Enabling Grids for E-sciencE

gLite Data Management Services

Emidio Giorgio

Giuseppe Platania

INFN

EGEE Tutorial

Roma, 03.11.2005

www.eu-egee.org

Information Society

Outline

- Grid Data Management Challenge
- Storage Elements, SRM and glite I/O
- File and Replica Catalog
- LFC (LFC File Catalog)
- File Transfer Components
- AMGA Metadata Catalog

The Grid DM Challenge

- **Heterogeneity**
  - Data are stored on different storage systems using different access technologies
  - Need common interface to storage resources
    - Storage Resource Manager (SRM)
- **Distribution**
  - Data are stored in different locations – in most cases there is no shared file system or common namespace
  - Data need to be moved between different locations
  - Need to keep track where data is stored
    - File and Replica Catalogs
  - Need scheduled, reliable file transfer
    - File transfer and placement services

Data Management Services

- **Storage Element** – **common interface to storage**
  - Storage Resource Manager — Castor, dCache, DPM, …
  - POSIX-I/O — gLite-I/O
  - Native Access protocols — rfio, dcap
  - Transfer protocols — gsiftp
- **Catalogs** – **keep track where data are stored**
  - File Catalog
  - Replica Catalog
  - File Authorization Service
  - Metadata Catalog

  gLite File and Replica Catalog

  FireMan

  LFC

  AMGA Metadata Catalogue
- **File Transfer** – **schedules reliable file transfer**
  - Data Scheduler — (only designs exist so far)
  - File Transfer Service — gLite FTS

    (manages physical transfers)
  - File Placement Service — gLite FPS

    (FTS and catalog interaction in a transactional way)

Data services in gLite

- **File Access Patterns:**
  - Write once, read-many
  - Rare append-only updates with one owner
  - Frequently updated at one source - replicas check/pull new version
  - (*NOT* frequent updates, many users, many sites)
- **File naming**
  - Mostly, see the “logical file name” (LFN)
  - LFN must be unique:
    - includes logical directory name
    - in a VO namespace
  - E.g. /gLite/myVOname.org/runs/12aug05/data1.res
- **3 service types for data**
  - Storage
  - Catalogs
  - Movement

SRM in an example

She is running a job which needs:
Data for physics event reconstruction
Simulated Data
Some data analysis files
She will write files remotely too

They are at CERN
In dCache

They are at Fermilab
In a disk array

They are at Nikhef
in a classic SE

SRM in an example

dCache

Own system, own protocols and parameters

classic SE

Independent system from dCache or Castor

Castor

No connection with dCache or classic SE

SRM

I talk to them on your behalf
I will even allocate space for your files
And I will use transfer protocols to send your files there

Storage Resource Management

- Data are stored on **disk pool servers** or **Mass Storage Systems**
- storage resource management needs to take into account
  - Transparent access to files (migration to/from disk pool)
  - Space reservation
  - File status notification
  - Life time management
- **SRM (Storage Resource Manager)** takes care of all these details
  - SRM is a Grid Service that takes care of local storage interaction and provides a Grid interface to outside world

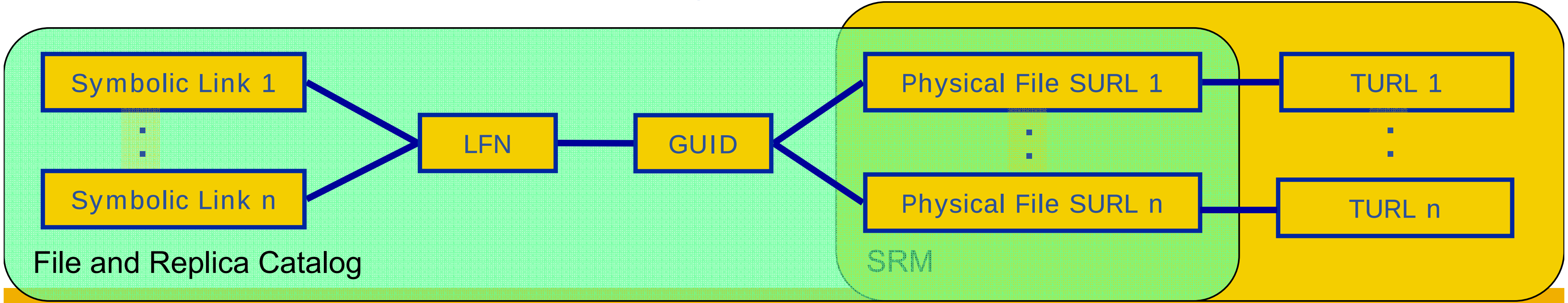
- Interactions with the SRM is hidden by higher level services (glite I/O)

Grid Storage Requirements

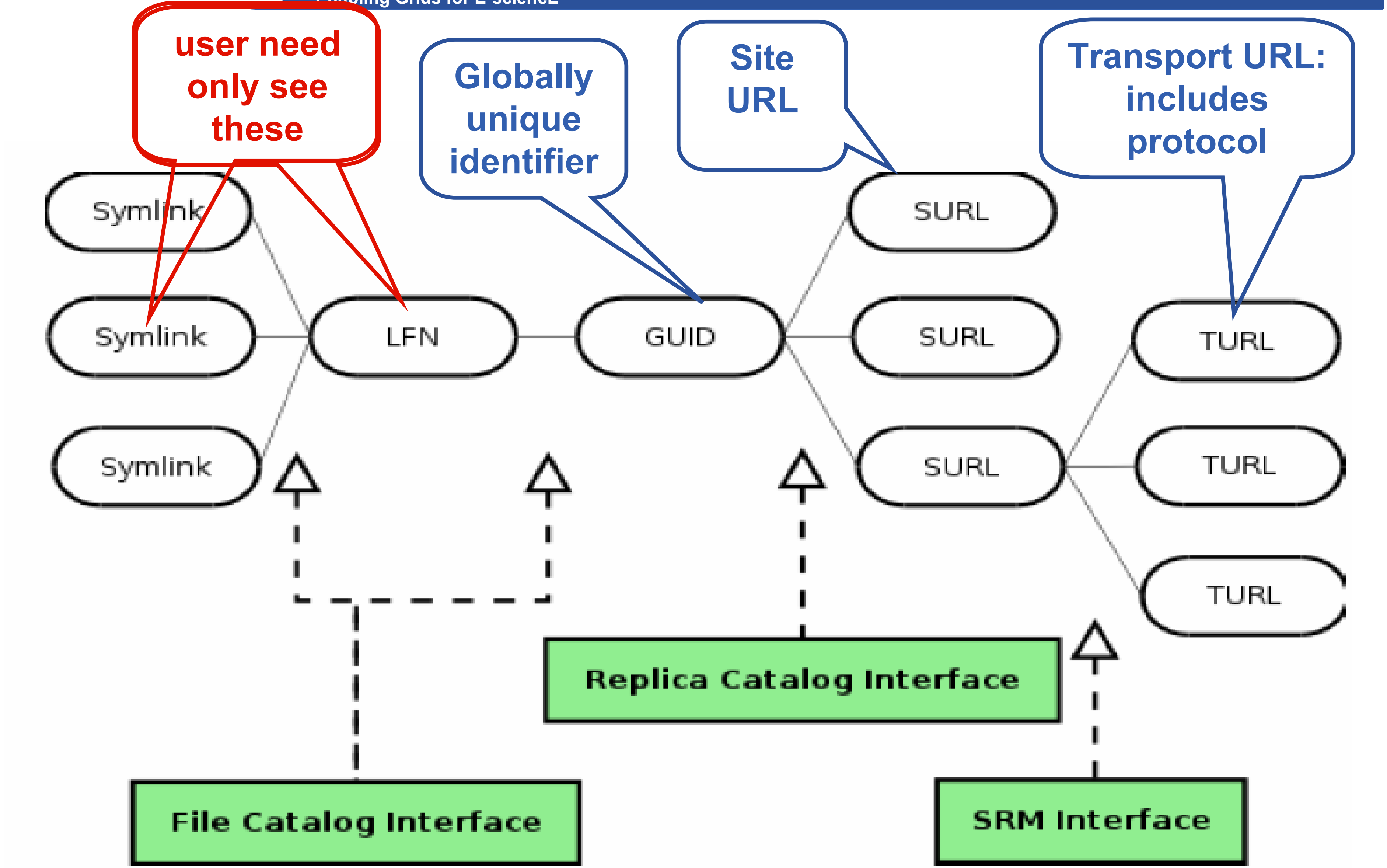
- **Manage local storage and interface to Mass Storage Systems like**
  - HPSS, CASTOR, DiskeXtender (UNITREE), …
- **Provide an SRM interface**
- **Support basic file transfer protocols**
  - GridFTP mandatory
  - Others if available (https, ftp, etc)
- **Support a native I/O access protocol**
  - POSIX (like) I/O client library for direct access of data

