DEVELOPING A ROOT INTERFACE FOR GLITE*

K. Schwarz, A. Manafov, P. Malzacher, GSI, 64291 Darmstadt, Germany

Abstract

The D-Grid [1] initiative, following similar programs in the USA and the UK, shall help to set up a nationwide German Grid infrastructure. Within work package 3 of the HEP [2] Community Grid distributed analysis tools under usage of grid resources shall be developed.

A starting point is the analysis framework ROOT [3]. A set of abstract ROOT classes (TGrid ...) provides the user interface to enable Grid access directly from within ROOT. A concrete implementation exists already for the ALICE Grid Environment AliEn [4]. We are developing an interface to the common HEP Grid middleware gLite [5]. This includes querying the gLite File Catalogue, access to individual files, job submission, getting the job status, and retrieving job output. Extensive tests will be done by using the gLite testbed of GSI

First interactive analysis environments have been set up at single sites using the Parallel ROOT Facility, PROOF [6]. This shall be extended towards several sites using existing Grid Middleware with the aim of a dynamic generation of Grid Analysis Clusters.

THE D-GRID PROJECT

The D-Grid project has been announced by the German Ministry of Science and Education during the 10th Global Grid Forum in March 2004 in Berlin. It follows similar programs in the UK and the USA and shall help to build a Grid infrastructure in Germany. The project structure supports six community Grid projects, namely AstroGrid, C3 for climate research, InGrid for engineering, MediGrid, TextGrid for humanities, and HEP-Grid, the Grid for the High Energy Physics community, as well as a comprehensive integration project DGI [7]. It should be possible to integrate the results of middleware developments within the community projects into the core D-Grid infrastructure. This way an application neutral grid platform shall be created and the approach to Grid technology should become facilitated for new communities. The resulting platform has to fit to international Grid projects though, especially to EU Grid projects like EGEE [8]. With this architecture community specific applications and requirements are supported as well as the development of a sustainable basic Grid Infrastructure, which shall be applicable for the whole German Scientific Community and help to establish methods of e-science in Germany.

The HEP-Grid community project

The HEP Community Grid project has been inspired by the LHC Computing Grid [9] of the 4 LHC [10] experiments at CERN. The software developments within this sub project shall concentrate on aspects which are considered important by Particle Physics and still need development efforts. The main points concentrate on dynamic data management, job scheduling, accounting and monitoring as well as the creation of tools for interactive data anysis on the Grid by individual users. The latter will be described more closely in this paper. The first task is to look for gaps in current interactive analysis systems. Since GSI is a member in the ALICE collaboration the work concentrates on software used by the ALICE experiment (ROOT/PROOF). Lateron existing ALICE specific solutions (TAliEn, an interface between ROOT and AliEn) shall be generalised by developing an interface between ROOT and the more general Grid middleware gLite.

In Munich similar work is performed for the ATLAS experiment [11].

THE ROOT - GLITE INTERFACE

The ROOT System

A starting point is the analysis software package ROOT, which is largely used within the HEP environment. Especially for members of the ALICE experiment ROOT provides the starting point for basically everything. Since the ALICE framework AliRoot [12] is based directly on ROOT, an ALICE user can do data simulation, reconstruction and analysis all by using the same framework and from within the ROOT-system. Additionally a working interface between ROOT and the ALICE Grid Environment AliEn exists so that also Grid-access from within ROOT is possible. According to the ALICE analysis model it is planned to provide a prompt analysis facility at CERN by using PROOF, the Parallel ROOT Facility, which enables parallel and interactive analysis of large data sets situated at one site. Also batch style analysis of distributed data sets by using ROOT and existing Grid infrastructures as well as interactive analysis of distributed data by combining PROOF and Grid should be possible.

To be able to realise all features of the analysis model described above, it is necessary to provide general Grid-access directly from ROOT, which is a well known environment for most Physicists working for ALICE or in the field of High Energy Physics. As a Grid middleware of general interest gLite has been chosen since it is planned to

* This work is supported by BMBF Förderkennzeichen 01AK802G

become the basis for all LHC experiments in near future.

For accomplishing this, ROOT offers a set of abstract base classes (Fig. 1), which provides an interface between ROOT and the Grid. A concrete implementation exists already for the ALICE Grid environment AliEn. An interface to the more general Grid middleware gLite is currently being created. The interface will include inquiring the file catalogue, access to individual files in the Grid, submission of jobs, inquiring the status and receiving the output of jobs.

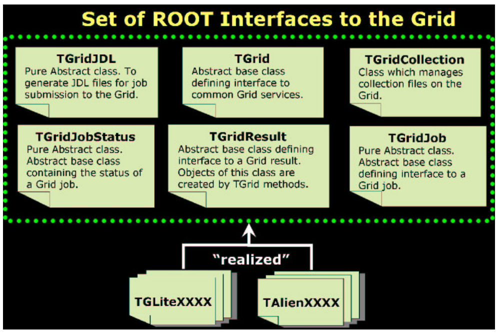

Set of ROOT Interfaces to the Grid
TGridJDL
Pure Abstract class. To generate JDL files for job submission to the Grid.
TGrid
Abstract base class defining interface to common Grid services.
TGridCollection
Class which manages collection files on the Grid.
TGridJobStatus
Pure Abstract class. Abstract base class containing the status of a Grid job.
TGridResult
Abstract base class defining interface to a Grid result. Objects of this class are created by TGrid methods.
TGridJob
Pure Abstract class. Abstract base class defining interface to a Grid job.
"realized"
TGLiteXXXX
TAlienXXXX

Figure 1: A set of abstract base classes providing an interface between ROOT and the Grid.

