRM)
    "sfn://lxshare0209.cern.ch/data/alice/ntuples.dat" (Classic SE)
- Transport URL (TURL)
  - Temporary locator of a replica + access protocol: understood by a SE, e.g.
    "rfio://lxshare0209.cern.ch//data/alice/ntuples.dat"



Storage in LCG

- The data may be accessed by using:
  - Files transfer: GridFTP (secure, multiple streams) (compulsory)
  - File I/O: RFIO (not GSI enabled) (optional)
    gsidcap (secure access to dCache) (optional)

- Currently supported Storage Elements in LCG-2:
  - Classic SE (disk server): GridFTP + RFIO
  - SRM - dCache disk pools: GridFTP + gsidcap
  - SRM - Castor Mass Storage Systems: GridFTP + RFIO

- Storage Resource Manager (SRM) interface:
  - Additional storage management capabilities
  - SRM protocol for storage management
  - File access using GridFTP, dcap, RFIO...

Some remarks on RFIO

- RFIO requires a specific format in SURLs and TURLs
  - For classic SEs: double slash after the hostname

    sfn://lxb0710.cern.ch//flatfiles/SE00/dteam/my_file

    rfio://lxb0710.cern.ch//flatfiles/SE00/dteam/my_file
  - For Castor backends: the hostname is included in the path

    sfn:///castor/cern.ch/grid/dteam/my_file

    rfio:///castor/cern.ch/grid/dteam/my_file
  - If the catalogs contain incorrect SURLs, programs using LFNs and GUIDs with GFAL and RFIO will fail ➔ this will be solved

- Programs using the RFIO API (e.g., under GFAL)...
  - ...will not work from the UI ➔ Have to be executed in a WN
  - ...will not work from a WN to access SE in a different site (LAN)
  - The reason is that RFIO is not GSI-enabled and requires exact mapping from user’s uids in the WN and the SEs.

Storage: SRM Interface (I)

Enabling Grids for E-science in Europe

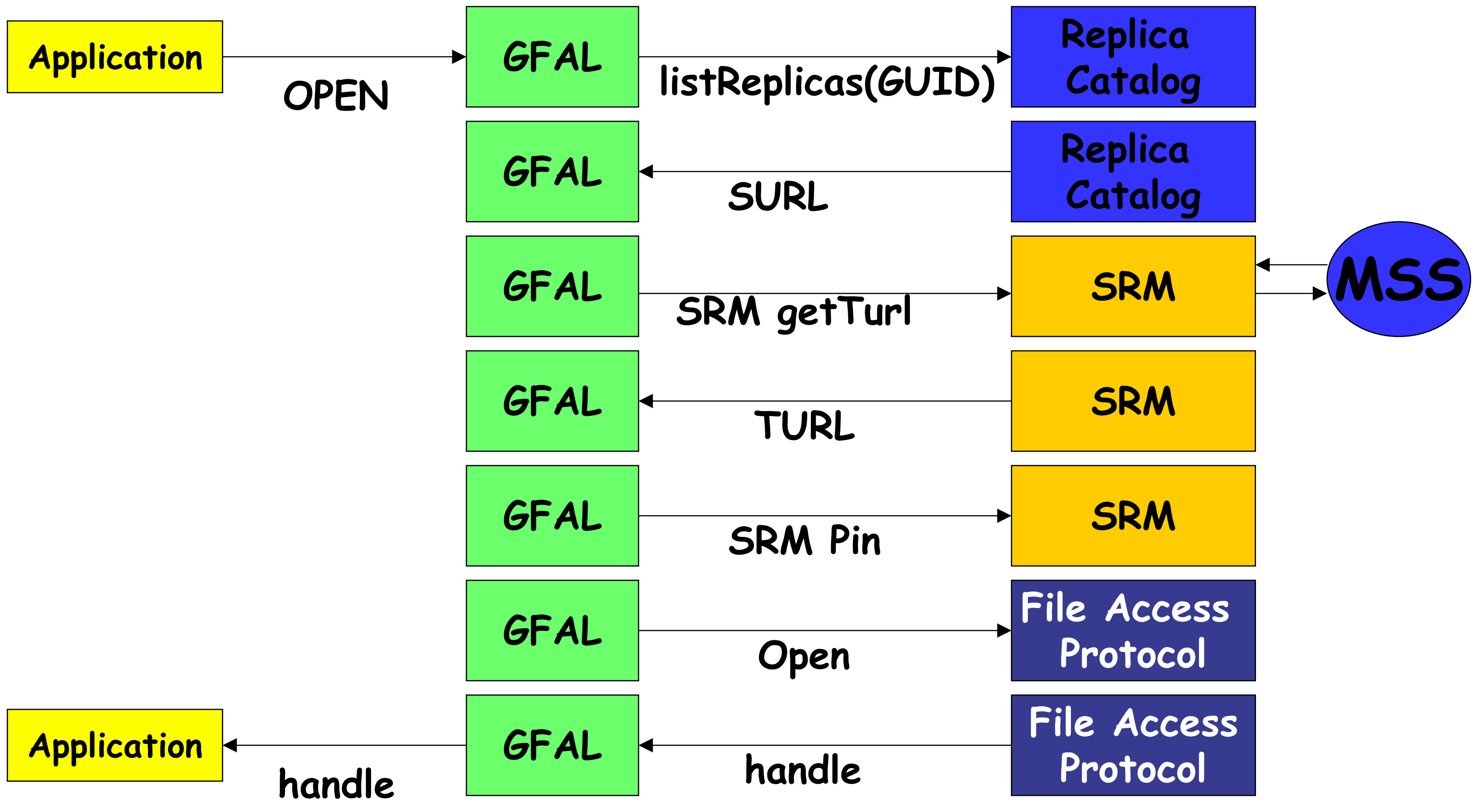
Original SRM design : LBML, JLab, FNAL, CERN