Files & replicas: Name Conventions

- **Symbolic Link** in logical filename space
- **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"



File names and identifiers in gLite



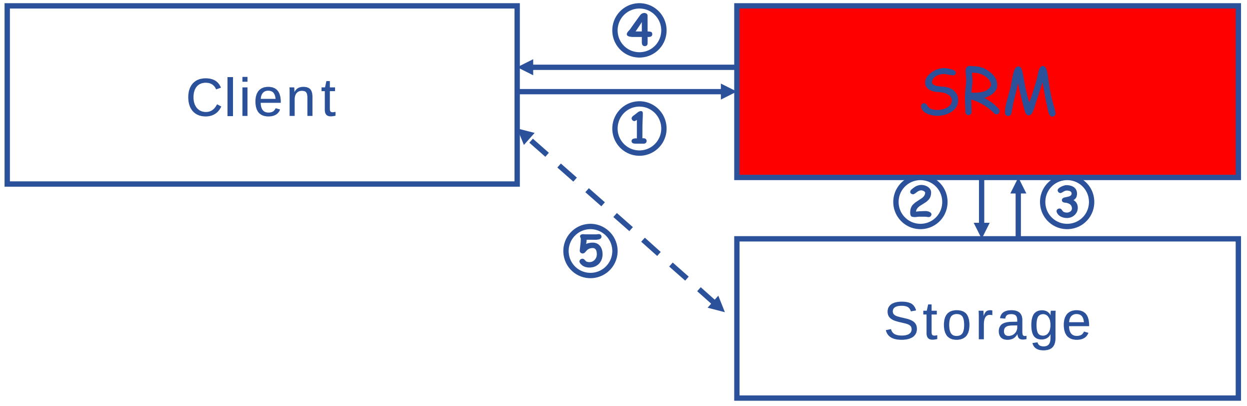
user need only see these

Globally unique identifier

Site URL

Transport URL: includes protocol

SRM Interactions



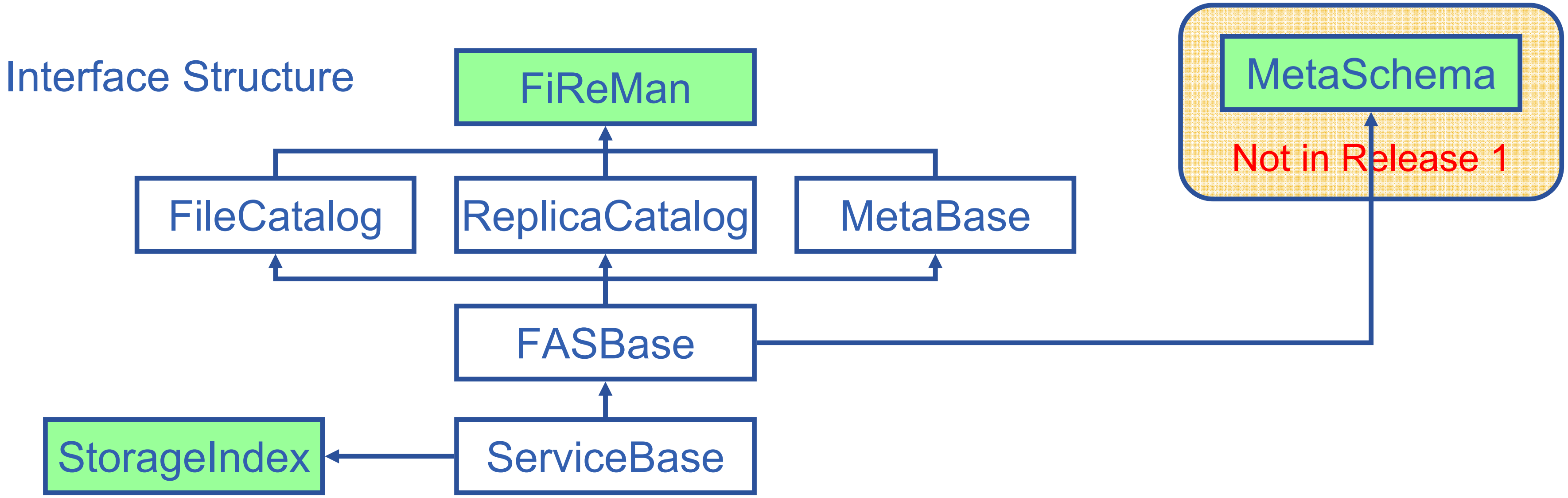
1. The client asks the SRM for the file providing an SURL (Site URL)
2. The SRM asks the storage system to provide the file
3. The storage system notifies the availability of the file and its location
4. The SRM returns a TURL (Transfer URL), i.e. the location from where the file can be accessed
5. The client interacts with the storage using the protocol specified in the TURL

FireMan: gLite File and Replica Catalog

- **File Catalog**
  - Allows for operation on the logical file namespaces that it manages (ex: making directories, renaming files, creating symbolic link)
  - Manages LFNs, keeping internally LFN-GUID mappings
- **Replica Catalog**
  - Exposes operations concerning the replication aspect of the grid files (ex: listing, adding and removing replicas to a file identified by its GUID)
  - Gives access to the GUID-SURL mappings
- **File Authorization Service (FAS)**
  - Request authorization - based on the DN and the Groups from the user’s delegated credentials
- **StorageIndex**
  - Allows WMS interactions (file location for the RB)
- **Metadata Catalog**
  - File-Based Metadata (not currently available)
- **Fireman = File and Replica Manager**
  - Provides all the previous services

Fireman Catalog Interface

Logical File Namespace management	FileCatalog
Replica locations	ReplicaCatalog
File-based metadata	MetaBase
Metadata Management	MetaSchema
Authentication and Authorization information (ACLs)	FASBase
Service Metadata	ServiceBase
WMS interaction and global file location	StorageIndex



Interface Structure

FiReMan

FileCatalog

ReplicaCatalog

MetaBase

FASBase

ServiceBase

StorageIndex

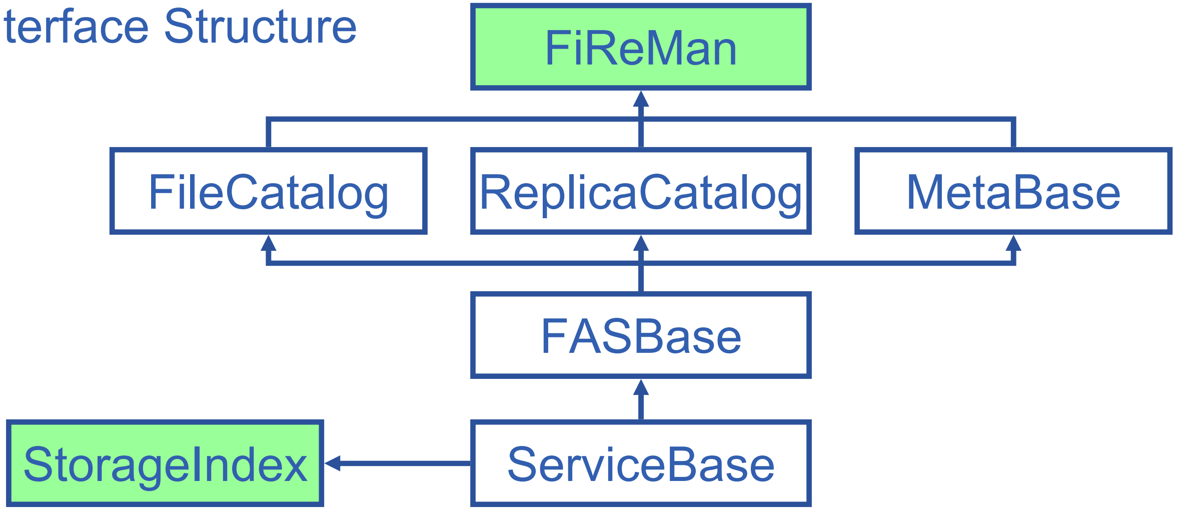
MetaSchema

Not in Release 1

gLite FiReMan Catalog details

- **Web Service interface (WSDL)**
- **Mostly Bulk operations**
- **Stateless interaction**
- **No transactions outside Bulk**

Interface Structure



- StorageIndex: file location for broker
- FAS: File Access Service (ACLs)
- File Catalog: directory structure in LFN namespace
- Replica Catalog: location of replicas
- Meta: additional (user defined metadata)

Implemented on top of Oracle and MySQL

LFC OVERVIEW

(LCG File Catalogue)

Giuseppe Platania

INFN Catania

EGEE Tutorial

Rome, November 2-4, 2005

Information Society

OUTLINE

- DESCRIPTION
- ARCHITECTURE
- LFC vs EDG RLS
- REQUIREMENTS
- INSTALLATION & CONFIGURATION

DESCRIPTION

The LCG File Catalog fixes the performance and scalability problems of EDG file catalogs.