TGLite

The concrete implementation of TGLite uses the gLite WMProxy C++ API [13]. The WMProxy is a simple service providing access to the WMS functionality (Workload Management System) of gLite through a Web Services based interface. It accepts job submission requests described with the job description language (JDL) and other job management and control requests such as job cancellation, job output retrieval etc. Compared to the older WM UI API it provides additional functionality such as bulk job submission and support for shared and compressed sandboxes. The WMProxy client API supplies the client applications with a set of corresponding interfaces and includes a Brokerinfo Access API, via which information from the Resource Broker can be retrieved. Also a simple API for the management of job identifiers is also provided.

The current implementation of TGLite does not use the API of gLite directly, though. Instead a glite-api-wrapper library which contains all necessary information is being created and used. The schema is displayed in (Fig. 2).

Currently the glite-api-wrapper library, ROOT and the TGLite-interface have to be installed on a gLite User Interface. After a grid-proxy-init, the user can start ROOT and connect to the Grid. An example session is shown in Fig. 3. The user connects to the Grid (gLite or AliEn), gets the list of files to be analysed from the Grid file catalogue, and fills the result into a ROOT Chain. In case a set of PROOF daemons have been submitted beforehand a connection to a remote PROOF master can be accomplished from the ROOT client on the Grid UI and a PROOF session analysing several files in the Grid can be initiated. Intensive tests are currently being performed on the gLite testbed which has been set up at GSI on a number of virtual XEN machines.

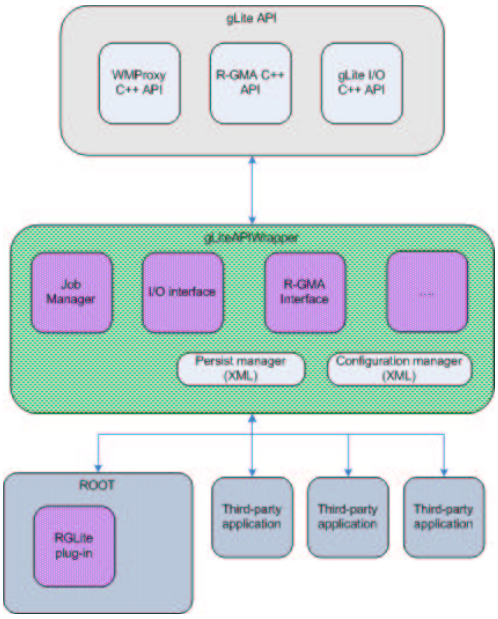
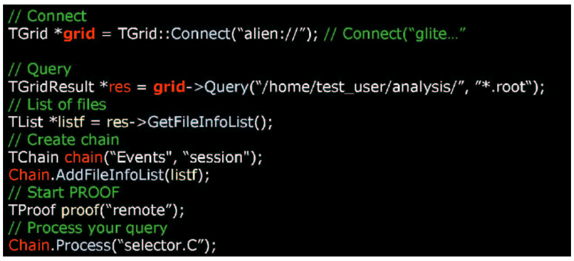


Figure 2: The ROOT application uses the gLite-API via a wrapper library in between.



```
// Connect
TGrid *grid = TGrid::Connect(“alien://”); // Connect(“glite...”

// Query
TGridResult *res = grid->Query(“/home/test_user/analysis/”, “*.root“);
// List of files
TList *listf = res->GetFileInfoList();
// Create chain
TChain chain(“Events”, “session”);
Chain.AddFileInfoList(listf);
// Start PROOF
TProof proof(“remote”);
// Process your query
Chain.Process(“selector.C”);
```

Figure 3: An example session running PROOF on the Grid.

REFERENCES

[1] http://www.d-grid.de.

[2] http://www.d-grid.de/index.php?id=44.

[3] http://root.cern.ch.

[4] http://alien.cern.ch/twiki/bin/view/AliEn/Home.

[5] http://glite.web.cern.ch/glite/.

[6] http://root.cern.ch/root/PROOF.html.

[7] http://dgi.d-grid.de/.

[8] http://public.eu-egee.org/.

[9] http://lcg.web.cern.ch/LCG/.

[10] http://lhc.web.cern.ch/lhc/.

[11] J. Elmsheuser, Distributed Analysis Experiences within D-Grid, this proceedings.

[12] http://root.cern.ch/root/R2000/Html/AliRoot/index.htm.

[13] http://egee-jra1-wm.mi.infn.it/egee-jra1-wm/glite-wmproxy-api-index.shtml.