Distributed File and Metadata Catalogs

Peter Kunszt

Distributed Deployment of Databases, 2004 December 13-15

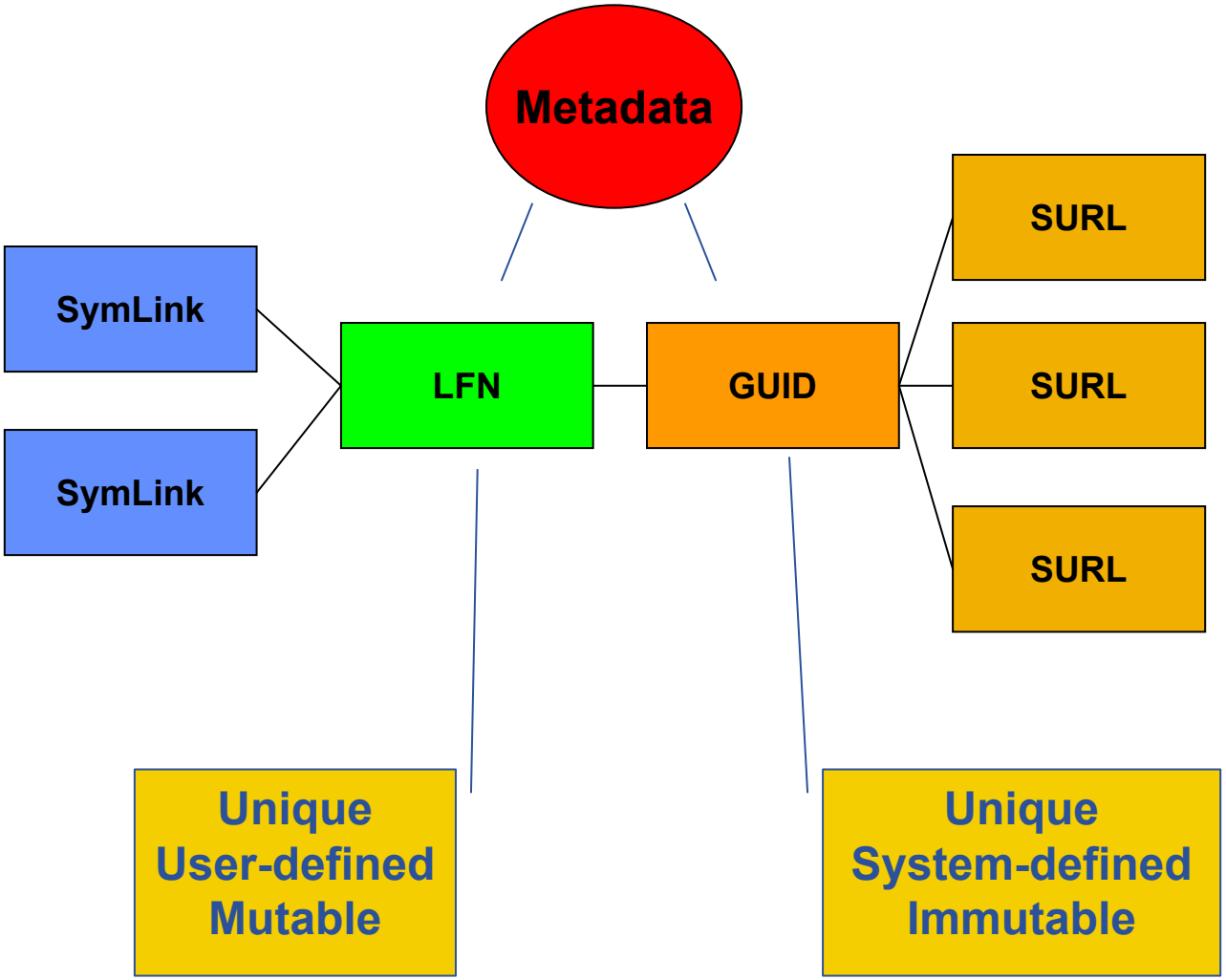
Contents

- **gLite Catalogs**
  - Overview
  - Concepts
  - Implementations
  - Distribution
- **Deployment models**
- **Distribution mechanisms**

gLite Catalogs

- **Taken recommendation from original LCG ARDA RTAG**
- **File and Metadata Catalog semantics largely taken from AliEn**
- **Design considerations**
  - Modularity
  - Flexible deployment models
  - Replaceable by custom implementations