Provides

- Cursors for large queries,
- Timeouts and retries for client operations.

Added features :

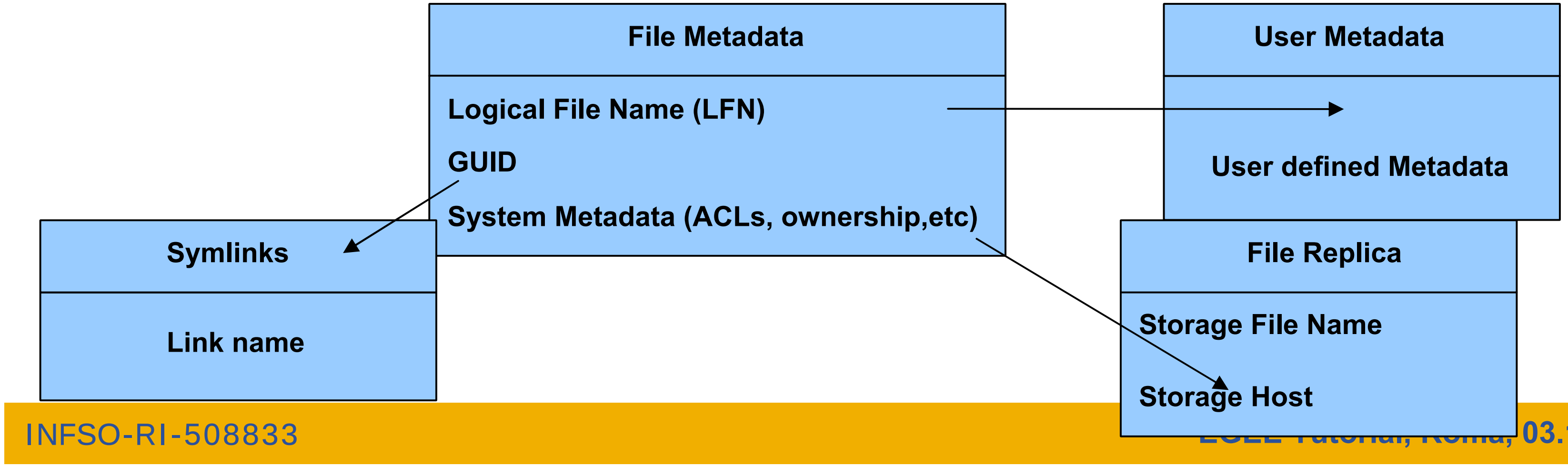
- User exposed transaction API,
- Hierarchical namespace and namespace operations,
- Integrated GSI Authentication and Authorization,
- Access Control Lists (Unix Permissions and POSIX ACLs),
- Checksums

Supported database backends : Oracle and MySQL

GFAL integration and support to lcg-* done by Grid Deployment group

ARCHITECTURE

- LFC stores both logical and physical mappings for the file in the same database → Speeds up of operations
- Entities treating UNIX filesystem-like .
- The file API also similar to UNIX filesystem API (create(), mkdir(), chown()….)
- Hierarchical namespace of Logical File Names (LFNs) mapped to the GUIDs.
- GUIDs mapped to the physical locations of file replicas in storage
- System attributes of files (creation time, file size and checksum…) stored as LFN attributes
- One field for user-defined metadata
- Multiple LFNs per GUID allowed as symbolic links to the primary LFN.



File Metadata

Logical File Name (LFN)

GUID

System Metadata (ACLs, ownership,etc)

User Metadata

User defined Metadata

Symlinks

Link name

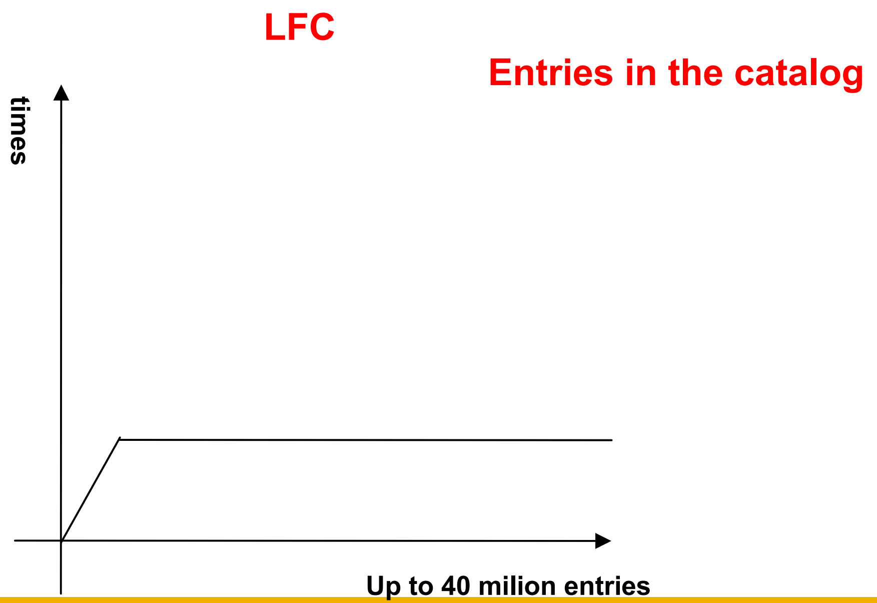
File Replica

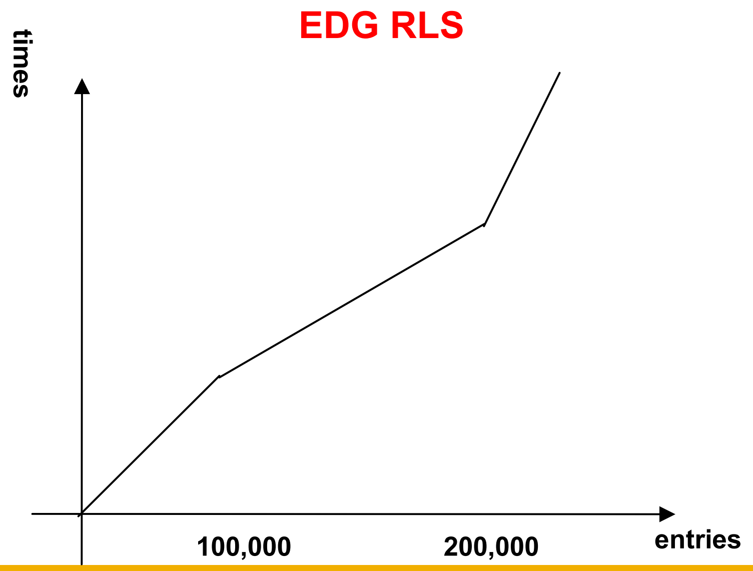
Storage File Name

Storage Host

LFC vs EDG RLS

- LFC
  - No significant difference in operation times with a large number of entries
  - Tested with up to 40 million entries
  - Insert time and query rate independent from catalogue number of entries
- EDG RLS
  - Individual query time increased rapidity up to 100,000 entries
  - Individual insert time for an individual increased beyond 200,000 entries.


LFC


EDG RLS

REQUIREMENTS

If a site acts as a central catalog for several VOs, it can either have:

- One LFC server, with one DB account containing the entries of all the supported VOs. You should then create one directory per VO.
- Several LFC servers, having each a DB account containing the entries for a given VO.

