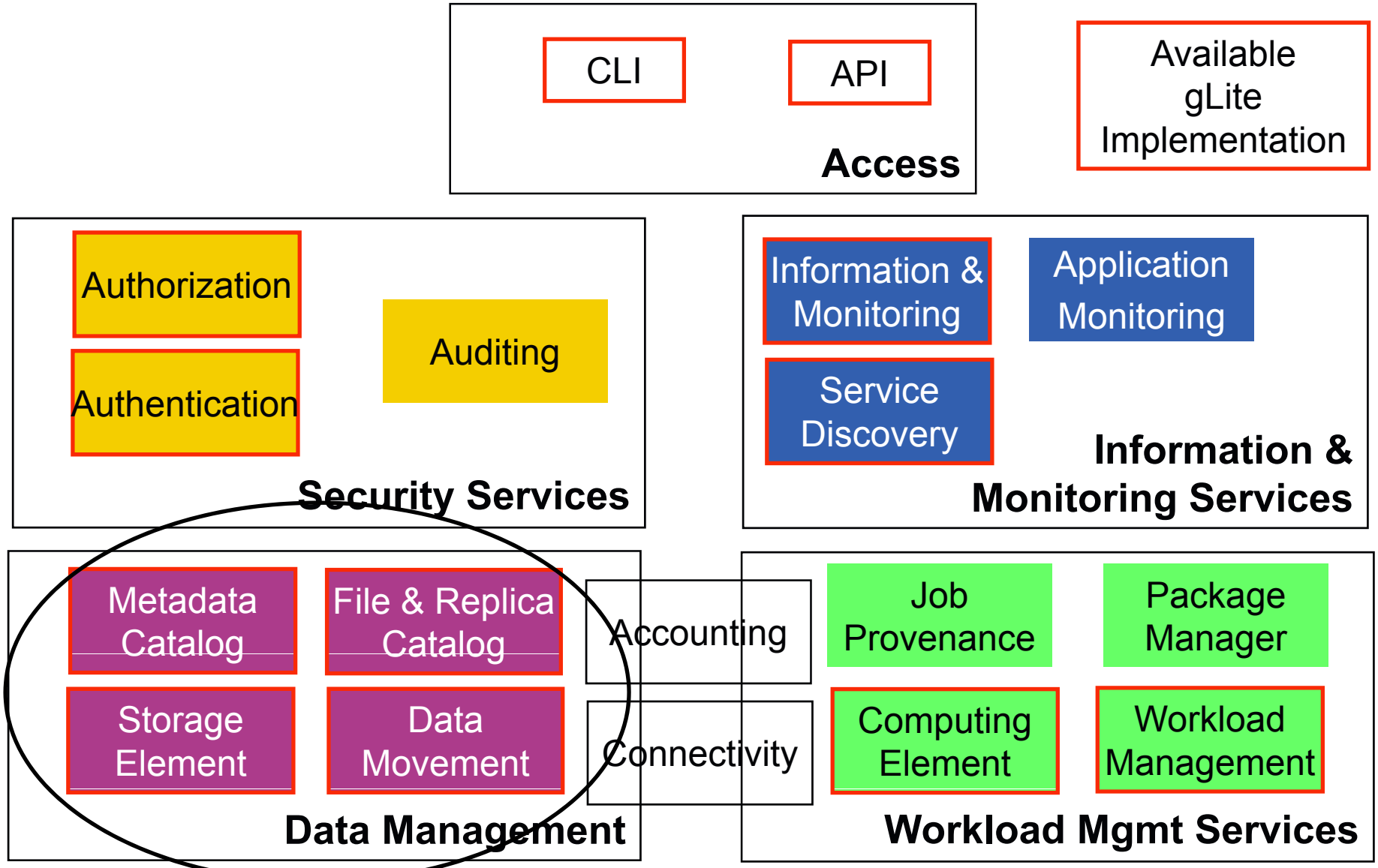
Система управления данными в gLite

Олешко С.Б.
Петербургский институт ядерной физики
г.Гатчина

Сервисы gLite



Введение

- **Предпосылки:**
  - пользователи и программы являются источником и потребителем данных
  - основным экземпляром данных принят файл (мы работаем с файлами, а не с объектами или реляционными таблицами
    - данные = файлы
- **Файлы:**
  - в основном записываются один раз, читаются многократно
  - размещены на Элементах Хранения - Storage Elements (SEs)
  - могут существовать несколько реплик одного файла на различных сайтах
  - доступны для пользователей Грид “отовсюду”
  - местоположение м.б. определено WMS (data requirements в JDL)
- **Также…**
  - WMS может пересылать небольшой объём данных с заданием или от выполненного задания: Input and Output Sandbox
  - файлы могут копироваться с локальной файловой системы (WNs, UIs) в Грид (SEs), и наоборот