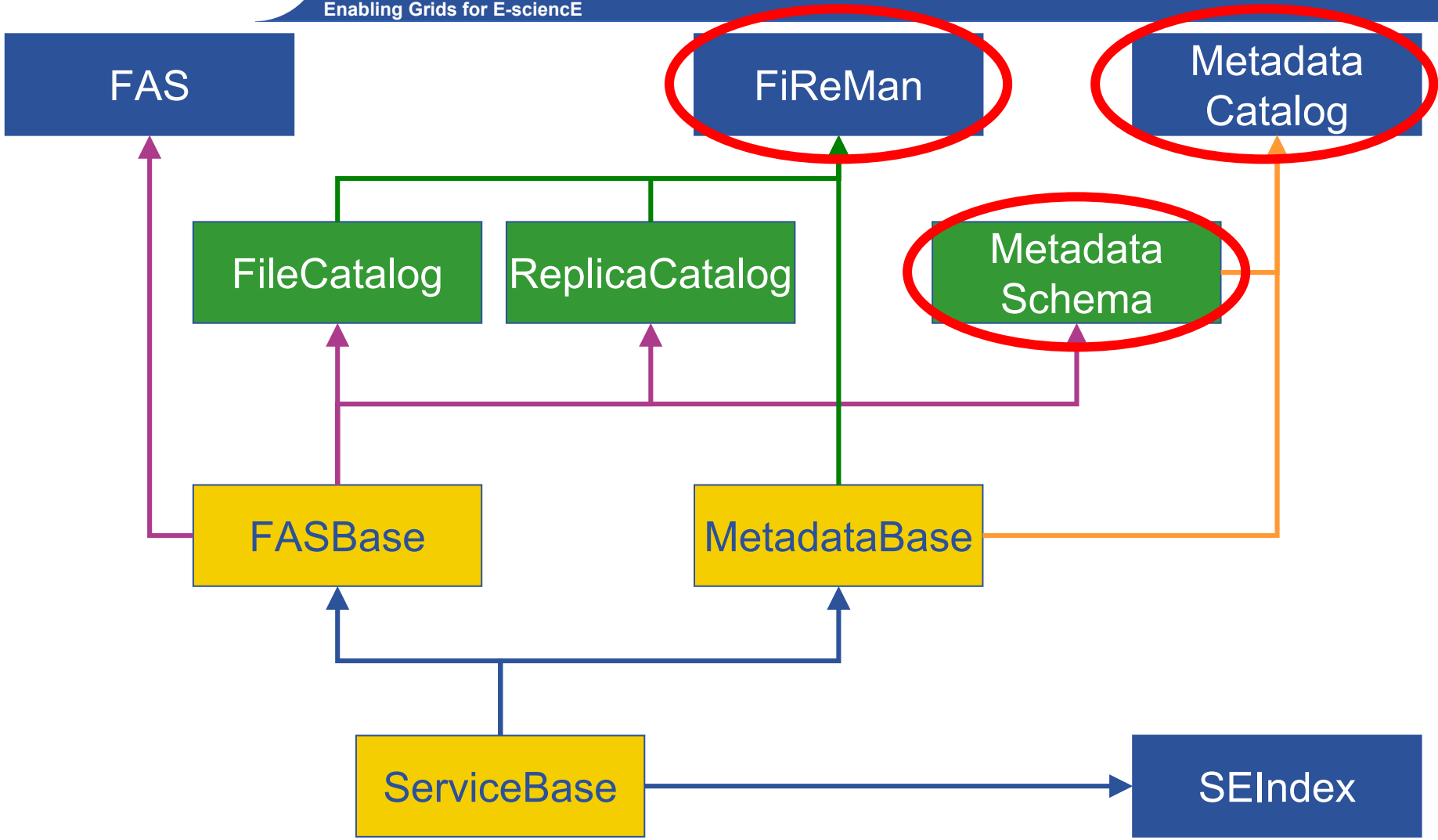
Recap: WHAT is in the Catalogs



Concepts

- Directories
- Symlinks
- Authorization: ACL and base (unix) permissions
- File metadata (size, ctime, mtime, checksum, status, type)
- File-based metadata (key-value pairs on files), the schema is associated per directory
- Extensible metadata including schema manipulation
- *Maybe virtual directories (cached metadata queries) in the future*

Interface Design



Base Interfaces Service Interfaces Feature Interfaces End-user Interface

gLite Catalog Implementations

DONE
In progress or planning

- **Fireman Interface**
  - **Oracle 9i implementation**
  - **MySQL implementation**
- **MetadataCatalog Interface**
  - **MySQL implementation**
  - *Oracle 9i implementation*
- **MetadataSchema Interface**
  - *MySQL implementation*
  - *Oracle 9i implementation*
- **Apply interfaces to existing implementations**
  - ***Will have a Fireman interface also over the AliEn FC***
  - *Fireman interface over the LCG FC*
  - *MetadataCatalog and MetadataSchema over existing application catalogs*
  - *…*

gLite Distribution Mechanisms 1

- **Data Scheduler (global and local schedulers)**
  - Global scheduler (VO-specific) takes requests like
    - Copy set of files from A to B
    - Make set of files available at C
    - Upload files from GSIFTP server to D
    - Delete files
    - *Maybe also metadata operations*
  - Local scheduler fetches tasks from known global schedulers
    - Coupled tightly to a local transfer service
    - Manage transfer where the local site is a target
    - Assure atomicity of transfer and catalog operations
- **Transfer Service**
  - Queue data transfers to/from a given Storage Element (SRM)
  - Receives jobs from local scheduler
  - Manages transfers through a set of states

Catalog Deployment Models

- **Single central catalog (AliEn, LCG-2 model)**
  - All operations go there
- **Local catalogs with a central component**
  - Update operation only on local catalogs
  - Update operation on both local and central catalogs