Both scenarios have consequences on the handling of database backups

- Minimum requirements (First scenario)
  - 2Ghz processor with 1GB of memory (not a hard requirement)
  - Dual power supply
  - Mirrored system disk

INSTALLATION

Two installation ways

- **YAIM**
- **Via RPMs (manual)**

References:

https://uimon.cern.ch/twiki/bin/view/LCG/LfcAdminGuide ***(HTML)***

https://edms.cern.ch/file/579088/1/LFC-Administrator-Guide-1.3.5.pdf

(PDF)

INSTALLATION: YAIM

At the moment, YAIM only supports LFC with MySQL backend (no Oracle)

The site-info.def file has to contain the following variables :

$MY_DOMAIN	-> your domain
$LFC_HOST	-> the LFC hostname
$LFC_DB_PASSWORD	-> the MySQL password for the "lfc" user
$MYSQL_PASSWORD	-> the root MySQL password
$LFC_TYPE	-> 'central' or 'local' (default value) depending on LFC installation type

The YAIM node type for LFC server for MySQL is "lcg-LFC-mysql" :

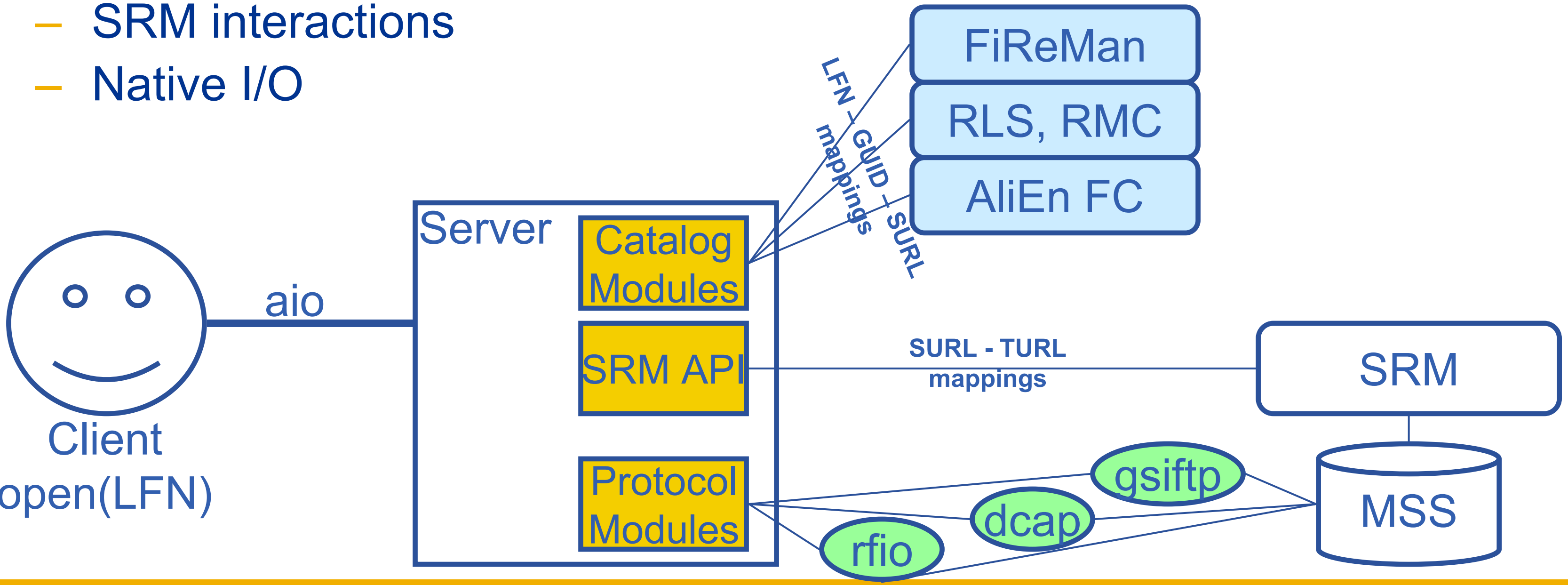
```
/opt/lcg/yaim/scripts/install_node <site-info.def> LFC_mysql
/opt/lcg/yaim/scripts/configure_node <site-info.def> LFC_mysql
```

INSTALLATION: via RPMs

- CERN Grid Deployment group provides RPMs for the LFC RPMs available for Scientific Linux 3.
- RPMs are installed in the /opt/lcg directory (so INSTALL DIR refers to /opt/lcg)
- Not necessary, but useful is installing the LFC client on the server machine
- The current rpms available for SL3 are :
  - lcg-dm-common-1.3.5-1 sl3.i386.rpm
  - LFC-client-1.3.5-1sec sl3.i386.rpm
  - LFC-server-mysql-1.3.5-1sec sl3.i386.rpm
  - LFC-server-oracle-1.3.5-1sec sl3.i386.rpm
- They can be found at http://www.cern.ch/grid-deployment/RpmDir/i386-sl3/external

gLite-I/O

- **Client only sees a simple API library and a Command Line Interface**
  - GUID or LFN can be used, i.e. open("/grid/myFile")
- **GSI Delegation to gLite I/O Server**
- **Server performs all operations on User's behalf**
  - Resolve LFN/GUID into SURL and TURL
- **Operations are pluggable**
  - Catalog interactions
  - SRM interactions
  - Native I/O



gLite I/O commands and API

Summary of the gLite I/O command line tools

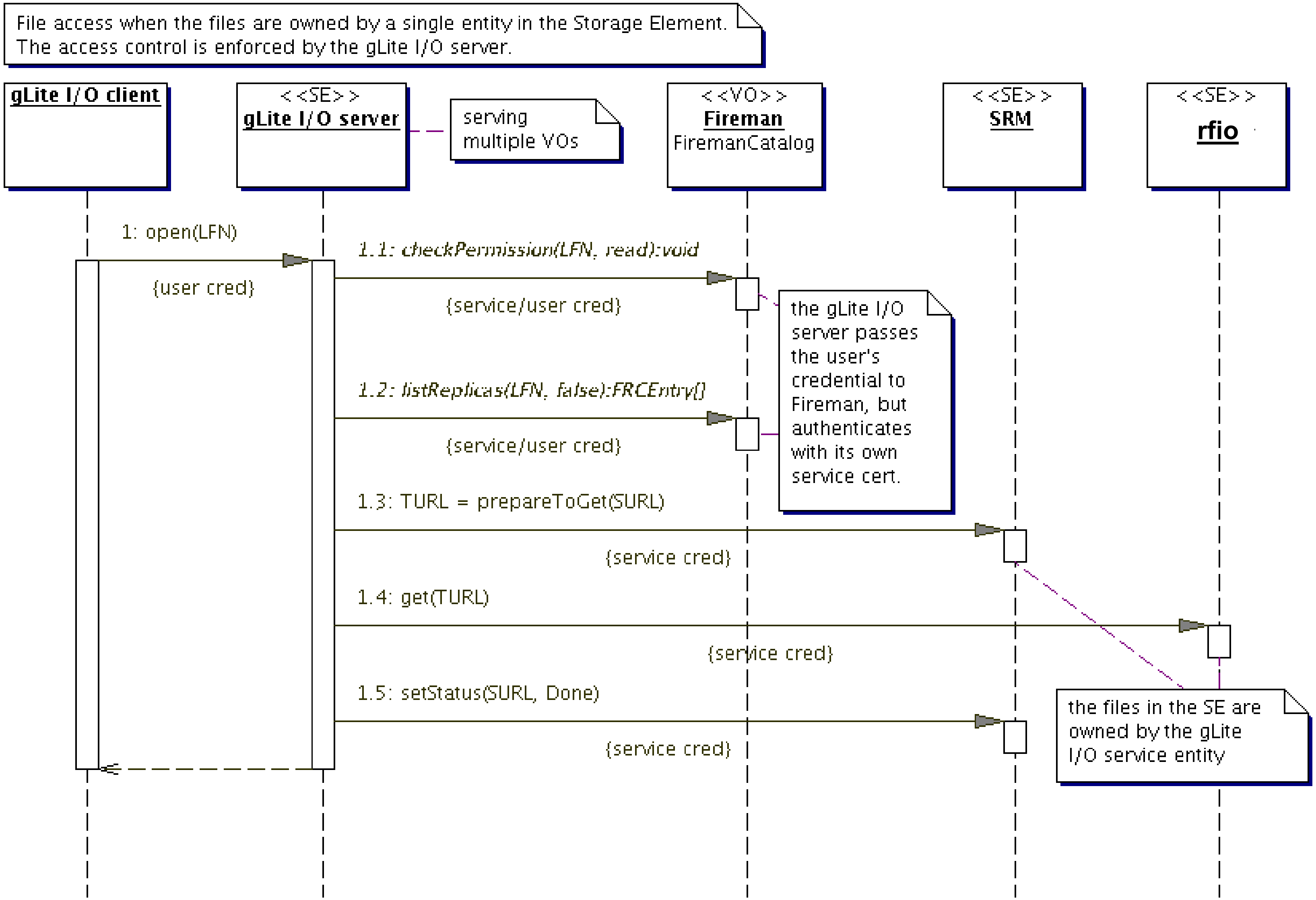
glite-get	Retrieve a file from the Grid using LFN or GUID
glite-put	Put a local file into the Grid, assigning LFN
glite-rm	Remove a file (replica!) from the Grid using LFN or GUID

Summary of the gLite I/O API calls (C only)

glite_open
glite_read
glite_write
glite_creat
glite_fstat
glite_lseek
glite_close
glite_unlink
glite_error
glite_strerror

glite_posix_open
glite_posix_read
glite_posix_write
glite_posix_creat
glite_posix_fstat
glite_posix_lseek
glite_posix_close
glite_posix_unlink
glite_filehandle

File Open


File access when the files are owned by a single entity in the Storage Element.
The access control is enforced by the gLite I/O server.