- Transparent migration from tape to disk
- Disk and tape resources are presented as a single element
- Support for local policy
  - Each storage resource managed independently
  - Internal priorities not sacrificed by data movement among Grid agents
- Temporary locking/pinning: read from disk rather than tape
- Reservation on demand and advanced reservation **SRM v2**
  - Reserve space for new files ➔ plan the storage usage
- File status notification
- Estimates on space availability/usage

Storage: SRM Interface (II)

- Lifetime management SRM v2
- Interaction with Grid services (catalogues, Grid agents...)
  - Notification of file additions, deletions, metadata changes...
  - Bi-directional (could influence file deletion policy of SRM)
- Pull/push mechanism for read-only/new files
  - The server does not contact the client
- Multiple-file requests
- Asynchronous and synchrounos operations
- Multiple protocols
  - Data Movement protocols (GridFTP, BBFTP, ...)
  - Request protocols (SOAP over HTTPS)
  - Security-releated protocols (authority information kept on the SRM)

Storage: SRM Interface (III)



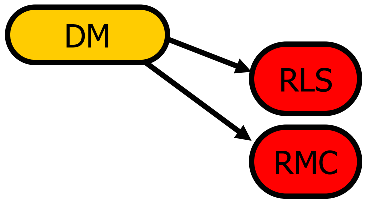
File Catalogs in LCG

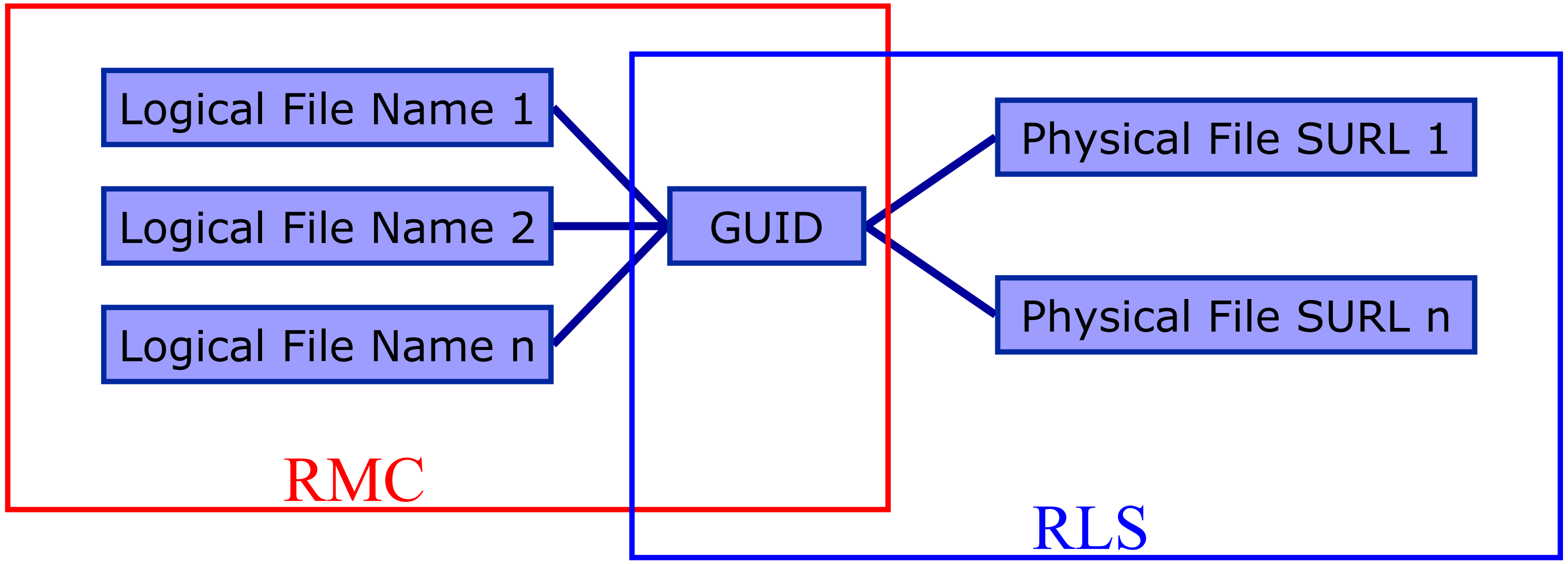
Enabling Grids for E-science in Europe

- File catalogs in LCG:
  - They keep track of the location of copies (replicas) of Grid files
  - The DM tools and APIs and the WMS interact with them
- 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 Data Challenges
- New LCG File Catalog (LFC)
  - In production in next LCG release; deployment in January 2005
  - Coexistence with RLS; migration tools provided
  - Accessible by defining: $LCG_CATALOG_TYPE=lfc and $LFC_HOST
  - Better performance and scalability
  - Provides new features: security, hierarchical namespace, transactions...