- **Local catalogs, no central component – only indices for certain queries**

Catalog Distribution Mechanism

- **Certainly possible to just rely on DDD**
- **Middleware distribution of updates between catalogs**
  - Using a messaging system (JMS using JORAM)
  - Publish updates to message queue locally
  - Subscribe to updates at central catalogs / index nodes
  - Asynchronous messaging queues take care of update delivery
  - Scales well to the number of sites we deal with
  - However, error messages have to be queued for retrieval as well

To be understood

- **What to distribute and how**
  - All of the data? (Replication)
  - Just parts? (Indexing)
  - Read-write mechanisms and updates between many copies (Policies)
- **Metadata usage**
  - 2 distinct metadata capabilities:
    - Keyed on GUID in the File Catalog using hierarchy
    - Generic Metadata – in order to link with files, also needs GUID or LFN relation
  - Schema manipulation capabilities – what is really needed
  - Metadata services by experiments may interface with gLite or implement the gLite interfaces themselves
    - Are a set of canned queries good enough? If yes, user does not need to have a generic query interface.
    - Does all of the metadata need to be local? Or will some metadata have to be fetched from remote sites?
    - What kinds of distributed queries are necessary at all?
    - What kind of metadata is for local/laptop usage?
    - What kinds of update semantics are needed if at all? (Single instance, single master, multi master)

Matching with DDD

- **Low-level DB replication and middleware-replication are complementary approaches**
- **Can be exploited through different deployment scenarios, for example**
  - DB replication between high-performance Tier-1s
  - Messaging to the many Tier-2,3 sites
- **Or the other way round**

Summary

- gLite provides middleware-level data and catalog distribution
- Can be set up as complementary to DDD
- Actual usage of application metadata needs to be understood

We are looking forward to work with the community to address these issues.