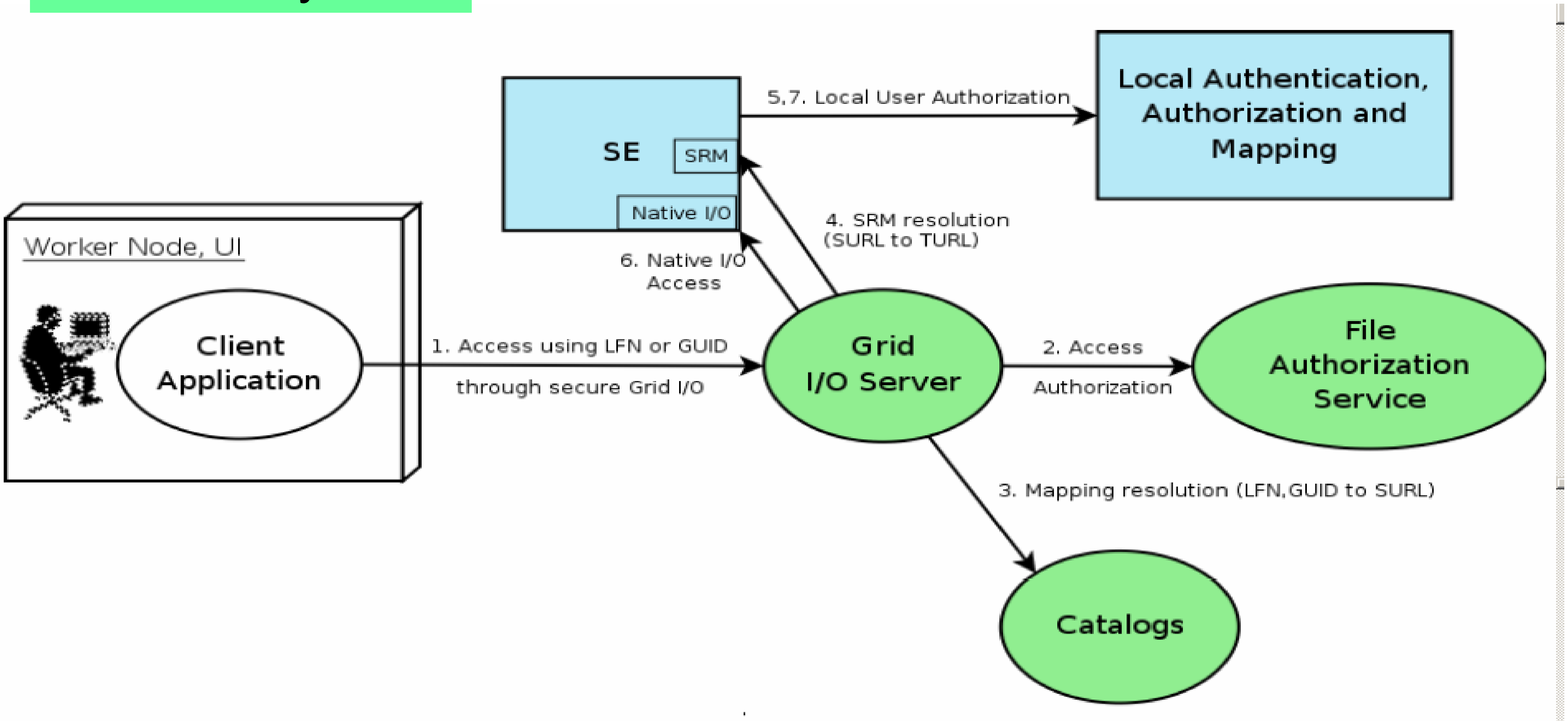
the gLite I/O server passes the user's credential to Fireman, but authenticates with its own service cert.

the files in the SE are owned by the gLite I/O service entity

I/O server interactions

Provided by site

Provided by VO


Local Authentication, Authorization and Mapping
5,7. Local User Authorization
SE
SRM
Native I/O
4. SRM resolution (SURL to TURL)
6. Native I/O Access
Worker Node, UI
Client Application
1. Access using LFN or GUID
through secure Grid I/O
Grid I/O Server
2. Access
Authorization
File Authorization Service
3. Mapping resolution (LFN,GUID to SURL)
Catalogs

Data Movement (I)

- **Many Grid applications will distribute a LOT of data across the Grid sites**
- **Need efficient and easy to manage File movement service**

- **gLite File Transfer Service FTS**
  - Manage the network and the storage at both ends
  - Define the concept of a CHANNEL: a link between two SEs
  - Channels can be managed by the channel administrators, i.e. the people responsible for the network link and storage systems
  - These are potentially different people for different channels
  - Optimize channel bandwidth usage – lots of parameters that can be tuned by the administrator
  - VOs using the channel can apply their own internal policies for queue ordering (i.e. professor's transfer jobs are more important than student's)
- **gLite File Placement Service**
  - It **IS** an FTS with the additional catalog lookup and registration steps, i.e. LFNs and GUIDs can be used to perform replication. Could've been called File Replication Service. (**replica = managed/catalogued copy**)

Data Movement (II)

- **File movement is asynchronous – submit a job**
  - Held in file transfer queue
- **Data scheduler**
  - Single service per VO – can be distributed
  - VO can apply policies (priorities, preferred sites, recovery modes..)
- **Client interfaces:**
  - Browser
  - APIs
  - Web service
- **"File transfer"**
  - Uses SURL
- **"File placement"**
  - Uses LFN or GUID, accesses Catalogues to resolve them

Data movement (II)

- **File movement is asynchronous – submit a job**
  - Held in file transfer queue
- **FPS fetches job transfer requests, contact File Catalogue obtaining** source / destination **SURLs**
- **Task execution is demanded to FTS**
- **User can monitor job status through jobID**
- **FTS maintains state of job transfers**
- **When job is done, FPS updates file entry in the catalogue adding the new replica**

Baseline: GridFTP

- **Data transfer and access protocol for** secure and efficient **data movement**
- **Standardized in the Global Grid Forum**
- extends **the standard** FTP **protocol**
  - – Public-key-based Grid Security Infrastructure (GSI) or Kerberos support (both accessible via GSS-API
  - Third-party control of data transfer
  - Parallel data transfer
  - Striped data transfer Partial file transfer
  - Automatic negotiation of TCP buffer/window sizes
  - Support for reliable and restartable data transfer
  - Integrated instrumentation, for monitoring ongoing transfer performance

Reliable File Transfer

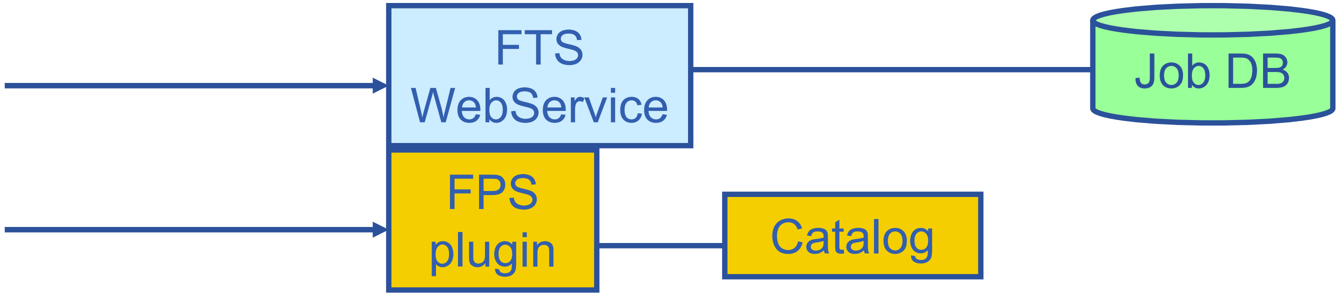
- **GridFTP is the basis of most transfer systems**
- **Retry functionality is limited**
  - Only retries in case of network problems; no possibility to recover from GridFTP a server crash
- **GridFTP handles one transfer at a time**
  - No possibility to do bulk optimization
  - No possibility to schedule parallel transfers

- **Need a layer on top of GridFTP that provides reliable scheduled file transfer**
  - FTS/FPS
  - Globus RFT (layer on top of single gridftp server)
  - Condor Stork