File Catalogs: The RLS

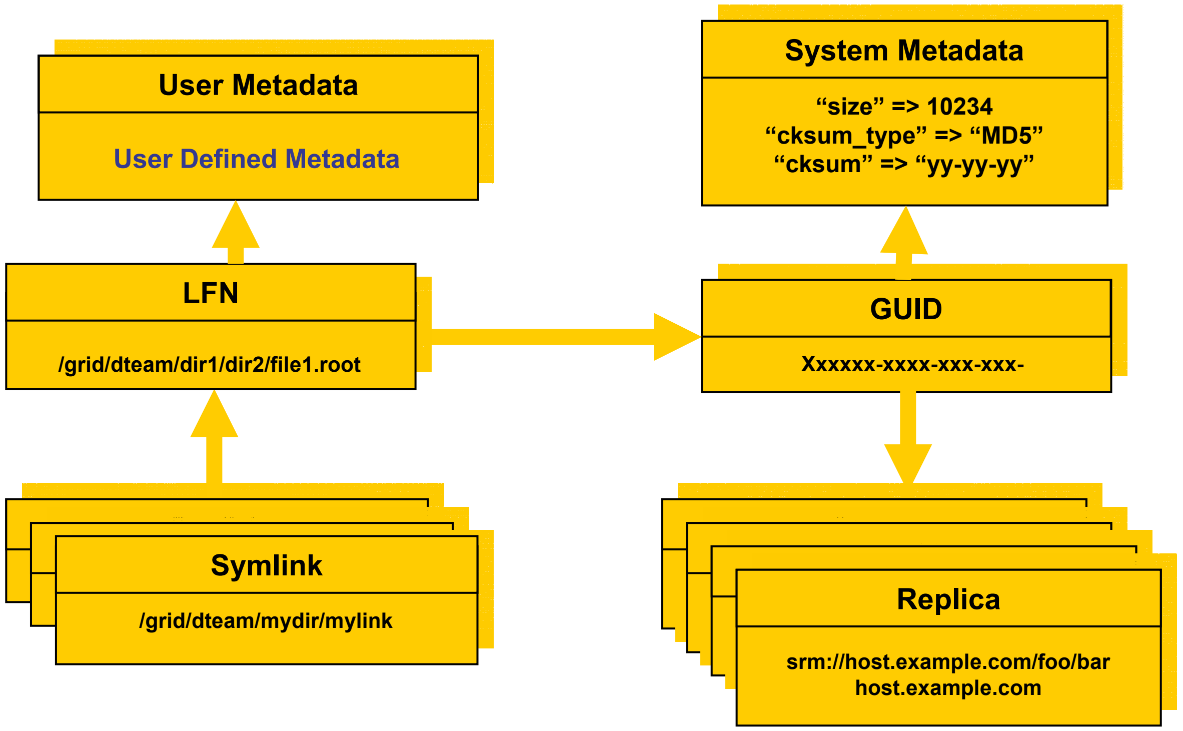
- RMC:
  - Stores LFN-GUID mappings
  - Accessible by edg-rmc CLI + API
- RLS:
  - Stores GUID-SURL mappings
  - Accessible by edg-lrc CLI + API





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



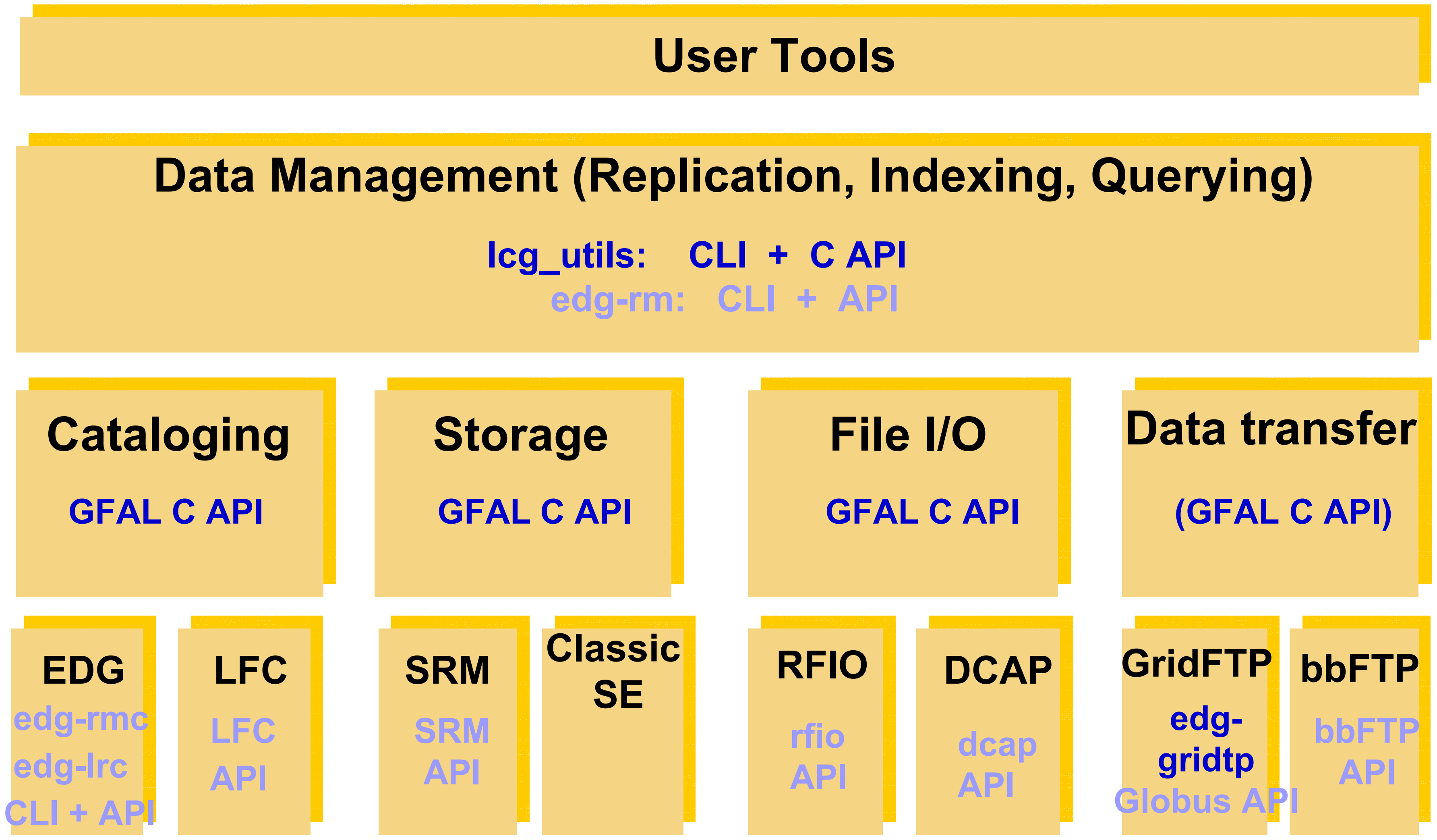
File Catalogs: The LFC (II)