Требования к gLite SE

- **Storage Element - это сервис, который позволяет пользователю или приложению сохранять данные для будущего использование**
- **Управление локальными ресурсами памяти (диски) и интерфейс к Mass Storage Systems (ленты), таким как**
  - HPSS, CASTOR, DiskeXtender (UNITREE), …
- **Способность управлять различными системами хранения данных единым способом и прозрачно для пользователя (обеспечивается через SRM интерфейс)**
- **Поддержка основных протоколов передачи данных**
  - GridFTP обязательно
  - Другие по возможности (https, ftp, etc…)
- **Поддержка “привычного” протокола доступа для ввода/вывода удалённых файлов**
  - POSIX (like) I/O client library for direct access of data (GFAL)

Пример из жизни

Она запускает задачу, которой нужны:

- данные реконструкции физического события
- данные симуляции
- некоторые файлы с данными анализа

Результаты также должны быть где-то сохранены

В CERN
на dCache

В Nikhef
на classic SE

В Fermilab
на дисковом массиве

Пример из жизни

dCache
Собственная система, свой протокол и параметры

gLite DPM
Система, независимая ни от dCache ни от Castor

Castor
Нет связи с dCache или classic SE

SRM

Я общаюсь с ними от вашего имени
Я буду выделять место для ваших файлов
И я буду использовать протоколы передачи данных, чтобы пересылать ваши файлы туда

Storage Resource Management

- Данные хранятся на disk pool servers или Mass Storage Systems
- Управление этими ресурсами должно обеспечивать:
  - Прозрачный доступ к файлам (migration to/from disk pool)
  - Выделение места для файлов (Space reservation)
  - Получение информации о статусе файлов (File status notification)
  - Управление временем жизни файлов (Life time management)
- SRM (Storage Resource Manager) сервис реализует все эти требования:
  - SRM это Грид сервис, который реализует взаимодействие с локальными ресурсами хранения данных и обеспечивает Грид-интерфейс для внешнего мира
  - SRM – это протокол управления ресурсами хранения данных, а не протокол доступа к файлам или протокол передачи файлов.
- SRM разработан, чтобы служить единым интерфейсом для управления дисковыми (или ленточными) ресурсами.
- В gLite взаимодействие с SRM обычно скрыто за сервисами более высокого уровня (DM tools и APIs)

Поддерживаемые протоколы

Протоколы доступа к файлам в gLite SE 3.0:

Протокол	Тип	GSI	Описание
GSIFTP(GridFTP)	Передача файлов	Да	Аналог FTP
gsidcap (GSI dCache Access Protocol)	Ввод/вывод	Да	Удалённый доступ
insecure RFIO (Remote File Input/Output Protocol)	Ввод/вывод	Нет	Удалённый доступ
secured RFIO (gsirfio)	Ввод/вывод	Да	Удалённый доступ

* Протокол file сейчас используется только для доступа к файлам на локальном компьютере (т.е. на UI или WN), но не к файлам на Грид SE

** GridFTP сейчас является обязательным для каждого из типов SE, поддерживаемых в gLite и основным для передачи файлов в Грид.

Типы SE в gLite (I)

- **Classic SE:**
  - GridFTP сервер
  - Insecure RFIO daemon (rfiod) – ограниченный доступ для LAN
  - Только одно дисковое устройство или дисковый массив
  - Нет возможности управлять квотами (только partitioning)
  - Не поддерживает SRM интерфейс
- **Mass Storage Systems**
  - Комплексная иерархическая система хранения: front-end диски и back-end ленты
  - GridFTP для front-end (file transfere)
  - File access: insecure RFIO (CASTOR), gsidcap для dCache (front-end disk pool)
  - Поддерживает SRM интерфейс (пока только для Castor)

Типы SE в gLite (II)

