ELFI - Egrid L. Filesystem Implementation

Workshop on Porting Scientific Applications on Computational GRIDs

Antonio Messina
<antonio.messina@ictp.it>

16 febbraio 2006

Outline

1. What is ELFI
   - A problem in data management
   - Requirements
   - A proposed solution
2. Use cases
   - Use cases - UI
   - Use cases - WN
3. How ELFI works
4. Demo

A problem in data management

- Access to the GRID is more cumbersome then you want:

from the UI, through command line programs

```
lcg-cp <some long and esoteric options>
```

from the job, through the API

```
rfio_read(...);
```

- Legacy application cannot access grid stored data (they must download the **whole** file and after upload it)

Present situation (2)

Moreover: LCG middleware is continuously evolving. Presently 3 catalogs are available:

- RLS (being phased out)
- LFC
- FiReMan

. . . and a lot of transport protocols. . .

- RFIO
- GridFTP
- . . .

not to mention SRM. . . !

Requirements

What do we want from a client?

- It must be simple to use!
- It must hide the implementation, so the upgrade of the underlying layer must interests only developers.
- should be accessible also from legacy software which uses ordinary POSIX calls.

A proposed solution

ELFI is a filesystem access grid stored data as if it were local to your linux box.

- Start from user command line:

  ```
  elfi -l lfc,gsirfio /tmp/fuse
  ```

- Manages grid storage through LFN
- Access grid files through POSIX standard sycall
  - Download/upload file through unix commands (`cp/mv/...`)
- Modular design:
  - each protocol/server implemented in a separate DLL
  - chooses which DLL to load from command line
  - easy to add support for more protocols (rfio module is $\sim$ 200 lines of code)

Use cases - UI

An user can copy files from/to the grid through any program...

- directory listing

  LCG-utils `lfc-ls -l /dir`

  ELFI `ls -l elfi/LFC/dir`

- download a file from the GRID

  LCG-utils `lcg-cp --vo VO lfn://dir/file file://local`

  ELFI `cp elfi/LFC/dir/file1 /local`

- upload a file to the GRID

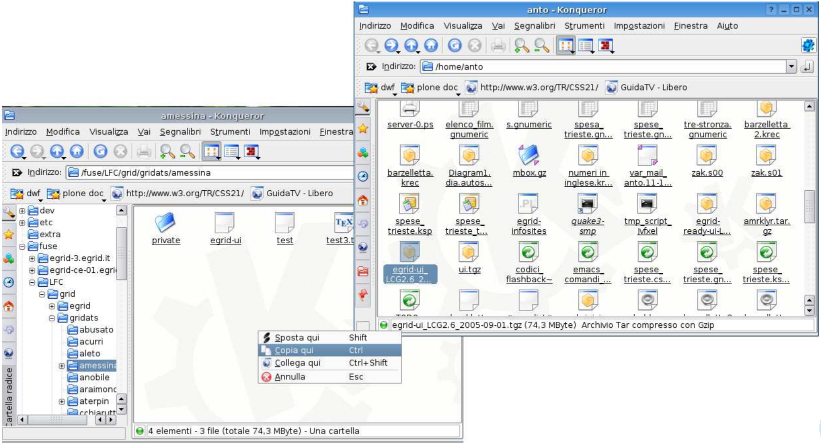
  LCG-utils `lcg-cr -d se-host -l lfn://dir/new file://local/file`

  ELFI `cp /local/file elfi/LFC/dir/new`

And also remove files, create replica, etc...

Use cases - UI (2)

You can run graphical filemanagers on the grid storage. . .



Indirizzo: /home/anto

egrid-ui_LCG2.6_2005-09-01.tgz (74,3 MByte) Archivio Tar compresso con Gzip

Indirizzo: /fuse/LFC/grid/gridats/amessina

4 elementi - 3 file (totale 74,3 MByte) - Una cartella

Use cases - WN

No need to adapt sources / recompile the application!

samplemain.c

```
main(){
        FILE *f;
        char ant[256];
        char *fn="elfi/LFC/path/to/file";
        f=fopen(fn,"r");
        while(!fscanf(f,"%s",ant))
                do_something(ant);
}
```

Submitted jobs can access data through standard POSIX syscall.