- 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)
  - 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

Agenda

- Introduction to Data Management (DM) in LCG
  - Files and replicas in LCG
  - Storage in LCG
  - File Catalogs in LCG
- **DM CLIs & APIs overview**
- DM command line tools
  - lcg_utils + edg-gridftp commands
  - OutputData JDL attribute
- DM APIs
  - lcg_utils API
  - GFAL API
  - Globus API

DM CLIs & APIs overview



DM CLIs & APIs: available tools

- lcg_utils: lcg-* commands + lcg_* API calls
  - Provide (all) the functionality needed by the LCG user
  - Transparent interaction with file catalogs and storage interfaces when needed
  - Abstraction from technology of specific implementations

- Grid File Access Library (GFAL): API
  - Adds file I/O and explicit catalog interaction functionality
  - Still provides the abstraction and transparency of lcg_utils

- edg-gridftp tools: CLI
  - Complete the lcg_utils with GridFTP operations
  - Functionality available as API in GFAL
  - May be generalized as lcg-* commands

DM CLIs & APIs: Old EDG tools

- All-purpose CLIs and APIs for EDG and LCG

- File & replica management
  - edg-rm
- Catalog interaction (only for EDG catalogs)
  - edg-lrc
  - edg-rmc

- Use discouraged
  - Worst performance (slower) than lcg_utils
  - New features added only to lcg_utils
  - Less general than GFAL and lcg_utils
  - When the EDG file catalog gets replaced by the LFC, these commands will stop working

DM CLIs & APIs: Other APIs

- File I/O protocols: CLIs + APIs
  - rfio (for Castor and classic SE) and gsidcap (for dCache)
  - rfio can only be used from a WN to a local SE (not GSI enabled)
  - Used transparently by GFAL depending on type of SE
  - Direct use discouraged in favor of GFAL
- Globus API
  - Very low-level API for GridFTP (also for WM, IS…)
  - Use discouraged in favor of GFAL
- Implementation-specific APIs
  - LCG File Catalog API, SRM API, bbFTP API
  - Use discouraged in favor of the more general GFAL
  - Required advanced functionality can be added to GFAL !

DM CLIs & APIs: LFC API

Low level methods (many POSIX-like):

lfc_access	lfc_deleteclass	lfc_listreplica	lfc_setacl
lfc_aborttrans	lfc_delreplica	lfc_lstat	lfc_setatime
lfc_addreplica	lfc_endtrans	lfc_mkdir	lfc_setcomment
lfc_apiinit	lfc_enterclass	lfc_modifyclass	lfc_seterrbuf
lfc_chclass	lfc_errmsg	lfc_opendir	lfc_setfsize
lfc_chdir	lfc_getacl	lfc_queryclass	lfc_starttrans
lfc_chmod	lfc_getcomment	lfc_readdir	lfc_stat
lfc_chown	lfc_getcwd	lfc_readlink	lfc_symlink
lfc_closedir	lfc_getpath	lfc_rename	lfc_umask
lfc_creat	lfc_lchown	lfc_rewind	lfc_undelete
lfc_delcomment	lfc_listclass	lfc_rmdir	lfc_unlink
lfc_delete	lfc_listlinks	lfc_selectsrvr	lfc_utime
			send2lfc

Agenda

- Introduction to Data Management (DM)
  - Files and replicas
  - Storage in LCG
  - File Catalogs in LCG
- DM CLIs & APIs overview
- **DM command line tools**
  - **lcg_utils + edg-gridftp commands**
  - **OutputData JDL attribute**
- DM APIs
  - lcg_utils API
  - GFAL API
  - Globus API

lcg_utils: Data transfer & Storage

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-infosites	Gives information about resources on the Grid
lcg-rep	Copies a file from SE to SE and registers it in the LRC

lcg-gt	Gets the TURL for a given SURL and transfer protocol
lcg-sd	Sets file status to “Done” in a specified request

lcg_utils: Catalog interaction

lcg-aa	Adds an alias for a given GUID
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 for a given GUID
lcg-rf	Registers an existing SE file in the catalog
lcg-uf	Unregisters a file residing on an SE

EDG gridftp commands

edg-gridftp-exists URL **gfal_stat (…)**	Checks if file/dir exists on an SE
edg-gridftp-ls URL **gfal_opendir (…)**	Lists a directory on a SE
edg-gridftp-mkdir URL **gfal_mkdir (…)**	Creates a directory on a SE
edg-gridftp-rename srcURL dstURL **gfal_rename (…)**	Renames a file on a SE
edg-gridftp-rm URL **gfal_unlink (…)**	Removes a file from a SE
edg-gridftp-rmdir URL **gfal_rmdir (…)**	Removes a directory on a SE
globus-url-copy srcURL dstURL **lcg_cp (…)**	Copies files between SEs

OutputData JDL attribute

- The 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 first close SE is chosen

- At the end of the job the files are moved from the WN and registered

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

Hands-on time!