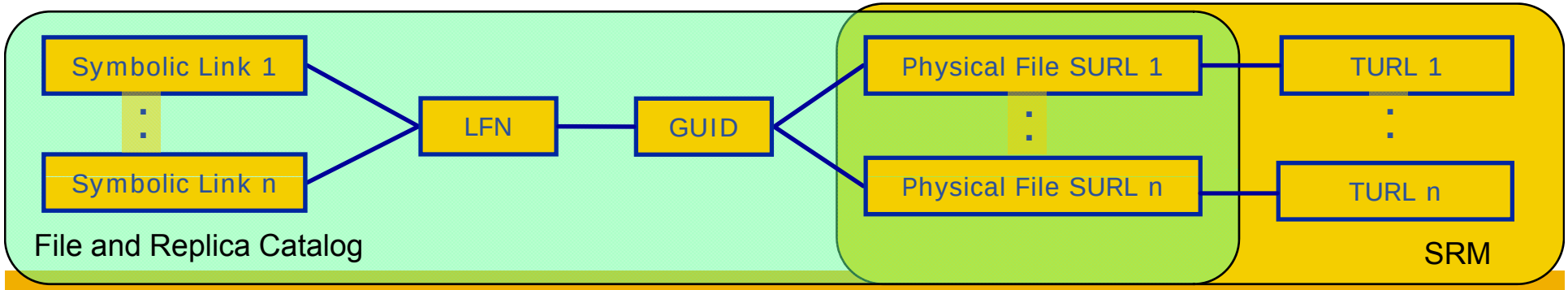
- **Disk pool managers (dCache and gLite DPM)**
  - Обеспечивает централизованное управление распределёнными серверами хранения данных
  - Физические диски и массивы объединены в общую (виртуальную) иерархическую файловую систему с единой точкой входа в SE
  - Диски могут быть динамически добавлены в пул
  - GridFTP сервер
  - Secure remote access protocols (gsidcap for dCache, gsirfio for DPM)
  - SRM интерфейс

Именование файлов

- **Symbolic Link в пространстве логических имён (**logical filename space)
- **Logical File Name (LFN)** [lfn:<anything_you_want>]
  - Имя, созданное пользователем для того чтобы ссылаться на некоторый элемент данных, напр. “lfn:cms/20030203/run2/track1”
- **Globally Unique Identifier (GUID)** [guid:<40_bytes_unique_string>]
  - Внутренний (машинный) идентификатор элемента данных, напр. “guid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6”
- **Site URL (SURL)** [<sfn | srm>://<SE_hostname>/<some_string>]

(or Physical File Name (PFN) or Site FN)