Use cases - WN (shortcomings)

- jobs need to call elfi on their own
    - at present, elfi is not natively supported by job manager or batch system scripts
- elfi needs FUSE system (kernel module + root-SUID binary + libfuse DLL)
    - system administrators must install a kernel module and a root-SUID binary (fusermount) on every WN
    - not included in Scientific Linux CERN to-date
    - but you may run “user mode linux” or Xen

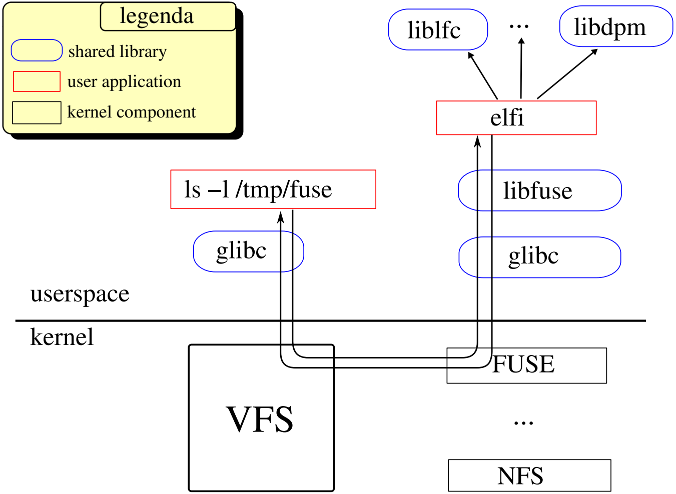
how ELFI works

- In the root directory of the ELFI filesystem there are one entry called **LFC** and one entry for each **SE**.
- On both **LFC** or **SE**'s directory only the Logical File Name are shown
- Inside the **LFC** directory tree all the files registered in the catalog are shown.
- Inside an **SE** directory tree only files which have a replica in that **SE** are shown.

Replica management

- When an existing file is opened inside the **LFC** directory ELFI chooses an **SE** in which a replica was found.
- When a file is created inside the **LFC** directory, ELFI chooses an **SE** to store the newly created file.
- When a file is opened or created inside one of the **SE** directory, ELFI try to use that **SE**.
- When a hard link is created inside an **SE** directory, a replica is added in that **SE**. (note: only LFC→SE or SE1→SE2 hard links are valid)
- If a file had been removed from the **LFC** directory then all the replicas are removed from the **SE**, and the LFN is unregistered from the catalog.
- If a file had been removed from a **SE** directory, then the replica inside that **SE** is removed, and if the file has no other replicas, then the LFN is unregistered from the catalog.

fuse schema


legenda
shared library
user application
kernel component
liblfc
...
libdpm
elfi
ls –l /tmp/fuse
libfuse
glibc
glibc
userspace
kernel
VFS
FUSE
...
NFS

ELFI Components

elfi Main program: collect access request from the fuse module, chooses what is to be done.

elfid Collect information about available **SEs** from the **BDII** and send it through a pipe to the elfi main program

libelfi_lfc.so library containing function to access to the **LFC** catalog

libelfi_gsirfio.so library containing function to access to the **SE** through GSI-RFIO transport protocol

Security

ELFI is as secure as the command line...

- Only the user that called ELFI can access the filesystem
- Access to the catalog or to the data is made through liblfc and libdpm APIs, which are GSI-enabled
- Access permissions to the metadata are checked by the **LFC** server
- Access permissions to the data are checked by the **RFIOD** server
- **root** cannot access the filesystem. Obviously root can become that user and so access the data
- No sensible information is sent through the network

Coming soon. . .

- Other transport protocols (only RFIO supported at present)
  - GFAL and gLite I/O already in the works
  - GridFTP should be do-able, a FUSE-based GridFTP filesystem is available on the FUSE website.
- Support for POSIX ACLs
  - some glue code needs to written
  - so coming soon
- SRM v1 and v2 support
  - hooks already present in code
  - will synch with StoRM SRM v2 server for release

Demo

let's try to break ELFI now ;-)