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

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

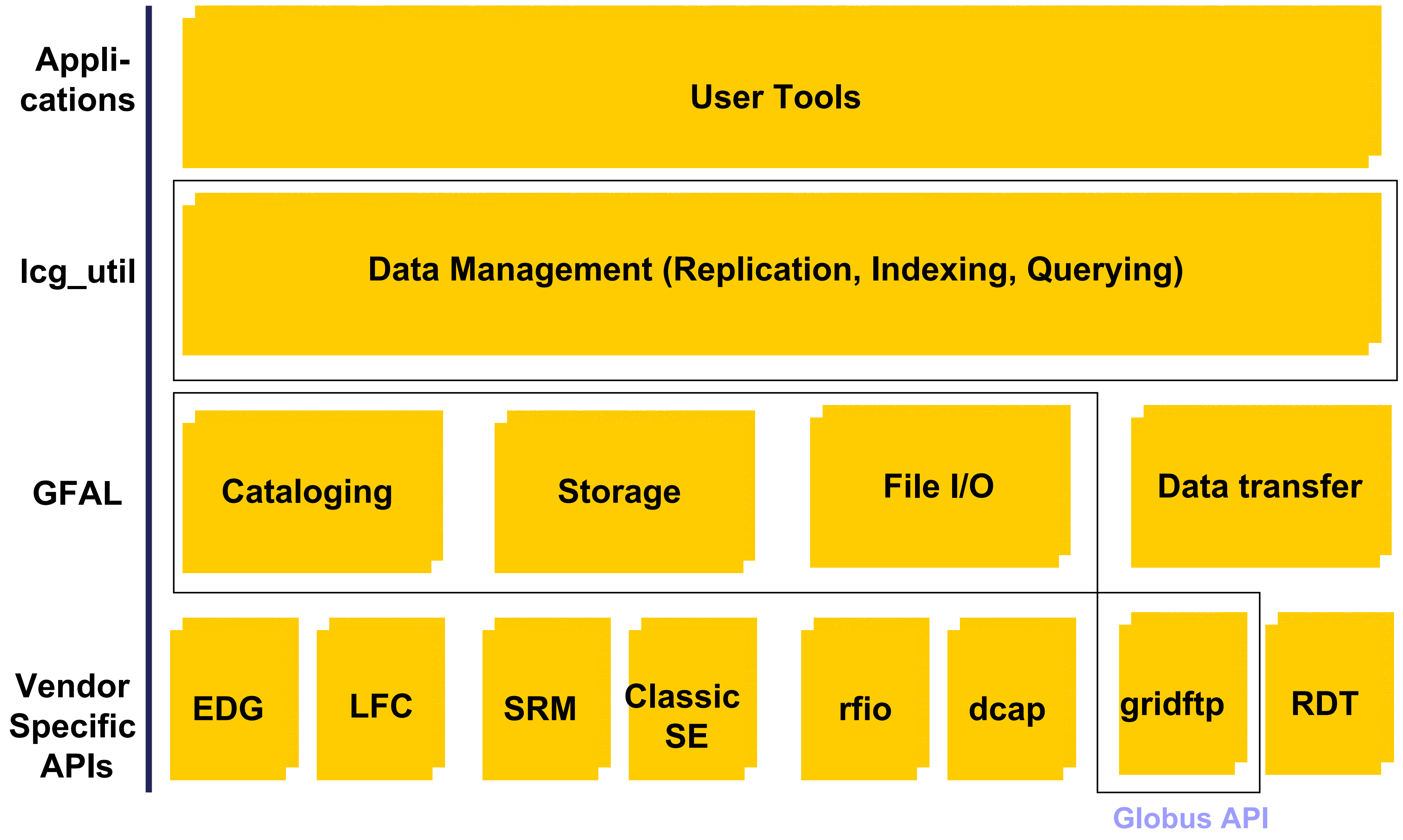
2. Submit a job that creates a file and automatically brings it to the Grid (OutputData attribute).
3. Check the LFN, GUID and SURL of the file.
4. Copy the file to two SEs with lcg-rep and lcg-cp. Check that both files exist but only one is registered.
5. Register the file that was not already in the catalog.
6. Copy the file back to the UI and check it.
7. Delete and unregister all the files and replicas created.