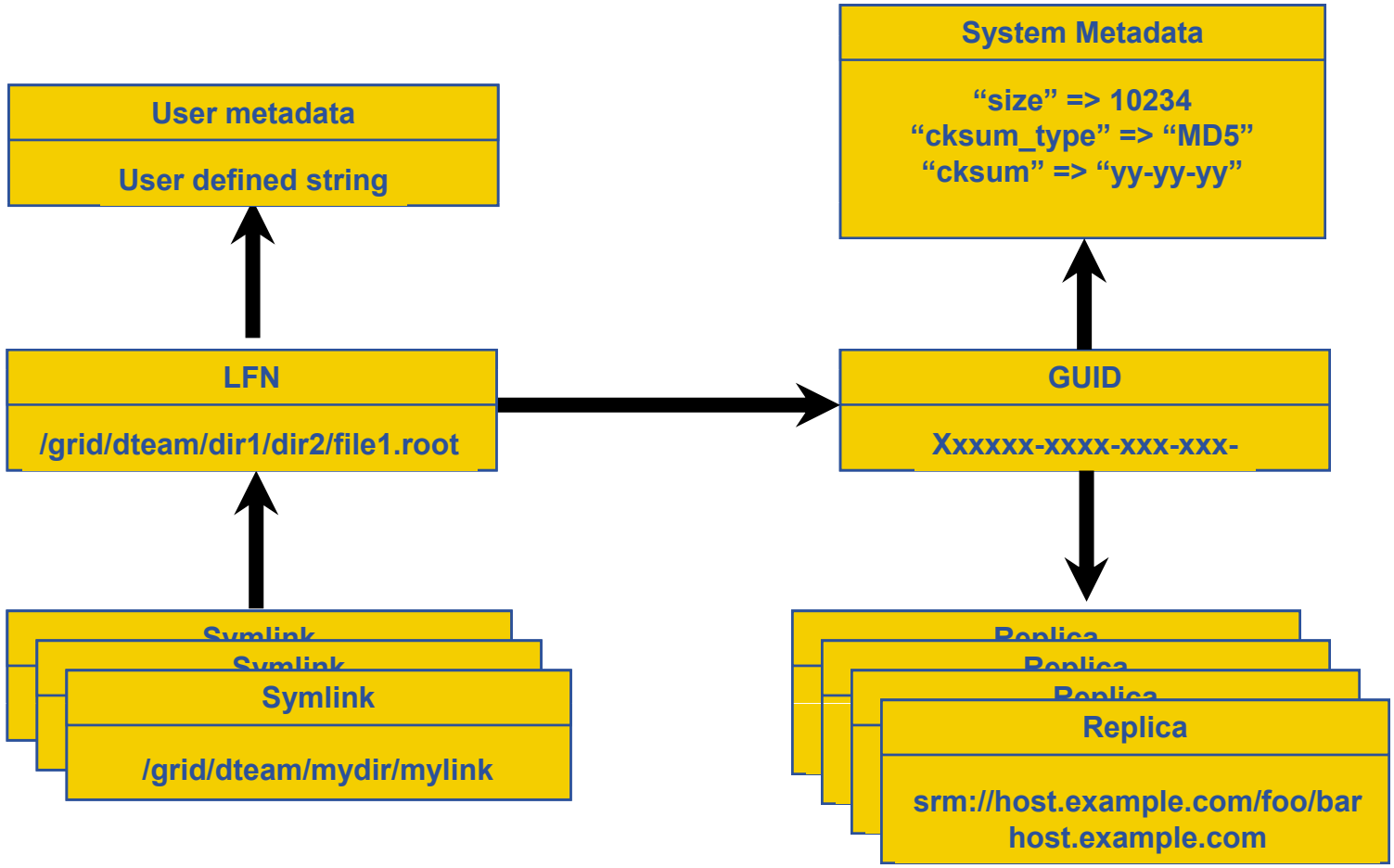
  - Физическое местоположение реплики элемента данных в системе хранения данных, напр. “srm://pcrd24.cern.ch/flatfiles/cms/output10_1” (SRM)
    “sfn://lxshare0209.cern.ch/data/alice/ntuples.dat” (Classic SE)
- **Transport URL (TURL)** [<protocol>://<some_string>]
  - Временный указатель на реплику + протокол доступа: распознаётся SE, напр. “rfio://lxshare0209.cern.ch//data/alice/ntuples.dat”



Каталоги (File and Replica Catalog)

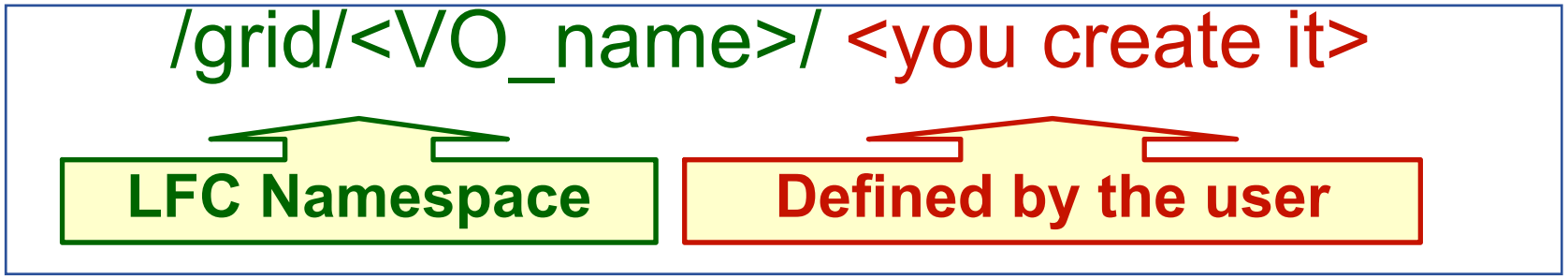
- **Главная цель - определить, где размещены файлы в Grid**
- **File and Replica Catalog - это сервис, который реализует это и поддерживает соответствие между LFNs, GUIDs и SURLs.**
- **В gLite поддерживаются 2 типа каталогов:**
  - Replica Location Server (RLS) - старый
    - Local Replica Catalog (LRC)
    - Replica Metadata Catalog (RMC)
  - LCG File Catalog (LFC) – по умолчанию
- **Тип используемого пользователем каталога определяется переменной окружения LCG_CATALOG_TYPE: edg для RLS, lfc для LFC**
- **Оба каталога между собой несовместимы!!! Однако есть средства миграции из RLS в LFC**
- **Файл данных только тогда может считаться Грид-файлом, когда он физически присутствует на каком-либо SE и зарегистрирован в каталоге**