FTS vs FPS

- **File Transfer Service (FTS)**
  - Acts only on SRM SURLs or gsiftp URLs
  - `submit(source-SURL, destination-SURL)`

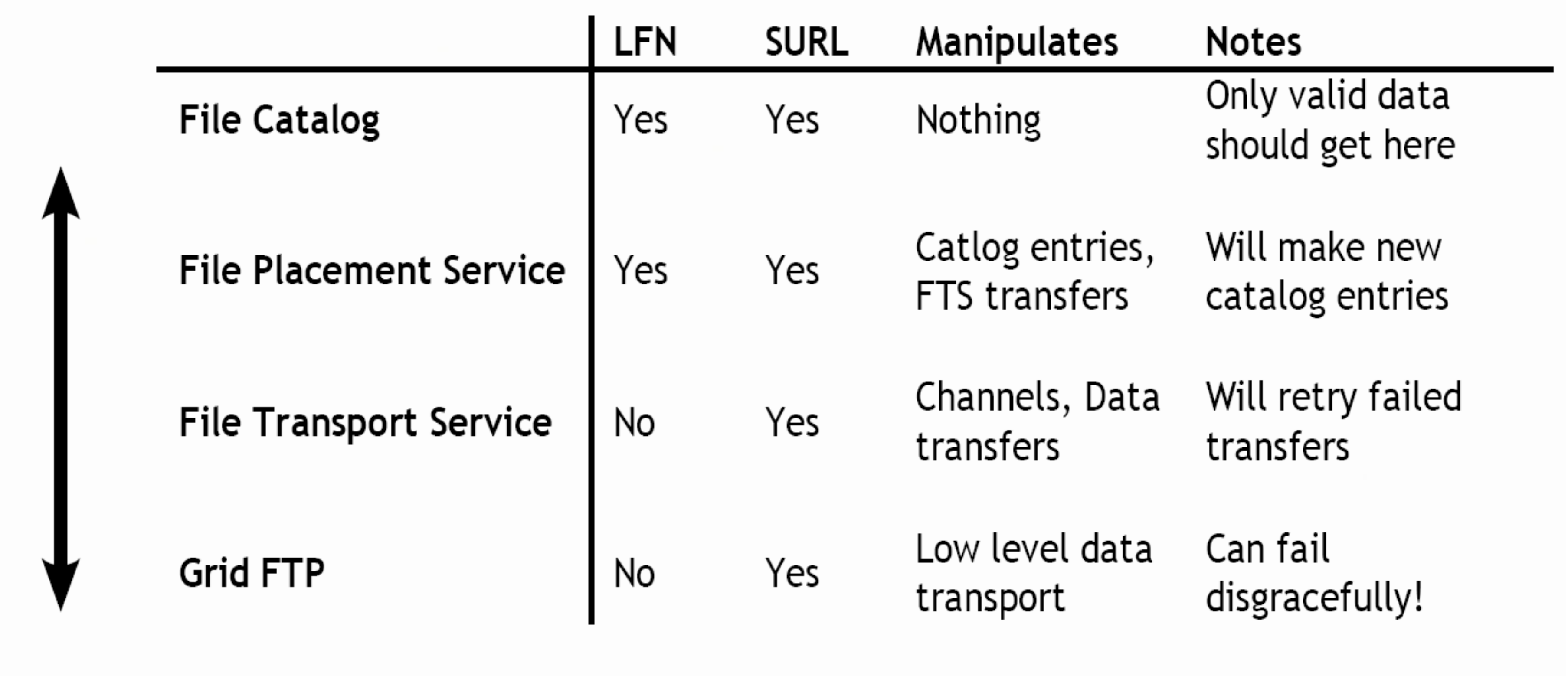
- **File Placement Service (FPS)**
  - A plug-in into the File Transfer that allows to act on logical file names (LFNs)
  - Interacts with replica catalogs (similar to gLite-I/O)
  - Registers replicas in the catalog
  - `submit(transferJobs) (transferJob = sourceLFN, destinationSE)`



FTS vs FPS (II)

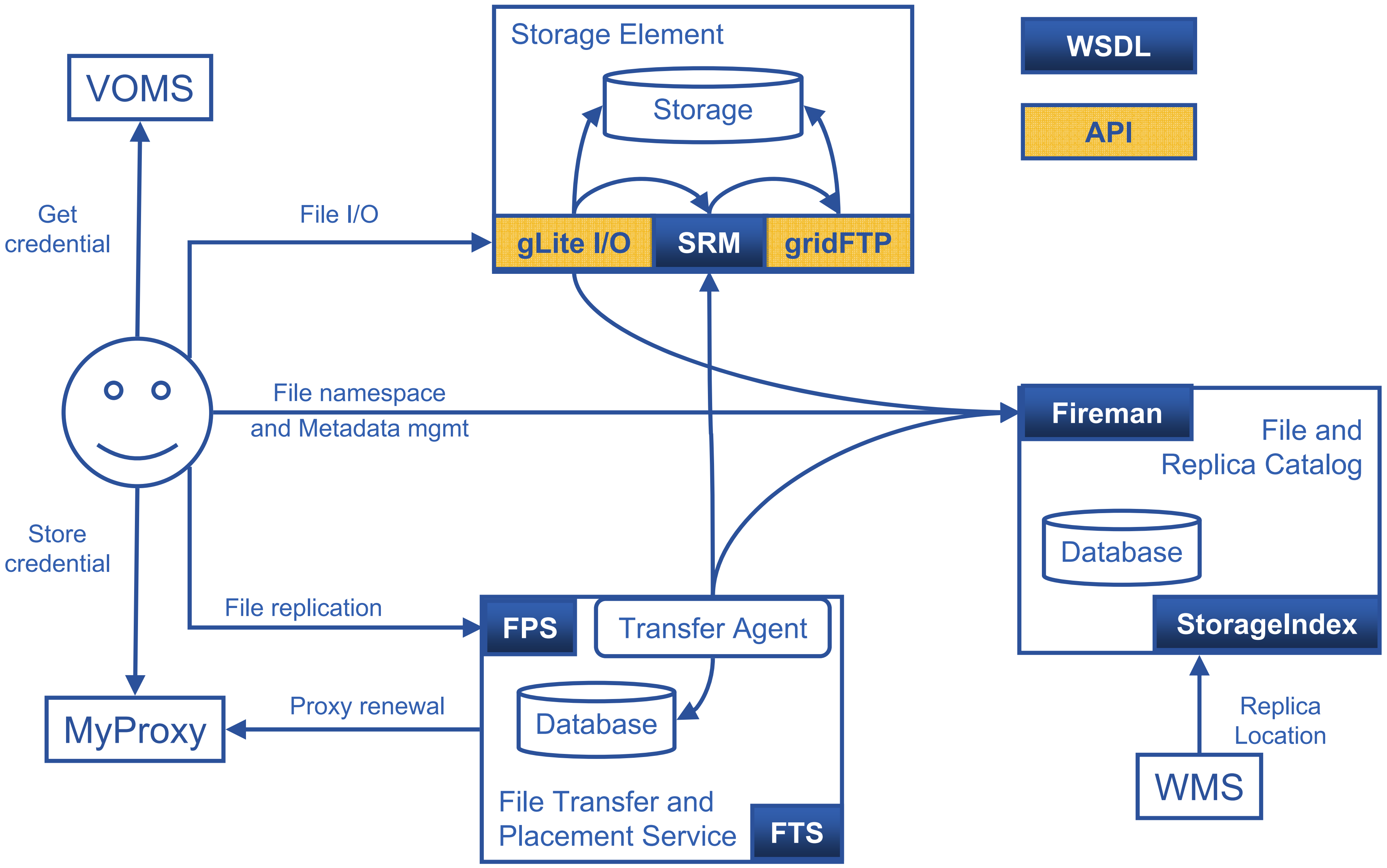
- **Using the File Transfer Service (FTS)**
  - Initiate and monitor transfer
  - Plugin takes care of catalog interactions
- **Using the File Placement Service (FPS)**
  - Lookup source SURL in replica catalog
  - Initiate and monitor transfer
  - After successful transfer register new replica in the catalog
- **FTS and FPS offer the same interface**
  - Difference only in input parameters to the submit command
    - SURLs vs. LFNs
  - Different configuration
    - FPS requires catalog endpoint

Data Movement Stack



	LFN	SURL	Manipulates	Notes
File Catalog	Yes	Yes	Nothing	Only valid data should get here
File Placement Service	Yes	Yes	Catlog entries, FTS transfers	Will make new catalog entries
File Transport Service	No	Yes	Channels, Data transfers	Will retry failed transfers
Grid FTP	No	Yes	Low level data transport	Can fail disgracefully!

