Grid File Access Library

Jean-Philippe Baud
IT/GD

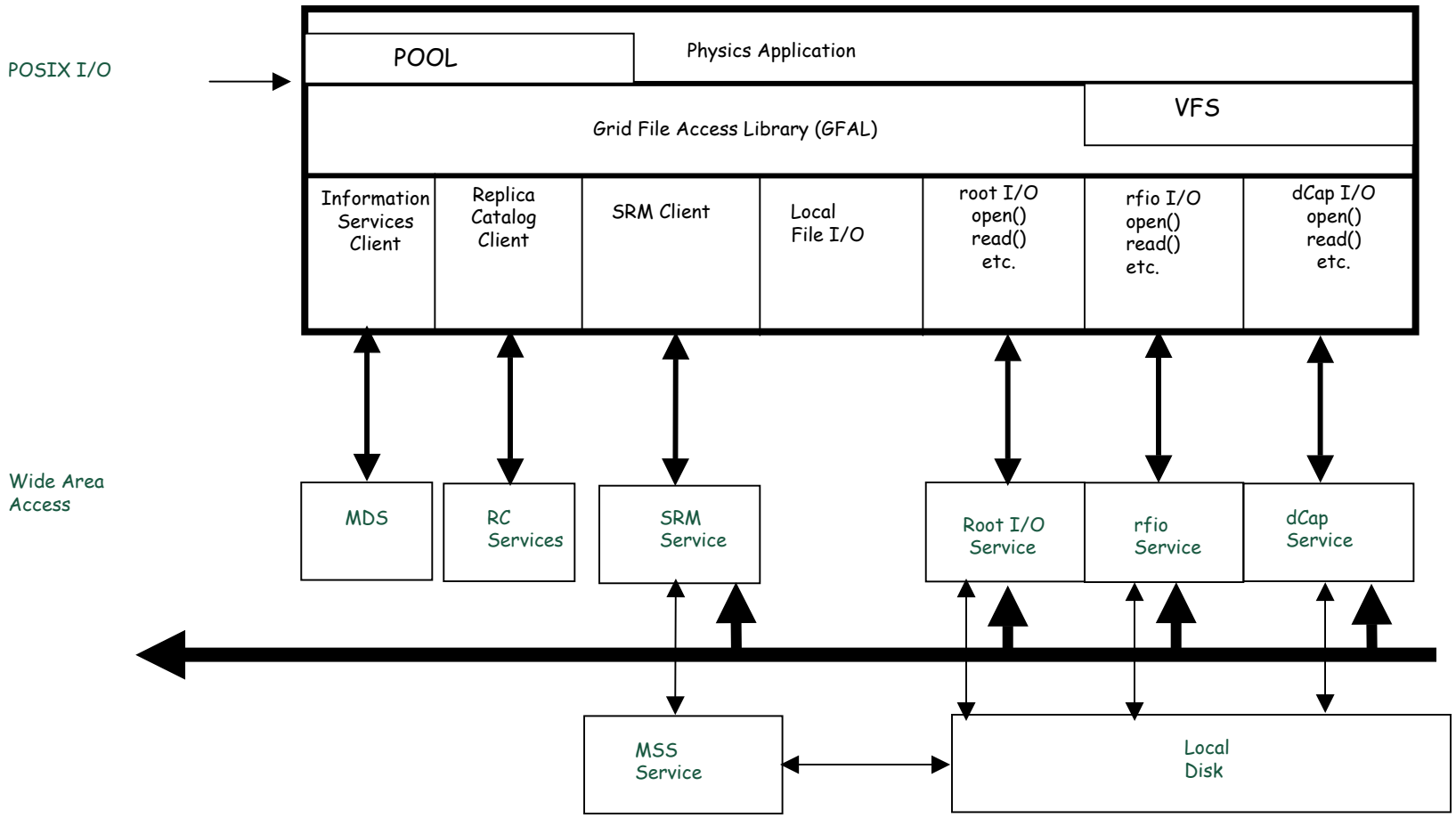
Goal

- Provide a Posix I/O interface to heterogeneous Mass Storage Systems in a GRID environment
- A job using GFAL should be able to run anywhere on the GRID without knowing about the services accessed and without re-linking

Services contacted

- Replica Catalogs
- Storage Resource Managers
- Mass Storage Systems through diverse File Access protocols like RFIO, DCAP, ROOT I/O
- (Information Services: MDS)

Flow diagram



GFAL possible implementations

- Library (archive and shared): need source modification and recompilation
- Pluggable File System
- VFS: LUFS and FUSE

Replica Catalogs

- GFAL currently only supports the EDG RMC and LRC
- Relatively easy to use Globus RLS: GFAL is very modular and the current RLS interface is automatically generated from the WSDL using gsoap + about 100 lines of C code

Storage Resource Managers

- Current implementation of GFAL requires version 1 of SRM installed on all Storage Elements
- The new version will support classic SEs with “file” protocol, using MDS to get information about mount points, but no disk pool manager
- However we recommend to have a true SRM installed on all SE
- An interim solution for sites not having an MSS would be to install a subset of CASTOR or dCache especially packaged for this (but support would have to come mostly from the community) or to develop their own SRM
- We also plan to provide a DRM based on SRM version 2.1 specifications for those sites

Current site status for SRMs

- CERN, FNAL have a working SRM to their MSS
- FZK is installing dCache
- INFN and PIC have CASTOR and would just need to have the CASTOR SRM installed
- RAL has an SRM implementation being tested with GFAL
- BNL has a working SRM/HRM in front of HPSS
- This leaves around 10 sites without SRM

Security

- Only the CERN and FNAL SRMs are protected by Grid certificates
- Replica catalogs are currently deployed in insecure mode
- RFIO does not handle yet Grid certificates (local access only)

File System

- LUFS implementation tried but problems with error reporting and concurrent access to files; author contacted but not very responsive
- GFALFS now based on FUSE (Filesystem in USErspace) file system developed by Miklos Szeredi
- Uses:
  - VFS interface
  - Communication with a daemon in user space (via character device)
  - Requires installation of a kernel module fuse.o and of the daemon gfalfs
  - The file system mount can be done at boot time

Library vs. File System

- Library:
  - Not transparent
  - Requires modifications to the source of the application to call the GFAL entry points
- File System
  - No modification to the source of the application
  - Possibility to use standard Unix utilities like ls, tar, …
  - Requires installation of a kernel module and a daemon

Current status

- GFAL library connecting to CERN and FNAL Mass Storage Systems(secure mode): ok
- Interfaces to EDG RLS (secure and insecure mode): ok

- GFAL filesystem at CERN and FNAL: basic tests ok

- GFAL and EDG SRM: in progress
- GFAL and Storage elements without SRM: in progress

Utilization

- The program must be linked to libgfal.a (static) or libgfal.so (dynamic)
- The application must set 2 environment variables:
  - setenv LCG_VO dteam
  - setenv LCG_BDII tbed0150.cern.ch:2170

Support

- GFAL library is very modular and is small (~ 1500 lines of C): effort would be minimal unless new protocols or new catalogs have to be supported
- Test suite available
- GFAL file system:
  - Kernel module: 2000 lines (FUSE unmodified)
  - Daemon: 1600 lines (FUSE unmodified) + 300 lines GFAL specific (separate file)
  - Utilities like mount: 600 lines (FUSE + 5 lines mod)

To be done

- Support for multi-threaded applications
- Test on Solaris (we use gcc to compile as the native SUN compiler rejects the empty structures generated by gsoap)
- Port to Windows 2000/XP
- Non Posix routines for optimization