Взаимосвязи в LFC



Структура LFC

LFC имеет иерархическую структуру



- Все члены данной ВО имеют права чтения/записи в соответствующую директорию
- Если соответствующей директории нет, то это означает, что данный LFC сервер не поддерживает эту ВО
- Команды работы с LFC похожи на соответствующие команды в UNIX (с префиксом *lfc-*)
- Переменная окружения $LFC_HOST должна содержать имя LFC сервера

Команды LFC

lfc-chmod	Изменить права доступа к файлу/директории LFC
lfc-chown	Изменить владельца и группу для файла/директории LFC
lfc-delcomment	Удалить комментарии, связанные с файлом/директорией
lfc-getacl	Показать ACL для файла/директории
lfc-ln	Создать символическую ссылку на файл/директорию
lfc-ls	Вывести список файлов в директории
lfc-mkdir	Создать директорию
lfc-rename	Переименовать файл/директорию
lfc-rm	Удалить файл/директорию
lfc-setacl	Установить/изменить ACL для файла/директории
lfc-setcomment	Добавить/заменить комментарий

Средства работы с данными

- LCG Data Management tools (обычно называемые *lcg-utils*) позволяют копировать файлы между UI, CE, WN и SE, регистрировать в File Catalogs и реплицировать данные между SEs.
- Поскольку *lcg-utils* используют ИС, то должна быть правильно установлена переменная окружения LCG_GFAL_INFOSYS, которая указывает на BDII сервер
- Почти все команды требуют обязательного параметра (если не установлена переменная LCG_GFAL_VO)

--vo <vo_name>

lcg-utils

Replica Management

lcg-cp	Копировать файл из Грид на локальный компьютер (UI)
lcg-cr	Копировать файл на SE и зарегистрировать его в каталоге
lcg-del	Удалить один файл (или реплику, или все реплики)
lcg-rep	Репликация между SE и регистрация реплики
lcg-gt	Получить TURL для данных SURL и протокола передачи
lcg-sd	Установить статус “Done” для данного SURL в SRM запросе

File Catalog Interaction

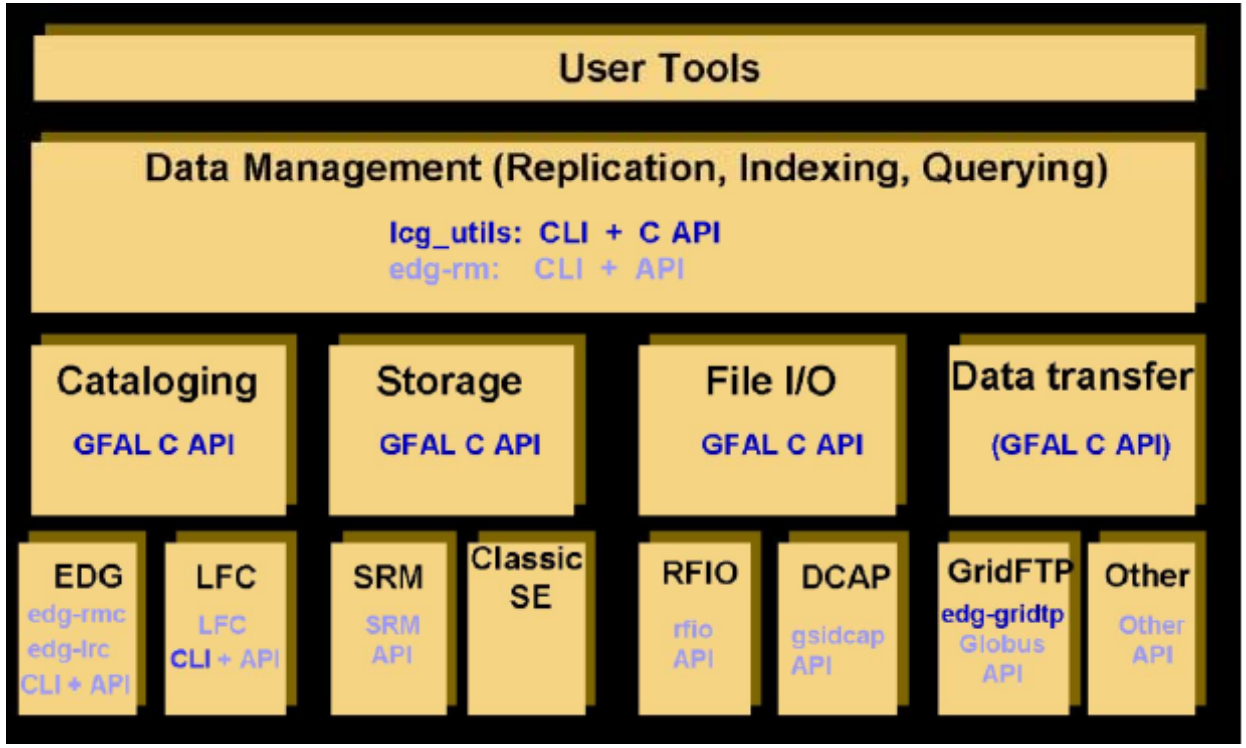
lcg-aa	Добавить синоним в LFC для данного GUID
lcg-ra	Удалить синоним в LFC для данного GUID
lcg-rf	Зарегистрировать в LFC файл, размещённый на SE
lcg-uf	Удалить регистрацию в LFC файла, размещённого на SE
lcg-la	Список всех синонимов для данного SURL, GUID или LFN
lcg-lg	Получить GUID для данного LFN или SURL
lcg-lr	Список всех реплик для данного GUID, SURL или LFN