DM Interaction Overview



Grid Metadata Services

- **Metadata services on the Grid comes in 2 flavours:**
  - File metadata

Files	
LFN	Production

  - Simple, generalized rel. DB services:

Images		
GUID	Date	Patient

Patient	
ID	Doctor

Doctor	
Name	Hospital

Example from
EGEE-BioMed community

What is AMGA?

AMGA is the Metadata Catalogue for gLite:

- **AMGA started out as ARDA's tool to investigate metadata access on the GRID**
- **AMGA will be officially released in gLite release 1.5**
- **AMGA works in 2 modes:**
  - Side-by-Side a File Catalogue (LFC): File Metadata
  - Standalone: General relational data on Grid
- **AMGA has 2 front ends:**
  - SOAP with PTF standardised interface
  - Text-based TCP streaming protocol (proprietary, documented)
- **AMGA has ideas from many people:**
  UK GridPP Metadata Group, GAG (HEP), gLite DM-team, PTF, LHCb

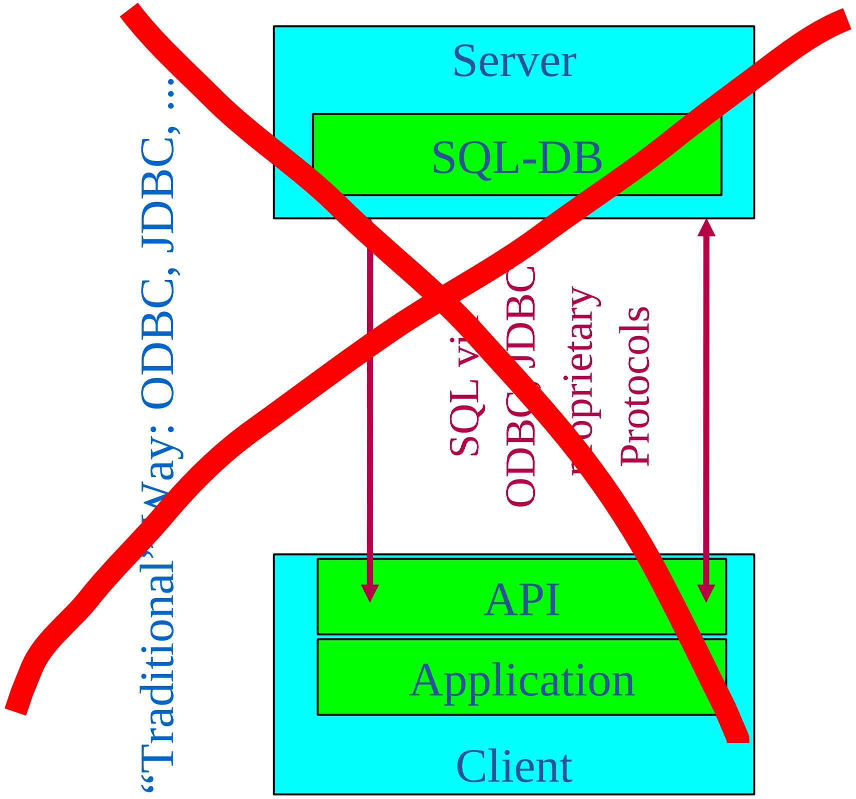
A Common Interface

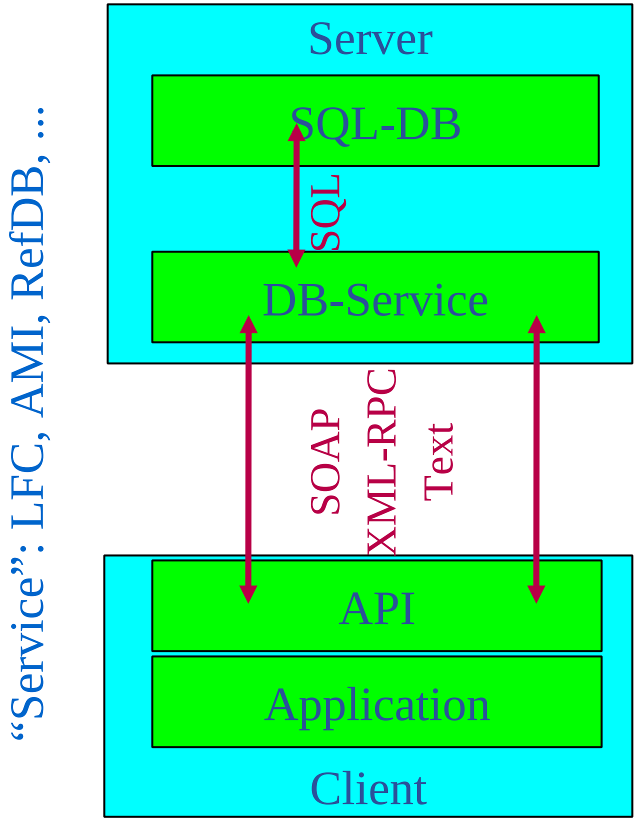
- **AMGA implements a common interface designed in close collaboration of gLite and ARDA teams**

  (P. Kunszt, R. Rocha, N. Santos, B. Koblitz)
- **Again: many ideas from UK GridPP Metadata group, LHCb (Bookkeeping, GANGA), GAG, PTF...**
- **Design Ideas:**
  - Versatility: Usable for HEP as well as Biomed (security)
  - Modular: Interface for Entry manipulation, schemes, security
    - Possible Add-on to File Catalogue
  - Allows stateless & statefull implementations
  - Few requirements on back end, can be SQL-DB, XML...
- **Description of WSDL at https://edms.cern.ch/document/573725**

DB Access on the Grid

- **Traditional DB access doesn't work on Grid:**





+Performance

+Simple Implementation

− Security, Monitoring

− Authentication, resource management??

+Lightweight Client

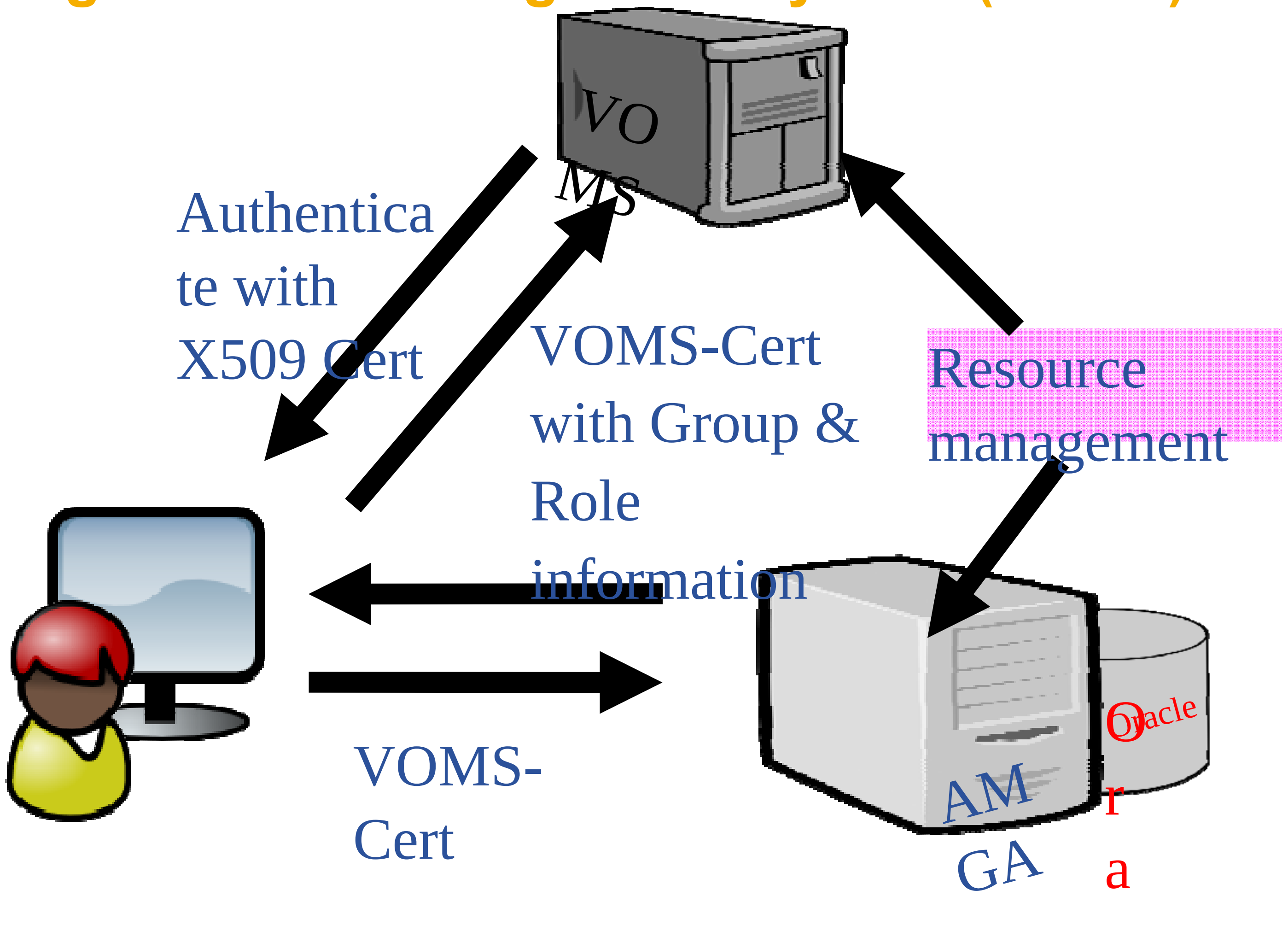
+Security: GSI, x509

− Performance

− Implementation: State

Access Control on the Grid

- Access control to resources on the Grid is done via a Virtual Organization Management System (VOMS):