(See the DM-HandsOn presentation)

Agenda

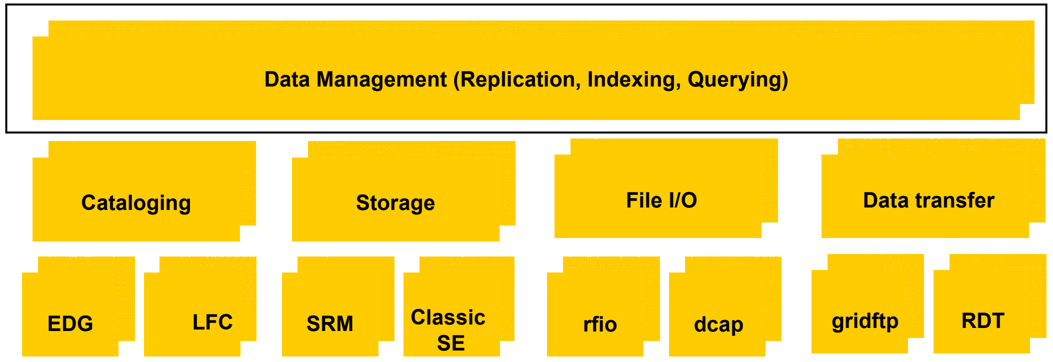
- Introduction to Data Management (DM) in LCG
  - Files and replicas in LCG
  - Storage in LCG
  - File Catalogs in LCG
- DM tools & APIs overview
- DM command line tools
  - lcg_utils + edg-gridftp commands
  - OutputData JDL attribute
- **DM APIs**
  - **lcg_utils API**
  - **GFAL API**
  - **Globus APi**

Layered Data Management APIs



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   (+ libgfal.so)
- Single header file
  - lcg_util.h



lcg_utils: Data Transfer and Storage

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

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

lcg_utils: Catalog interaction

Enabling Grids for E-science in Europe

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

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);

Hands-on time!

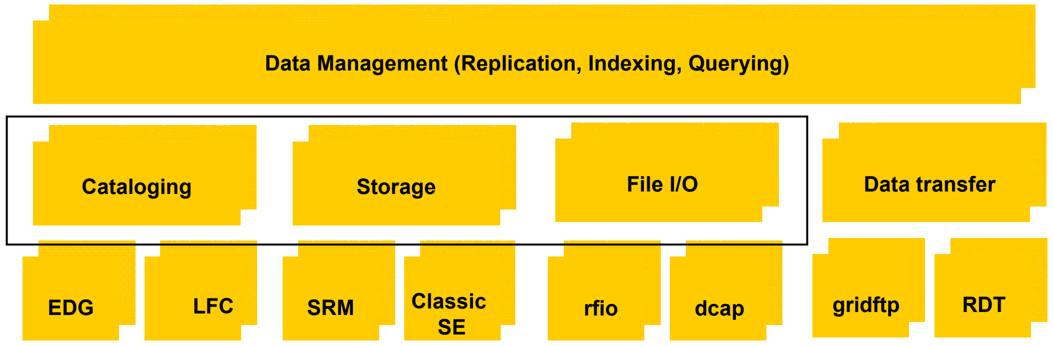
Enabling Grids for E-science in Europe

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

(See the DM-HandsOn presentation)

Grid File Access Library

- GFAL is a library to provide access to Grid files
  - File I/O, Catalog Interaction, Storage Interaction
- Abstraction from specific implementations
- Transparent interaction with the information service, the file catalogs…
- Single shared library in threaded and unthreaded versions
  - libgfal.so, libgfal_pthr.so
- Single header file
  - gfal_api.h



GFAL: Catalog API

```
int create_alias (const char *guid, const char *lfn, long long size)
int guid_exists (const char *guid)
char *guidforpfn (const char *surl)
char *guidfromlfn (const char *lfn)
char **lfnsforguid (const char *guid)
int register_alias (const char *guid, const char *lfn)
int register_pfn (const char *guid, const char *surl)
int setfilesize (const char *surl, long long size)
char *surlfromguid (const char *guid)
char **surlsfromguid (const char *guid)
int unregister_alias (const char *guid, const char *lfn)
int unregister_pfn (const char *guid, const char *surl)
```

GFAL: Storage API

int **deletesurl** (const char *surl)

int **getfilemd** (const char *surl, struct stat64 *statbuf)

int **set_xfer_done** (const char *surl, int reqid, int fileid, char *token, int oflag)

int **set_xfer_running** (const char *surl, int reqid, int fileid, char *token)

char ***turlfromsurl** (const char *surl, char **protocols, int oflag, int *reqid, int *fileid, char **token)

int **srm_get** (int nbfiles, char **surls, int nbprotocols, char **protocols, int *reqid, char **token, struct srm_filestatus **filestatuses)

int **srm_getstatus** (int nbfiles, char **surls, int reqid, char *token, struct srm_filestatus **filestatuses)

GFAL: File I/O API (I)

Enabling Grids for E-science in Europe

int **gfal_access** (const char *path, int amode);

int **gfal_chmod** (const char *path, mode_t mode);

int **gfal_close** (int fd);

int **gfal_creat** (const char *filename, mode_t mode);

off_t **gfal_lseek** (int fd, off_t offset, int whence);

int **gfal_open** (const char * filename, int flags, mode_t mode);

ssize_t **gfal_read** (int fd, void *buf, size_t size);

int **gfal_rename** (const char *old_name, const char *new_name);

ssize_t **gfal_setfilchg** (int, const void *, size_t);

int **gfal_stat** (const char *filename, struct stat *statbuf);

int **gfal_unlink** (const char *filename);

ssize_t **gfal_write** (int fd, const void *buf, size_t size);

GFAL: File I/O API (II)

int **gfal_closedir** (DIR *dirp);

int **gfal_mkdir** (const char *dirname, mode_t mode);