gLite I/O API

Свойства GFAL API

- GFAL (Grid File Access Library) – это API, имеющий POSIX интерфейс для операций с файлами, расположенными на SE
- Позволяет удалённую работу с файлами (особенно полезно, если нужен доступ к части очень большого файла)
- Библиотеки на C и могут быть включены в программы на C/C++
- Есть Java API
- SE должен поддерживать протокол secure rfio (поэтому для *classic SEs* использовать нельзя)
- Скрывает взаимодействие с SRM для пользователя

Data Management API



GFAL: переменные окружения

- **Для некоторых функций GFAL необходимо взаимодействие с каталогом, а он BO-зависим, поэтому должны быть установлены следующие переменные окружения:**
  - LCG_GFAL_VO
  - LCG_GFAL_INFOSYS
  - LCG_CATALOG_TYPE
  - LFC_HOST
- **Кроме того:**
  - LCG_RFIO_TYPE
  - LD_LIBRARY_PATH

GFAL: доступные API

- **C API**
  - Файл заголовков gfal_api.h.
  - Вызов функции – добавление префикса **gfal_** к POSIX имени функции (open(), read()…), например gfal_open, gfal_read,...
  - Список аргументов и возвращаемые значения – идентичны POSIX.
  - Переменная errno устанавливается в соответствии с **Posix Error Codes** в случае ошибки.
- **Java API (C API Wrapper)**
  - Поддерживает три основных Java Objects, которые должны быть импортированы в Java-программу.
    - GFalFile : обработка и чтение/запись в файлы
    - GFalDirectory : обработка и управление директориями (создание, удаление, список)
    - GFalUtilities : управление файлами (переименование, удаление, свойства)