VO
MS
Authenticate with X509 Cert
VOMS-Cert with Group & Role information
Resource management
VOMS-Cert
AM
GA
Oracle
O
r
a
c

Security Concepts

- **Security very important for BioMed, not for HEP**

  Security ↔ Speed

- **Standalone catalogue has:**
  - ACLs for dirs and Unix permissions dirs/entries
  - Built-in group-management as in AFS
- **AMGA + LFC back end:**
  - Posix ACLs + Unix permissions for dirs/entries (ACLs currently not checked: slow!)
  - Users/groups via VOMS
- **Currently no security on attribute basis**
  - AMGA allows to create views: Safer, faster, similar to RDBMS

Security tested by GILDA team for standalone catalogue, liked built-in group management & ACLs

Basic Concepts

- **Entry**
  - Has key (unique string) and attributes
- **Attribute**
  - Has name (string),
    type (depends on backend, support for basic types)
  - Belongs to schema
  - An entry in a schema has a value for each attribute
- **Schema (in AMGA: directory)**
  - Has name and list of attributes
  - In AMGA: Every entry belongs to one schema, schemas are hierarchical: `/collaboration1/jobs`
- **Query**
  - SELECT ... WHERE ... clause in SQL-like query language

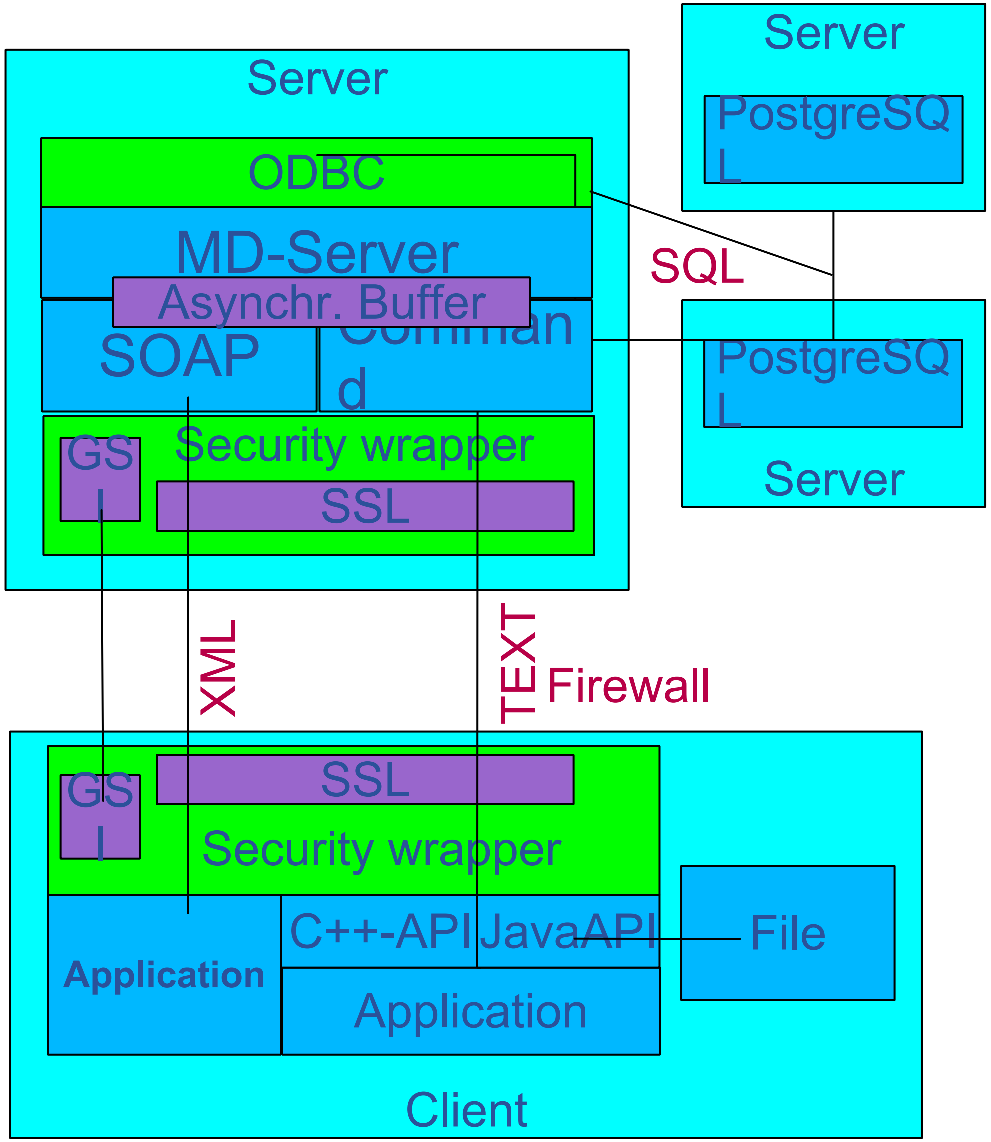
Example command line session:

```
mdclient -p8822 lxb0709
Connected to lxb0709:8822
ARDA Metadata Server 0.9.4
Query> dir /
>> >grid<
>> >collection<
Query> dir /grid/arda
>> >lfn-0.dat<
  [... rest of LFC entries]
Query> addattr /grid/arda i int t text
Query> listattr /grid/arda
>> >i<
>> >int<
>> >t<
>> >text<
```

```
Query> addentries /grid/arda/lfn-0.dat /grid/arda/lfn-1.dat
Query> listentries /grid//arda
>> >lfn-0.dat<
>> >lfn-1.dat<
Query> addentry /grid/arda/lfn-2.dat i 2 t 'A test'
Query> listentries /grid/arda
>> >lfn-0.dat<
>> >lfn-1.dat<
>> >lfn-2.dat<
Query> addattr /grid/arda f float
Query> find /grid/arda/* 'i=2'
>> >lfn-2.dat<
```

AMGA Implementation

- **AMGA Implementation:**
  - SOAP and Text frontends
  - Supports single calls, sessions & connections
  - SSL security with grid certs
  - PostgreSQL, Oracle, MySQL, SQLite backends
  - Works alongside LFC
  - C++, Java, Python clients
- **See & download at** http://project-arda-dev.web.cern.ch/project-arda-dev/metadata/



AMGA in Use

AMGA in preproduction within several projects:

- LHCb and ATLAS: GANGA
- LHCb Logging and Bookkeeping
- EGEE BioMed applications
  - Highly secure access to medical images metadata
- Generic applications:
  - Metadata for EGEE-GILDA Movie-On-Demand application (gMOD)
  - UNOSAT project: Used side-by side with LFC catalogue for file-metadata of satellite images

References

- **gLite homepage**
  - http://www.glite.org
- **DM subsystem documentation**
  - http://egee-jra1-dm.web.cern.ch/egee-jra1-dm/doc.htm
- **FiReMan catalog user guide**
  - https://edms.cern.ch/file/570780/1/EGEE-TECH-570780-v1.0.pdf
- **gLite-I/O user guide**
  - https://edms.cern.ch/file/570771/1.1/EGEE-TECH-570771-v1.1.pdf
- **FTS/FPS user guide**
  - https://edms.cern.ch/file/591792/1/EGEE-TECH-591792-Transfer-CLI-v1.0.pdf
- **AMGA documentation**
  - http://project-arda-dev.web.cern.ch/project-arda-dev/metadata/

Questions…