DIR ***gfal_opendir** (const char *dirname);

struct dirent ***gfal_readdir** (DIR *dirp);

int **gfal_rmdir** (const char *dirname);

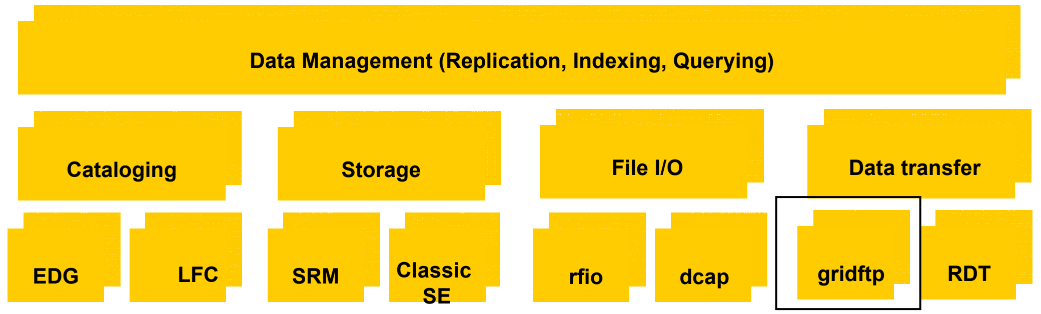
Hands-on time!

1. Check the syntax of the GFAL API in the manpages and in $LCG_LOCATION/include/gfal_api.h .
2. Submit a job that opens the previously created “file1” and reads only the first 20 lines. Retrieve them with the standard output of the job.
3. Submit a job that creates a file, writes the date in it every second for 10 seconds, and registers the file.
4. Compare access time for files in CASTOR that have been pre-staged and files only in tape. Create an application that asks for the stage of a file. Enhance the application so that it waits for the staging to complete, and then copies the file to the local filesystem.

(See the DM-HandsOn presentation)

Globus GridFTP API

- Some LCG middleware is based on Globus
  - Globus (2.4.3) libraries included in VDT (1.14)
  - In Data Management: GSIFTP (GridFTP)
- Globus provides a low-level API
  - Lots of libraries in $GLOBUS_LOCATION/lib/ (/opt/globus)
  - Header files in /opt/globus/<globus_flavor>/include/ (gcc32, gcc32pthr)
  - GridFTP functionality:
    - libglobus_ftp_client_<flavor>.so
    - libglobus_ftp_control_<flavor>.so
    - globus_ftp_client.h
    - globus_ftp_control.h



Globus API: Header files

- Some globus libraries have hidden dependencies
  - This must be known when linking statically
  - Include directives must be added, libraries linked, variables defined

- The globus-makefile-header command can be used
  - $GLOBUS_LOCATION/bin/globus-makefile-header \
    --flavor=<flavor> [--static] <package_name>
  - List of packages: $GLOBUS_LOCATION/etc/globus_packages/
  - GridFTP packages: globus_ftp_client, globus_ftp_control
  - The output of the command can be included in the makefile

Globus API: Modules and Callbacks

- The Globus API is split into different module groups.
  - A module must be activated before any function within it can be used.
    - globus_module_activate(GLOBUS_FTP_CLIENT_MODULE)
    - globus_module_deactivate(<module>),   globus_module_deactivate_all()

- Two types of operations in Globus: blocking and asynchronous
  - Callback: function provided as a parameter to an asynchronous call
  - The callback is called by the Globus framework on completion or status change. It runs on a separate thread from the main program
  - The main thread must wait for the callback
    - globus_cond_wait(&condition, &mutex),   globus_cond_signal(&condition)
  - Mutex can be used to ensure thread-safety
    - globus_mutex_lock(&mutex),   globus_mutex_unlock(&mutex)

Globus example: our gridftp-exists

- Initialization of module and necessary variables

```
/* Module initialization */
    int status;
    status=globus_module_activate(GLOBUS_FTP_CLIENT_MODULE);

/* Create an empty ftp client attribute set */
    globus_result_t gresult;
    globus_ftp_client_handleattr_t ftp_handle_attr;
    gresult = globus_ftp_client_handleattr_init(&ftp_handle_attr);

/* Create an ftp client handle */
    globus_ftp_client_handle_t ftp_handle;
    gresult = globus_ftp_client_handle_init(&ftp_handle, &ftp_handle_attr);

/* Create an ftp operations attribute */
    globus_ftp_client_operationattr_t ftp_op_attr;
    gresult = globus_ftp_client_operationattr_init(&ftp_op_attr);
```

Our gridftp-exists: Shared data

- Shared object between application and callback function

```
class CBData {
        globus_mutex_t mutex;                          // To lock the shared data
        globus_cond_t cond;                            // The condition for wait and signal
        globus_bool_t done;                            // Boolean: finished?
        globus_bool_t failed;                          // Error or success for last call
   public:
        CBData(); ~CBData();                           // Constructor & Destructor
        globus_bool_t isDone();                        // Check condition
        void setDone();                                // Set as done and signal
        void setFailed(globus_object_t error);         // Set error or success value
        void waitForDone();                            // Wait for completion
};
```

```
void CallBackData::waitForDone(){
        globus_mutex_lock(&mutex);
        while(!isDone())
          globus_cond_wait(&cond, &mutex);
        globus_mutex_unlock(&mutex);
      }
```

```
void CallBackData::setDone(){
  globus_mutex_lock(&mutex);
  done = GLOBUS_TRUE;
  globus_cond_signal(&cond);
  globus_mutex_unlock(&mutex);
}
```

Our gridftp-exists: Callback function

- The callback function

```
static void existsCallback( void* user_cb_arg,
                            globus_ftp_client_handle_t *handle,
                            globus_object_t *error) {

    CBData * shared = (CBData*) user_cb_arg;

    if (error != GLOBUS_SUCCESS)    cerr << “Error (file may not exist)" << endl;
    else                            cout << "File exists" << endl;

    shared->setFailed(error);
    shared->setDone();
}
```

- The asynchronous call (in the main program)

```
CBData existsData;
gresult = globus_ftp_client_exists( &ftp_handle,   url,   &ftp_op_attr,
                                    &existsCallback,    (void *) &existsData);
existsData.waitForDone();
```

Hands-on time!