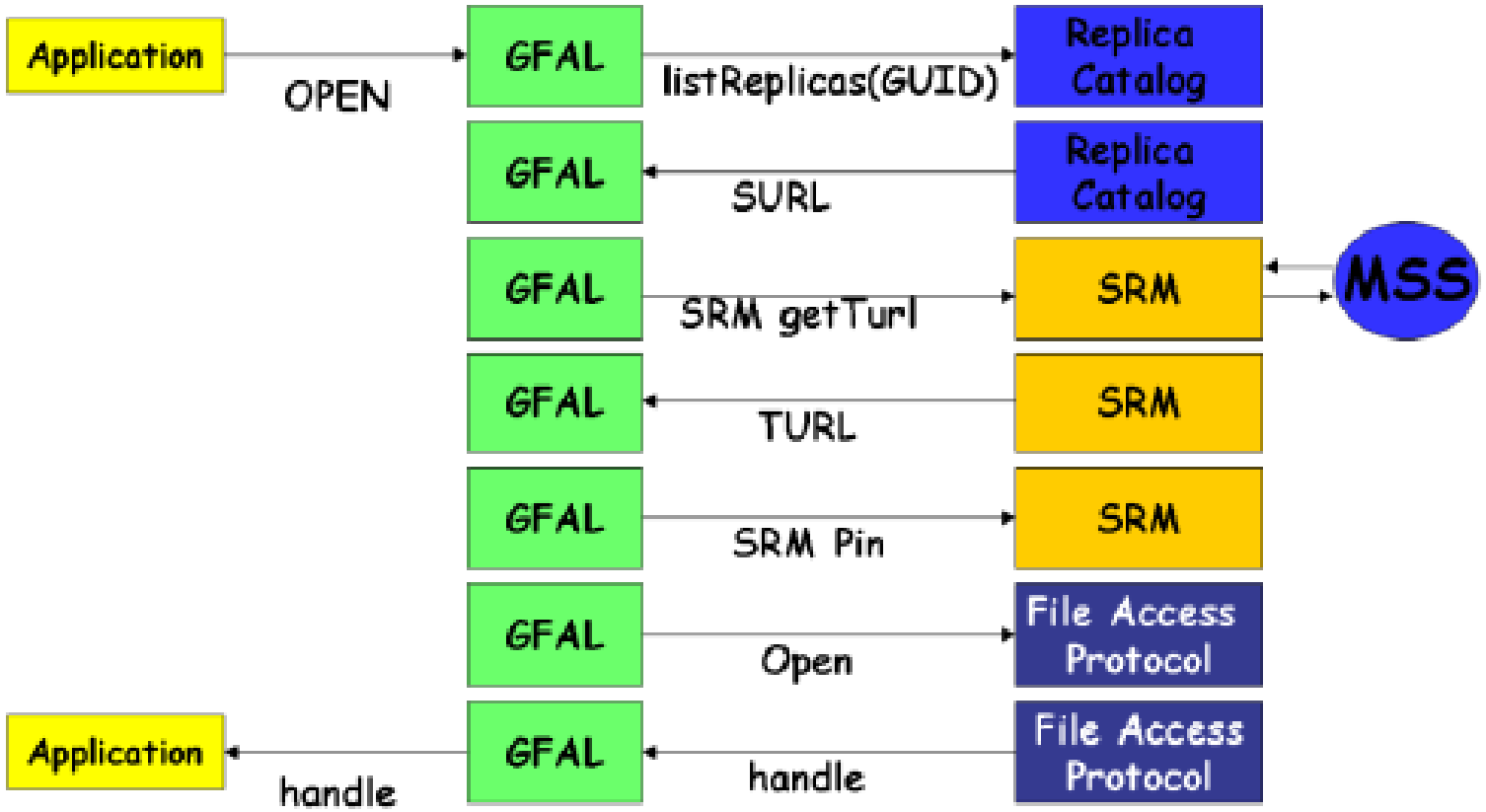
GFAL: функции I/O API (1)

```
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: функции I/O API (2)

```
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);
```

Пример выполнения gfal_open



Ссылки

- **Examples in gLite3 User Guide (Appendix F)**
  - https://edms.cern.ch/file/722398//gLite-3-UserGuide.pdf
- **GFAL C API Description:**
  - http://grid-deployment.web.cern.ch/grid-deployment/documentation/LFC_DPM/gfal/html/
- **GFAL JAVA API**
  - https://grid.ct.infn.it/twiki/bin/view/GILDA/APIGFAL
- **GFAL Java API code and libraries:**
  - https://grid.ct.infn.it/twiki/pub/GILDA/APIGFAL/GFAL_Java_API.zip
- **On-line JavaDoc of Java API:**
  - https://grid.ct.infn.it/twiki/GFAL/
- **GFAL Excercises (C/Java):**
  - https://grid.ct.infn.it/twiki/bin/view/GILDA/UsingGFAL

JDL атрибуты для работы с данными

Sandboxes

- **InputSandbox – файл (список файлов) на *локальном диске UI*, которые будут переданы через узел WMS на WN при запуске задания. Все имена файлов должны быть различны (даже если они в разных директориях).**
- **OutputSandbox – файл (список файлов), которые в результате выполнения задания создаются на узле WMS и могут быть переданы на UI при помощи команды glite-wms-job-output.**

Эти файлы не могут быть на SE, т.е. нельзя использовать LFN(Logical File Name). Существует ограничение на размеры файлов для Sandboxes, т.е. файлы должны быть небольшого размера (ориентировочно < 100Mb).

InputData(deprecated)

- **InputData – строка (список строк), представляющие в одном из допустимых форматов (LFN, GUID, ..) имена входных файлов. Они используются WMS только для получения PFN (Physical File Name), по которым затем WMS на этапе *matchmaking* сможет определить CE, имеющий максимальное количество физических файлов (реплик) на ближайшем SE (CloseSE). В зависимости от префикса имени файла будет выбираться тип каталога для определения PFN (RLS, StorageIndex, DLI). По умолчанию для lfn: и guid: используется RLS.**
- **StorageIndex – URL сервиса gLite Storage Index. Если указан, то для определения PFN файлов с lfn: и guid будет использоваться этот каталог.**
- **DataCatalog - URL сервиса LCG Data Location Interface. Если указан, то для определения PFN файлов с lfn: и guid будет использоваться этот каталог.**

InputData (пример)