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

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

(See the DM-HandsOn presentation)

Summary

… and that was it.

- We saw an introduction to the LCG-2 Data Management architecture
  - Different types of SEs, file catalogs, SRM interface
- We described and saw examples of the available CLIs
  - *lcg_util* and *edg-gridftp* commands
- We presented and exercised the available APIs
  - *lcg_util* and *GFAL*
- We showed how to use the *Globus GridFTP* API

See also the APIs-Tutorial-DM-HandsOn presentation
http://agenda.cern.ch/askArchive.php?base=agenda&categ=a044732&id=a044732s1t2/transparencies

Bibliography

- General LCG-2 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 (GIDA 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
  - bbFTP
    http://doc.in2p3.fr/bbftp/

- Information on Storage Elements
  - SRM:
    http://sdm.lbl.gov/srm-wg/
  - CASTOR:
    http://castor.web.cern.ch/castor/
  - dCache:
    http://www.dcache.org/

Bibliography

- Information on LCG tools and APIs
  - Manpages (in UI)
    - lcg_utils: lcg-* (commands), lcg_* (C functions)
    - GFAL: gfal_*   (the rest of the commands will be added)
  - Header files (in $LCG_LOCATION/include)
    - lcg_util.h,   gfal_api.h
  - CVS developement (sources for LCG commands)

    http://isscvs.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
  - RFIO

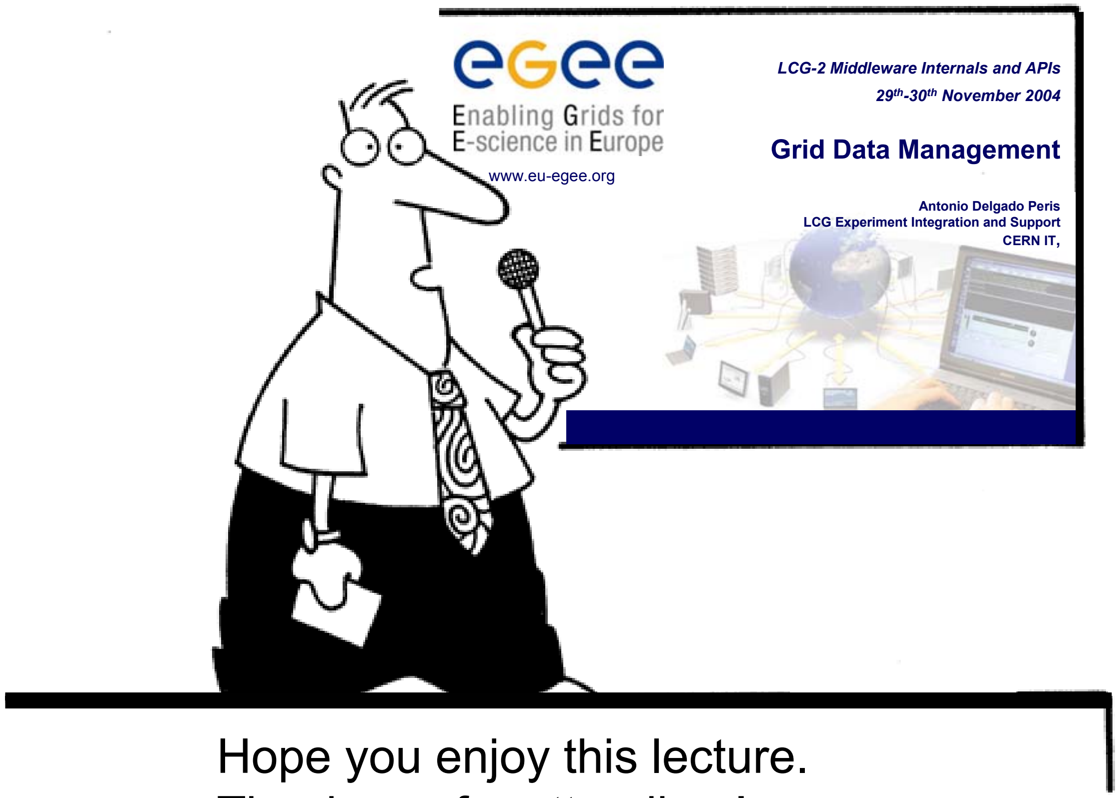
    http://doc.in2p3.fr/doc/public/products/rfio/rfio.html   (In French!)
  - dcap

    http://www.dcache.org/manuals/libdcap.shtml
  - Globus

    http://www-unix.globus.org/api/c/ , ...globus_ftp_client/html , ...globus_ftp_control/html
  - Article on Globus usage (callbacks, etc)

    http://www-106.ibm.com/developerworks/grid/library/gr-cglobus/

The End


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

Hope you enjoy this lecture.
Thank you for attending !