```
InputData = {
    "lfn:/mydata/file1",
    "lfn:/mydata/file2",
    "guid:135b7b23-4a6a-11d7-87e7-9d101f8c8b70"
};
// Do not need to specify this attribute if you want to use the VO
// default StorageIndex catalog
StorageIndex =
    "http://lxb1434.cern.ch:8080/EGEE/glite-data-/FiremanCatalog";
```

DataRequirements

DataRequirements - более гибкая форма задания атрибутов для требований на входные файлы. Состоит из групп, в каждой из которых могут быть указаны 3 атрибута:

- **InputData - строка (список строк), представляющие в одном из допустимых форматов (LFN, GUID, ..) имена входных файлов.**
- **DataCatalogType – тип каталога, который будет использоваться для данной группы**
  - RLS - LCG Replica Location Service
  - SI – gLite Storage Index
  - DLI - LCG Data Location Interface
- **DataCatalog - URL сервиса каталога (может определятся, если он отличается от каталога по умолчанию для BO)**

DataRequirements (пример)

```
DataRequirements = {
      [
        DataCatalogType = "DLI";
        DataCatalog = "https://cms.org:8877/dli";
        InputData = {"lfn:/my/test.data1",
                    "guid:44rr44rr77hh77kkaa3”};
      ],
      [
        DataCatalogType = "SI";
        DataCatalog = "https://glite.org:9443/StorageIndex";
        InputData = {"lfn:/eo/test.file", "guid:ddffrg5451"};
      ],
       [
        DataCatalogType = "RLS";
        DataCatalog = "https://eu-datagrid.org/RLS";
        InputData = {"lfn:/atlas/test.file", "guid:ggrgrg5656"};
      ],
                [
        DataCatalogType = "RLS";
        InputData = {"lfn:/myvo/test.file","guid:adbdefgilm1234"};
      ],
      ....
   };
```

DataAccessProtocol

Если определён атрибут InputData либо DataRequirements, то должен быть указан атрибут DataAccessProtocol, который определяет список имён протоколов, которые приложение может использовать для доступа к файлам.

```
DataAccessProtocol = {
        “file”,
        “gridftp”
        };
```

OutputSE

- **OutputSE – представляет URL того SE, где пользователь хочет сохранять выходные файлы. Используется RB для определения CE, “ближайшего”(close) к данному SE.**

Следует использовать осторожно, т.к. разные брокеры по-разному интерпретируют присутствие этого атрибута. Например LCG RB аварийно завершает задачу, если нет CE, определённого, как “ближайший” для OutputSE.

OutputData

- **Пока не реализовано в gLite 3.0**
- **Позволяет пользователю автоматически пересылать на SE и регистрировать в каталоге выходные файлы задания.**
- **Для каждого файла могут быть определены 3 атрибута:**
  - OutputFile (обязательный) – имя выходного файла
  - StorageElement (необязательный) – SE, где должен быть сохранён файл
  - LogicalFileName (необязательный) – LFN, под который должен быть зарегестрирован файл в каталоге

OutputData (пример)

```
OutputData = {
        [
                OutputFile = "dataset_1.out ";
                LogicalFileName = "lfn:/test/result1";
         ],
        [
                OutputFile = "dataset_2.out ";
                StorageElement = "se001.cnaf.infn.it";
         ],
        ]
                OutputFile = "cms/dataset_3.out";
                StorageElement = "se012.to.infn.it";
                LogicalFileName = "lfn:/cms/outfile1";
         ],
        [
                OutputFile = "dataset_4.out ";
         ]
        };
```